Large Scale Partial- and Near-Duplicate Image Retrieval using Spatial Information of Local Features

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieurin (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von M.Sc. Afra'a Ahmad Alyosef

geb. am 28.06.1980 in Aleppo, Syrien

Gutachterinnen/Gutachter

Prof. Dr.-Ing. Andreas Nürnberger
Prof. Dr.-Ing. Klaus Tönnies
Prof. Dr. Anders Hast

Magdeburg, den 20.11.2023

Abstract

The rapid development of smart devices and cameras together with the increased use of social media that support image sharing results in large amounts of redundant images. Many of these images can be characterized as near-duplicate images, that present the same scene but are captured by different users with different cameras, resolutions, scales or viewpoints. Moreover, image editing apps to modify and enhance images can be easily applied and produce even more near-duplicates, sometimes – intended or unintended – infringing copyrights of the creator. Therefore, methods to efficiently detect the correspondence between these near-duplicates in large datasets are valuable for commercial or legal reasons as well as for personal use as an aid to organize large image collections.

Existing methods approach the problem of near-duplicate detection by either improving the local features, that describe specific areas in images, or by combining the benefits of two or more types of local and global features, that report details about all areas in an image. Unfortunately, these methods either tend to retrieve non-relevant images on top of their result lists or have high computational requirements. Recent techniques to construct panorama images or detect copyright infringements utilize the spatial correlation among the corresponding features linking images. These techniques suffer from either low accuracy or expensive computation costs as well.

In this thesis, we propose methods to improve and accelerate the usage of local features in algorithms for near-duplicate image retrieval. In addition, to enhance rankings of images in the retrieved list, we introduce an approach that combines information from local and global features to filter false positives. To determine the spatial correlation between near-duplicate images, we propose an algorithm that identifies whether compared images are similar – or have a similar region – and concurrently derive the kind of similarity – or the geometrical transformation – between them. The proposed algorithm does not require any prior knowledge about the content or image domain.

We compare our algorithms to several existing state-of-the-art approaches for different retrieval tasks, namely, accelerating image retrieval, enhancing the ranking of relevant images in the retrieved list, rejecting non-relevant images and estimating the spatial correlation between images. Our experiments show that, in most cases, the performance of the proposed methods outperforms other hand-crafted approaches in some cases while having even lower computational costs.

Zusammenfassung

Die rapide Entwicklung von Smart-Devices und Kameras in Kombination mit der gestiegenen Nutzung sozialer Medien - und deren Möglichkeiten Bilder untereinander auszutauschen - resultiert in enormen Mengen an redundanten Bildern. Viele dieser Bilder können als sog. Near-Duplicates bezeichnet werden, da sie die selbe Szene darstellen, jedoch durch verschiedene Nutzer mit unterschiedlichen Geräten, Auflösung, Maßstab oder Betrachtungswinkel aufgenommen wurden. Zahlreiche Anwendersoftware zum einfachen Editieren, Modifizieren und Verbessern von Bildern erhöhen die Anzahl der Near-Duplicates weiterhin und verletzen dabei manchmal – gewollt oder nicht – das Urheberrecht. Aus diesem Grund erweisen sich Methoden zur effizienten Detektierung der Übereinstimmung zwischen solchen Near-Duplicates, besonders in großen Datensätzen, als wertvolles Mittel sowohl aus kommerziellen als auch rechtlichen Gründen. Aber auch für den persönlichen Gebrauch bieten derartige Methoden ein Hilfsmittel zur Organisation großer Bildsammlungen.

Aktuelle Methoden näheren sich dem Problem der Near-Duplicate Erkennung entweder durch die Verbesserung lokaler Bildmerkmale oder versuchen die Vorteile von mehreren lokalen und globalen Merkmals-Typen zu kombinieren. Erstere beschreiben spezifische Bereiche in den Bildern, Zweitere nutzen die Beschreibung von sämtlichen Details im ganzen Bild. Nachteile der aktuellen Methoden äußern sich entweder im Auffinden vieler nicht relevanter Bilder in den obersten Suchergebnissen – wenn die Erkennung als Retrieval Problem betrachtet wird - oder haben hohe Rechenkosten zur Folge. Die jüngsten Techniken, um z.B. Panorama-Bilder zu konstruieren oder um Urheberrechtsverletzungen zu finden, verwenden die räumliche Übereinstimmung zwischen den korrespondierenden Merkmalen, die die Bilder miteinander verknüpfen. Aber auch diese Techniken haben Nachteile in puncto geringe Genauigkeit oder hoher Rechenkosten.

In dieser Arbeit werden neue Methoden vorgestellt, welche die Verwendung lokaler Bildmerkmale beschleunigen und verbessern, um so zur Lösung der Aufgaben des Near-Duplicate-Retrievals beizutragen. Um zusätzlich die Rangfolge der Near-Duplicates in der Suchergebnisliste zu verbessern, wird ein optimierter/neuer Ansatz vorgestellt, der die lokalen und globalen Merkmale kombiniert und so die False-Positives minimiert. Um die räumliche Übereinstimmung zwischen Near-Duplicates zu bestimmen, wird weiterhin ein Algorithmus beschrieben, der feststellt, ob die miteinander verglichenen Bilder ähnlich sind oder ähnliche Bereiche aufweisen. Dabei wird gleichzeitig die Art der Ähnlichkeit sowie die Art der Modifikation (geometrische Transformation) bestimmt. Ein weiterer Vorteil des Algorithmus ist, dass kein Vorwissen über den Bildinhalt oder die Bilddomäne erforderlich ist.

Die hier vorgestellten Algorithmen und Methoden werden mit verschiedenen, bereits existierenden State-of-the-Art Ansätzen bzgl. der Beschleunigung des Bild-Retrievals, der Verbesserung der Rangfolge der relevanten Bilder in der Ergebnisliste, dem Aussortieren von nicht relevanten Bildern und der Abschätzung der räumlichen

Übereinstimmung zwischen den Bildern verglichen. Die dafür durchgeführten Experimente zeigen, dass in den meisten Fällen die in dieser Arbeit vorgestellten neuen Verfahren die bisherigen Ansätze übertreffen oder zumindest die gleichen Ergebnisse liefern und dabei sogar teilweise geringere Rechenkosten verursachen.

Contents

Glossary

BRIEF Binary Robust Independent Elementary Features.

BRISK Binary Robust Invariant Scalable Keypoints.

CNN Convolutional Neural Network.

COP Combined-Orientation-Position.

COTA COngruent Triangles Approach.

DoG Difference of Gaussian.

DWT Discrete Wavelet Transform.

ECOTA Extended COngruent Triangles Approach.

F-HS Fuzzy HS model.

F-HSV Fuzzy HSV model.

FAST Features from Accelerated Segment Test.

FP-HS Fuzzy Partition HS model.

FP-HSV Fuzzy Partition HSV model.

GN Gaussian White Noise.

GOODSAC GOOD Sample Consensus.

HS Hue Saturation color histogram.

HSV Hue Saturation Value color space.

HSV-SIFT The Hue Saturation Value colored SIFT descriptor.

k-d tree k-dimensional tree.

L1 Manhattan distance or $L1$-norm.

L2 Euclidean distance or $L2$-norm.

L***a*****b*** Lightness, Red/Green, Blue/Yellow color space.

LMEDS Least MEdian Squares.

MAP Mean Average Precision.

MLESAC Maximum Likelihood Estimation by SAmpling Consensus.

MP Mean Precision.

MPN Multiplicative Noise.

MR Mean Recall.

MSER Maximally Stable Extremal Regions.

NAPSAC N Adjacent Points SAmple Consensus.

ND Near-duplicate.

ORB Oriented Fast and Rotated BRIEF.

PCA Principal Component Analysis.

PCA−SIFT Principal Component Analysis of Scale Invariant Feature Transformation.

PROSAC PROgressive SAmple Consensus.

PUMA Putative Match Analysis.

RANSAC RANdom Sample Consensus.

RC Region Compressed.

RC-SIFT Region Compressed SIFT.

RGB Red Green Blue color space.

RGB-SIFT The Red Green Blue colored SIFT descriptor.

rgSIFT The Red Green colored SIFT descriptor.

RoI Region of Interest.

SIFT Scale Invariant Feature Transformation Algorithm.

SPN Salt and Pepper Noise.

SURF Speed Up Robust Feature Detector and Descriptor.

U-SURF Upright-Speed Up Robust Feature Detector and Descriptor.

VR Variance of Recall.

WaldSAC RANdom Sample Consensus based on Wald's theory.

YCbCr Y: Luminance or brightness Cb: Chrominance difference of blue component and Cr: Chrominance difference of red component.

List of Figures

List of Tables

1 Introduction

Content-based near-duplicate (ND) image retrieval becomes more challenging in the last decade due to the dramatic increase in the size of image datasets, the use of images without any textual description, and the widespread use of image altering tools. ND-image retrieval has application in various fields such as construction for panoramas (i.e. stitching images that associate with a specific scene), copyright violation detection, and logos tracking. In recent years, various techniques have been developed to accelerate image categorization and retrieval. Some techniques process images based on analyzing the textual description around them others describe images based on specific properties of their content such as color, texture, blobs, or shapes. The representation of images based on various properties is necessary since one sort of property ignores many details that could be important to solve specific tasks. These properties are called image features. The extraction of features depends on the task and the kind of images e.g., for gray-scale images no meaning to use color features, to retrieve all images that contain a blue car, color features are not enough since all images that contain blue objects will be retrieved. Therefore, the similarity between queries and dataset images and the rank of retrieved images depends on the type of the extracted features.

The most significant components in designing an image retrieval system are feature extraction, index construction, feature matching, retrieval and ranking of results, and performance evaluation. In a near-duplicate retrieval system, an additional stage is essential, that is, spatial correlation detection. Feature extraction is the primary process of retrieval systems. It projects the high dimensions and complex content of images into feature space. Feature space, comparing to image space, is "low dimension space", which presents images utilizing one or more of their properties. The second step in the retrieval system is feature structuring, where similar features are aggregated together using various techniques. This step aims to speed up and simplify the matching of the high-dimensional features. After completing this step, the structured features are stored and used to identify the similarity with any given query image. Finally, the system should be evaluated and updated based on the requirement of the user. In the case of a near-duplicate retrieval system, a necessary step is to identify the correlation between a query image and the list of retrieved ones. This is important to exclude the non-relevant images of the retrieved list and to estimate the exact spatial transformation between the ND-images. We suggest utilizing this procedure after the retrieval step since there is no meaning of predicting the correlation with all images in the dataset when we know previously that only a very small set of them are near-duplicates to the query image.

In the scope of this thesis, we aim to improve the near-duplicate retrieval system in three stages i.e. feature extraction, retrieval improvement, and correlation detection through understanding content of images and analyzing the methods of feature extraction to declare how the similarity between two images is found and why are

they similar. For this we do not need any training stage since we process images without any prior knowledge about their content. Therefore, we do not use or compare with the deep learning techniques in our work.

1.1 Motivation

The milestones in improving image near-duplicate retrieval systems as presented in this thesis are: to speed up the retrieval process, enhance the retrieval of the relevant images, filter out the non-relevant images of the retrieved list, and determine the exact spatial correlation between near-duplicate images. The acceleration of the retrieval step causes the loss of valuable details about the similarity between features. This decreases the performance of the retrieval system too. The similar global appearance of some images causes the retrieval of non-relevant ones on top of retrieved results. Since they may have comparable colors and illuminations or are taken from similar viewpoints like query images. To overcome this issue, the correlation detection between images has been introduced based on the information about feature matches. The challenge in this case is the occurrence of false matches. The more false feature matches are obtained, the higher is the chance to estimate wrong or no correlation between images even when they are near-duplicate. The reason is the usage of non-deterministic models that employ "raw" feature matches without any pre-processing step. However, the deterministic methods that detect and exclude the false matches do not solve this problem since most of them filter out a subset of correct feature matches as false ones and they require long computation-times.

To overcome these problems, we introduce an innovative technique to concurrently speed up the retrieval step and preserve the robustness of detected features. Moreover, we develop a method to combine the advantages of global and local features to accelerate the retrieval process and improve the near-duplicate images ranking in the retrieved list. To describe the relationship between the query and retrieved ND-images, we develop our own "deterministic" algorithm to first filter the false matched features and avoid their influence on the correlation computation phase. Second, exploit location details of correct matches to predict the spatial correlation between images. We improve our method in a manner that minimizes the required time and memory usage to perform correlation detection. Based on our algorithm, we figure out the exact relationship between two ND-images. We extend our algorithm to filter out the non-relevant images from the retrieved list. Moreover, our method gives a plausible explanation of the correlation between ND-images.

1.2 Research Question

This thesis answers the following research questions. An overview of the belonging processing steps in a retrieval system is given in Figure 1.

RQ.1 How can we improve keypoint feature extraction to:

(a) Accelerate image near-duplicate retrieval.

(b) Preserve the invariant and robust properties of the keypoint features.

(c) Reduce the amount of utilized features.

RQ.2 How can we improve the ranking of the retrieved list of near-duplicate images?

(a) Can the combination of more than one type of feature improve the near-duplicate retrieval?

(b) Can we accelerate the retrieval process through feature combination?

RQ.3 How can we improve correlation detection between near-duplicate images?

(a) Can we determine the correlation between near-duplicate images based on the detected correlation?

(b) Can we apply the detected correlation to exclude the non-relevant images of the retrieved set.

1.3 Near-Duplicate (ND) Images

The main goal of this thesis is to improve near-duplicate image retrieval. Therefore, in this section, we first clarify the concept of a *near-duplicate image*. We distinguish between three types of duplicate images:

- **Type1: Exact-duplicate images:** Two images are considered as exact duplicates iff they are identical [46], i.e. the corresponding pixels are identical (in color and intensity).
- **Type2: Near-duplicate images:** In general [188], [46] two images are defined as near-duplicates if they show the same scene (the same object) but they differ (slightly) in:
  - Some processing steps (such as noise, blurring, compression, contrast etc.).
  - Time conditions (e.g. lighting change).
  - Transformations (e.g. affine transformation described in Subsection 2.2 or viewpoint perspective.
- **Type3: Partial-duplicate images:** Two images are partially duplicate if they show identical regions, objects, or logos as parts of both of them [184]. One of them can be the original image and the other is the faked one. In some cases, both of them are faked, but they share the same region taken of another original image. The challenges in partial-duplicate images are that the shared regions contain additional altering like affine transformations, viewpoint or lighting changes, adding blur or noise [25].

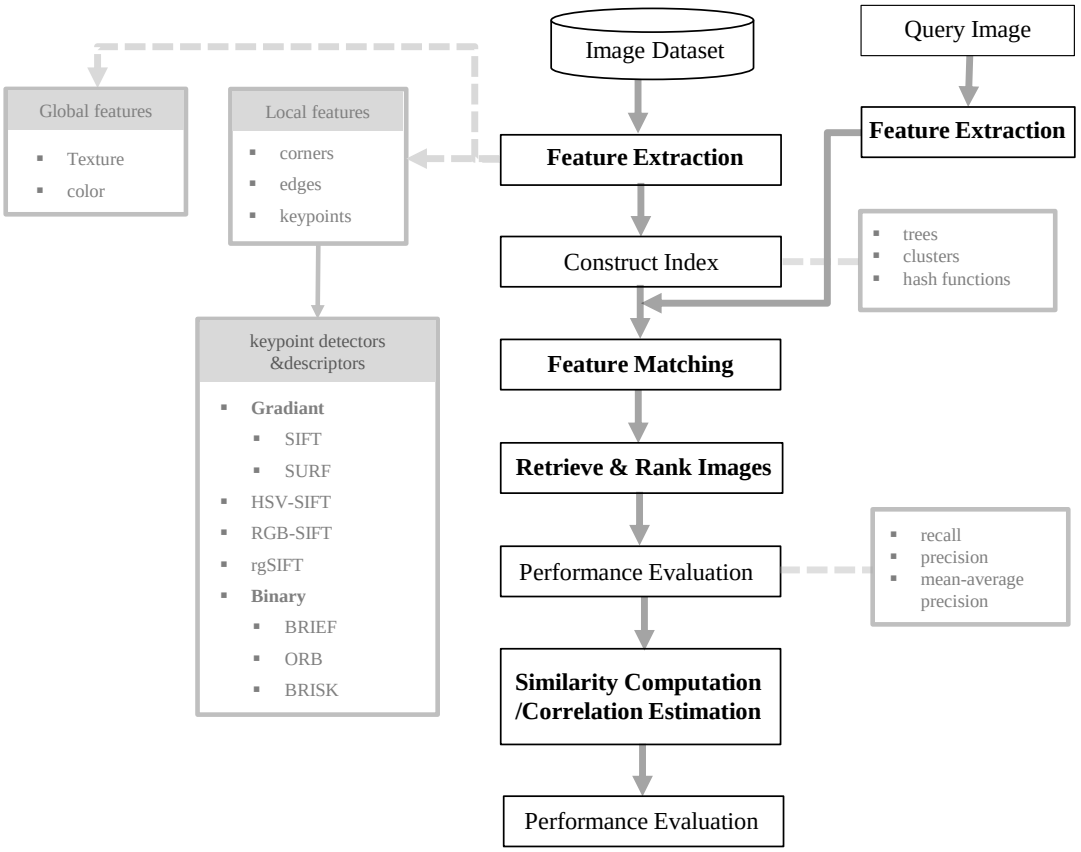


Figure 1: Near-duplicate image retrieval system. The bold parts present the stages we are focusing on in scope of this thesis. We work deeply on three main phases i.e. improve feature extraction (see Chapter 5), improve the ranking of retrieved results (see Chapter 6) and estimate the correlation between ND-images (see Chapters 7 and 8).

Near- and partial-duplicate have many significant applications in multimedia linking such as threading news stories, query by example applications [184], [25], panorama-images construction and copyright infringement detection [194], [188]. However, it is not determined in the previous researches the range of the applied transformations in which images are still considered near-duplicate. In this thesis, the term *near-duplicate* refers to types 2 and 3 of near-duplicate images, i.e. *near- and partial duplicate images*. Figure 2 presents samples of near-duplicate images types 2 and 3.

1.4 Thesis Outline

This thesis discusses the improvement of image near-duplicate retrieval in three stages, the first is feature extraction improvement. The goal of this stage is to accelerate and enhance ND-retrieval. The second is combining more than one kind of features to improve the rank of retrieved and relevant ND-images in the retrieved

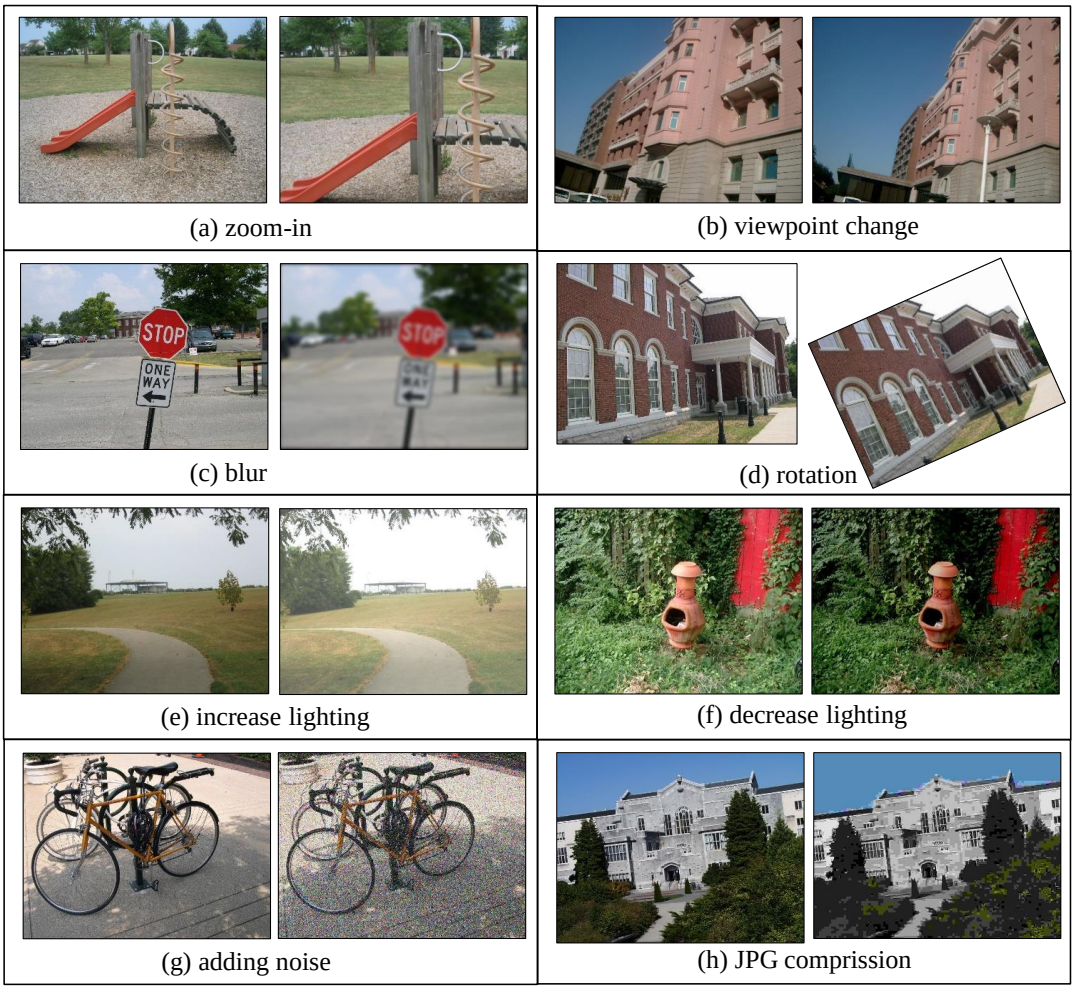


Figure 2: Samples of near- and partial-duplicate images.

list. The third is spatial correlation detection between ND-images to clarify whether a retrieved image is relevant to a given query. Figure 1 presents the main stages of the near-duplicate retrieval system. The boldly marked steps show the package that we discuss in this thesis. We work intensively on those stages since they have significant effects on the performance of near-duplicate retrieval systems.

The outline of this thesis is structured as follows. Chapter 2 presents the relevant fundamentals and the important concepts and algorithms that are utilized in this work. Chapter 3 discusses some recent studies that approached near-duplicate retrieval problems by either enhancing the feature extraction step, combining various kinds of features or identifying the correlation between images. The main parts of this thesis are presented in chapters: Chapters 5, 6, 7 and 8. Chapter 5, discusses and solves research questions **RQ.1**(a) and (b) by introducing our adaptation of the feature extraction stage by improving local features for solving near-duplicate Retrieval tasks. In addition, we analyze the influence of feature properties on the performance

of near-duplicate retrieval tasks and hence we clarify research question **RQ.1**(c). Chapter 6 presents our hybrid approach by combining the global and Local features to accelerate the retrieval process (so we elucidate research question **RQ.2**(b)). Moreover, we enhance the ranking of the relevant images in the retrieved list thus, we find out a suitable approach to tackle research question **RQ.2**(a). The answers of the **RQ.3**(a) and **RQ.3**(b) are given in Chapter 7 by the proposing our algorithms to estimate the spatial correlation between the near-duplicate images. In Chapter 8, we combine and extend our approaches presented in Chapters 5 and 7 to solve interactive near-duplicate retrieval problems. Finally, we summarize our thesis and present the possible future works in Chapter 9.

2 Fundamentals

The goal of the thesis is to improve the solving of image retrieval tasks. Therefore, in this chapter, we explain the basic concepts and algorithms that we employed to achieve our goal. As shown in Figure 1, the first step in content-based image retrieval systems is to extract ***features*** of images. These features present the significant extractable characteristics of an image constructed from the original raw input data of images. These features form the abstracted level of images that simplify the solving of specific tasks, such as image classification, pattern and object recognition, image retrieval, near-duplicate identification, ... etc. The usage of the suitable type of features is determined through the goal of the study [137], [77], [180]. Image features are classified into global and local features. ***Global features*** describe an image as a whole and are computed by exploiting information of all pixels. Whereas, ***local features*** describe specific areas of images. Both global and local features have been used in image retrieval systems. However, local features are preferable in solving tasks of retrieving images that belong to the same scene since they describe specific details and position in images [93], [128]. We present the employed local and global features extraction algorithms in Section 2.3. Through the discussion of feature extraction, two relevant concepts are employed. These are the *scale space* and *affine transformations*, therefore, we introduce these concepts in Sections 2.1 before reporting feature extraction algorithms.

After feature extraction step, the similarity between the extracted features of query and database images is computed to determine the similarity score of images to a given query. The suggested metrics to calculate these similarities are presented in Subsection 2.5.1. To accelerate the matching process, methods to structure features have been used in various researches. We explain shortly in Section 2.4 these methods since we only use them without any change. Finally, to evaluate the image retrieval system as a whole, we present the common evaluation measures in Section 2.5.2.

2.1 Image Scale-Space

The importance of scale-space comes from the fact that objects in the real world appear as recognized entities only over a specific range of scales (depending on the distance between the perceived object and the observer) [111], [109]. The different appearances of the same objects obtain different images of them. To simulate this difference, Gaussian kernels of various sizes are employed. This issue is clarified in Figure 3(a) and (b), where Gaussian kernels of sizes 7×7 and 73×73 are convolved with the input image to detect the most important details in the image. However, when a Gaussian kernel of bigger size is employed (i.e. of size 73×73), the fine detected details in Figure 3(a) vanish. Only the details, that have bigger intensity than their sounding, appear in this stage, as shown in Figure 3(b). So the challenge here is how to specify the suitable kernel to detect invariant features to scale change. This problem has been discussed in the earlier research of signal

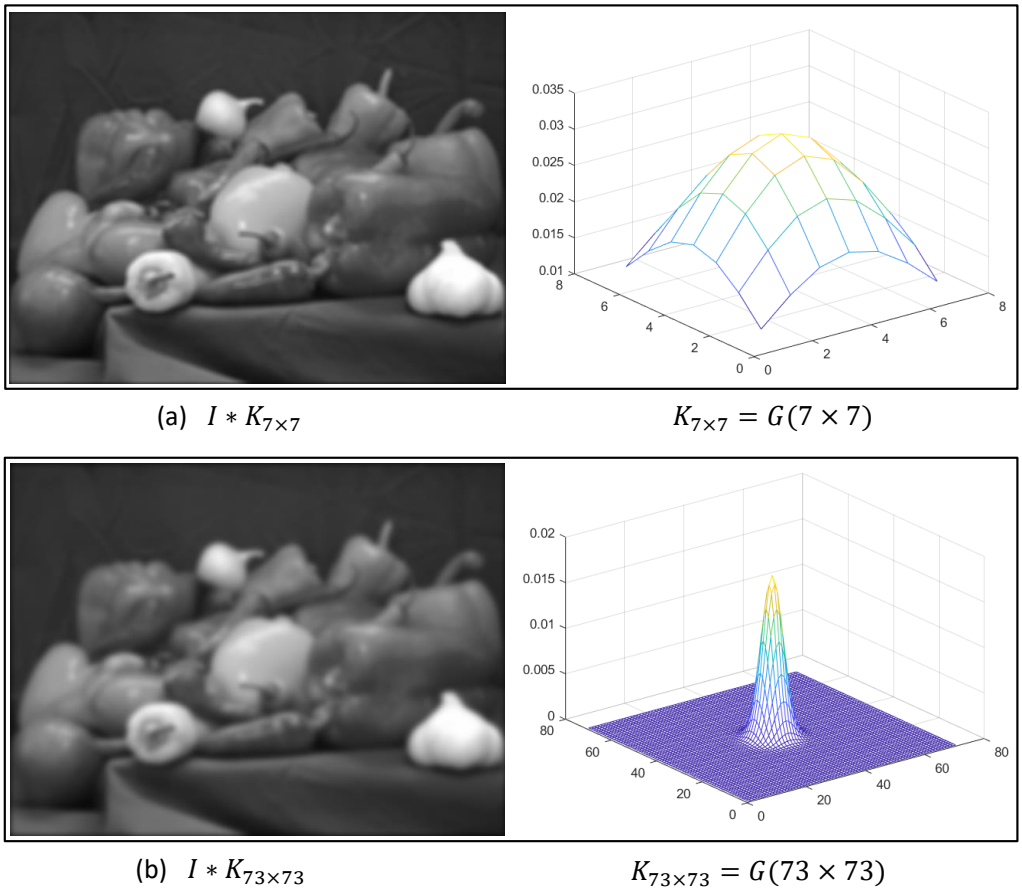
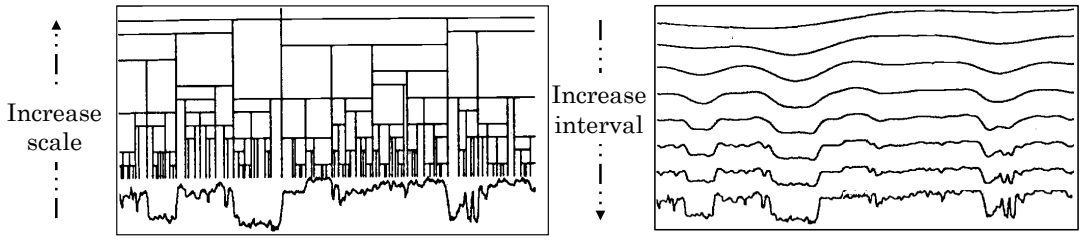


Figure 3: The convolution of the gray-scale ''pepper'' image with Gaussian kernels of various sizes. (a) Gaussian kernel of size 7×7 and (b) of size 73×73. The shapes of kernels are presented near to the convolved images.

processing. Gaussian Scale-space has been firstly modeled as a research result by Taizo Iijima at 1959 [179] and published in at 1962 [87]. The scale-space has been proposed firstly for one-dimensional signals [87]. The building of scale-space in tree form for a given signal has been described in [181]. The idea of interval-tree provides multiple descriptions of a one-dimensional signal. Figure 4 presents the fine to coarse levels of signal presentation i.e. the scale-space [181]. It has been proven that the building of scale-space employing the Gaussian filter is convenient to extract invariant features under changes in light shift and scale conditions [110], [164]. Therefore, scale-space filtering theory has been employed in many research to extract scale and transformation invariant features [116]. These invariant features were called *keypoints* and used to detect the similarity between images. The details of the invariant scale feature extraction methods are presented in Subsections 2.3.4, 2.3.5 and 2.3.6.



(a) Interval tree (b) Scale-space of a signal

Figure 4: The concept of space-scale to filter a one-dimensional signal. (a) Interval tree of the given signal. (b) The results of convolving the one-dimensional signal with Gaussian filter at different scales [181].

2.2 Image Affine Transformation

Affine transformation of an image is a linear geometrical function that alters its pixels but preserves the co-linearity and parallelisms properties of its pixels [122], [89]. Affine transformations of an image include translation, scaling, rotation, reflecting and shearing [199], [89]. Affine transformations are presented utilizing a 3×3 matrix with six freedom degrees for scaling, shearing, rotation, translation and reflection. To transform a pixel (x, y) of an image I to pixel $(x`, y`)$ we write them firstly using the homogeneous coordinates [182], [131], [73] (in the homogeneous coordination system the point (xz, yz, z) is the homogeneous coordinates of a plan point (x, y) where $z \in R^*$). After that, we apply the affine transformation matrix:

$$\begin{bmatrix} x` \\ y` \\ 1 \end{bmatrix} = \begin{bmatrix} (-1)^{b_x} s_x cos\theta & -c_x sin\theta & t_x \\ c_y sin\theta & (-1)^{b_y} s_y cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

where b_x and b_y are binary parameters that present the reflection cases, s_x, s_y stand for scale, c_x, c_y for shearing and t_y, t_y for translation in x and y directions, respectively. θ is the rotation angle. We present the transformation matrix in general i.e. as composition of all possible transformation but there are simple forms of this matrix when only one type of affine transformations is applied. The following matrices present the cases of scaling, rotating, reflecting, or shearing of an image, respectively

$$A_{scale} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad A_{rotation} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$A_{reflect} = \begin{bmatrix} (-1)^{b_x} & 0 & 0 \\ 0 & (-1)^{b_y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad A_{sheer} = \begin{bmatrix} 0 & c_x & 0 \\ c_y & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

The results of applying various kinds of affine transformation on an image are presented in Figure 66 in Appendix A. The input image in this case contains only one object, that is the letter F [48].

2.3 Image Feature Extractors

Color and gradient features, as global features, present the entire image in one vector. Therefore, they perform weakly in ranking benchmark images based on their similarity with a given query image. Edge and corner features are relative concepts since their detection depends on the size of the considered area or so-called *window* or *kernel* around it. These features are not robust to small changes in color or scale, therefore they are called weak features. More types of features have been developed, that are robust to various kinds of image transformations. These features are extracted through a scale-space (see Subsection 2.1). They are considered as invariant and strong features and are called *blob* or *keypoint* features. Keypoints describe the distinction of specific areas of an image of their surround.

The keypoint detector and descriptor techniques have been proposed to solve image retrieval, near-duplicate retrieval, and object recognition tasks. Contrary to the corner and edge features, keypoints present information about specific pixels and their surrounding regions. The robustness of keypoints comes from the methodology of their extraction. They are identified at different scales of an image. Various methods have been introduced to extract image keypoints. These methods are either *gradient* or *binary* based methods. Gradient-based methods build their descriptor vectors based on the gradient magnitude and direction of pixels as described in Subsection 2.3.2. The binary-based methods construct their descriptors as binary strings formed employing the gradient differences. Scale-invariant feature transformation (SIFT) algorithm and speed up robust feature [26] (SURF) are the most popular gradient-based methods. Binary robust independent elementary features (BRIEF) [39], oriented fast and rotated BRIEF (ORB) [157] and binary robust invariant scalable keypoints (BRISK) are samples of binary-based keypoint detectors and descriptors. In both BRIEF and ORB, FAST corners are detected as basic for their keypoints [153]. However, BRIEF lacks the rotation invariant, and ORB needs a training step for each image dataset separately. However, the SIFT, SURF, and BRISK descriptors perform better than the others in the case of image near-duplicate retrieval and under different kinds of image affine transformations and changes in illumination, blurring, and viewpoint. Therefore, in this thesis, we employ SIFT, SURF and BRISK to solve most tasks related to image retrieval.

In the following subsections, we detail the applied image feature detectors and descriptors in this thesis. These features are color, gradient and keypoint features. Color and gradient features can be employed as global or local features, depending on the extraction methodology (either of the whole or specific areas of an image). Keypoint features (SIFT, SURF and BRISK) have been introduced to detect the

distinct regions in images i.e. as local features.

2.3.1 Color Features

Color features have been used widely in image retrieval, object recognition, and image classification due to their fast and simple computation methods [42]. Color Properties can be extracted as global (when they present all pixels in an image) or local properties (when they describe specific objects or regions of images). Different kinds of color spaces may be used depending on the task. In [118] the performance of different kinds of color histogram models (i.e. RGB, HSV, L*a*b*,..., etc.) have been compared to find out which of them simulates the human visual system [85], [84]. However, it has been shown that the model which uses Hue, saturation, and value (HSV) color space is almost capable to predict similar results to the human judgment. To improve the performance of retrieval tasks, it has been suggested to combine the color and texture features [161].

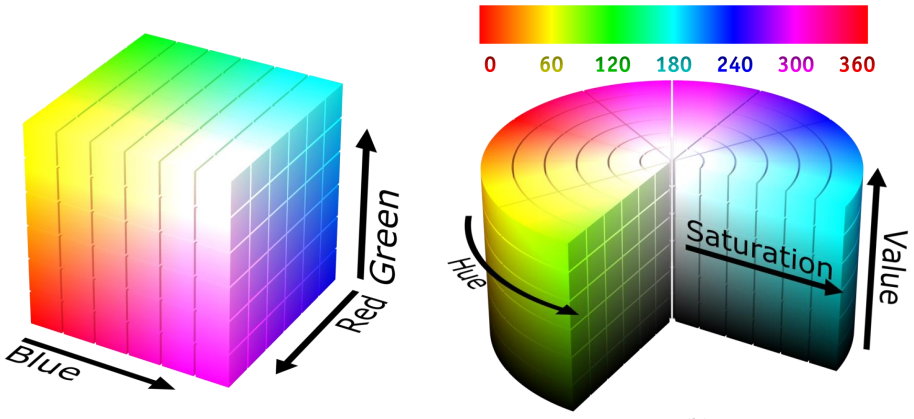
Hue Saturation Value color Space Hue saturation and value color space (HSV) is produced based on the red, green, and blue value of pixels (i.e. based on the RGB color space). The values red, green and blue channels belong to the range $[0, 255]$ (see Figure 5(a)). The values HSV channels have been defined as follows [162]:

$$H = \begin{cases} 0 & \text{if } max(R,G,B) = min(R,G,B) \\ 60 \times \frac{G-B}{max(R,G,B)-min(R,G,B)} & \text{if } max(R,G,B) = RandG \geq B \\ 60 \times \frac{G-B}{max(R,G,B)-min(R,G,B)} + 360 & \text{if } max(R,G,B) = RandG < B \\ 60 \times \frac{B-R}{max(R,G,B)-min(R,G,B)} + 120 & \text{if } max(R,G,B) = G \\ 60 \times \frac{R-G}{max(R,G,B)-min(R,G,B)} + 240 & \text{if } max(R,G,B) = B \end{cases} \quad (4)$$

$$S = \begin{cases} 0 & \text{if } max(R,G,B) = 0 \\ max(R,G,B) - min(R,G,B) & \text{otherwise} \end{cases} \quad (5)$$

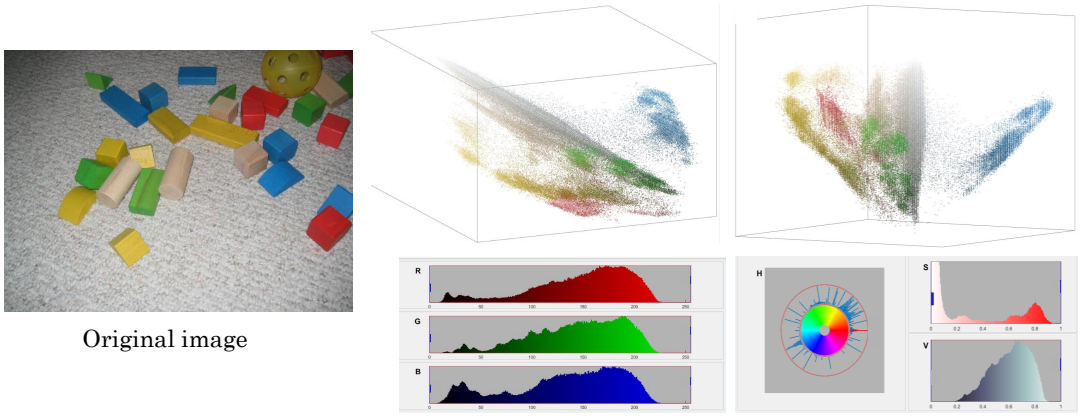
$$V = max(R,G,B) \quad (6)$$

From the previous Equations (4) , (5) and (6), we notice that *hue* presents the color itself i.e. red, yellow, blue, green, etc. and its value belongs to the range $[0^o, 360^o]$. *Saturation* measures the pureness of color, and *value* describes the amount of light in color. The values of saturation and value belong to the range $[0, 255]$. Figure 5(b) displays the HSV color space and the range of hue channel. Figure 6 presents an example of building the RGB and HSV color spaces and histograms of an image from UKBench benchmark [135]. Both color spaces in Figure 6 present the distributions and values of color of all pixels in the given image.



(a) (b)

Figure 5: (a) The RGB color space. (b) The HSV color space Hue presents color values "0" presents the red color, "60" yellow color etc. [4].



Original image RGB color space HSV color space

Figure 6: The presentation of an image employing the RGB and HSV color spaces. The histograms of both color spaces are constructed too. RGB channels belong to rang $[0, 255]$. H channel presents in range $[0^o, 360^o]$ and S and V channels are normalized and presented in the range $[0, 1]$.

2.3.2 Gradient Features

Image gradient has many applications such as edge and corner detection, blob detection, and object recognition. The gradient presents the change in intensity value and direction. Therefore, it is computed based on the concept of derivative [151], [146]. Since digital images are discrete functions and the derivative is only defined for continuous functions, images are convolved with a kernel to approximate the gradient. Most common kernels are Gaussian [34], Sobel [163] and Prewitt [148] operators. Before explaining the way of gradient computation, we clarify the concept of convolution.

The Concept of Convolution Convolution in computer vision is the process of multiplying the high dimensions image array I with a kernel array K of the same dimensionality as the input image but of various sizes to obtain a third array A that has the same dimensionality [57], [54].

$$\begin{aligned} &A = I * K \ \ where \\ &A(x,y) = \sum_i \sum_j I(i,j).K(x-i, y-j) \end{aligned} \tag{7}$$

Figure 7 presents an example of convolving 3×3 kernel with 3×3 gray-level image (i.e. one dimension). It shows how the new values of pixels are simply computed. Considering that the origin $(0,0)$ located in the center of kernel $A(1,1)$ is computed as follow:

$$\begin{aligned} A(1,1) =& 1 \times (-1) + 2 \times 0 + 0 \times (-1) \\ & 0 \times 0 + 0 \times 0 + 3 \times 0 \\ & 1 \times 1 + 0 \times 0 + 0 \times 1 \\ A(1,1) =& 0 \end{aligned}$$

Gradient Computation After Convolving a given image with a kernel as presented in Equation (7), the gradient is computed as the partial derivative concerning x and y coordination [151] i.e.:

$$G_I(x,y) = \begin{pmatrix} g_x \\ g_y \end{pmatrix} \tag{8}$$

where:

$$g_x = \frac{\partial A}{\partial x} \quad and \quad g_y = \frac{\partial A}{\partial y} \tag{9}$$

The magnitude and the direction of gradient are calculated as:

$$|G_I(x,y)| = \sqrt{g_x^2 + g_y^2} \quad and \quad \theta = atan2\frac{g_y}{g_x} \tag{10}$$

To simplify the concept of image gradient, we present in Figure 8 an example of convolving the pepper image (of Matlab images) with Sobel kernel of size 3×3. Sobel kernel is given as:

$$K_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad and \qquad K_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{11}$$

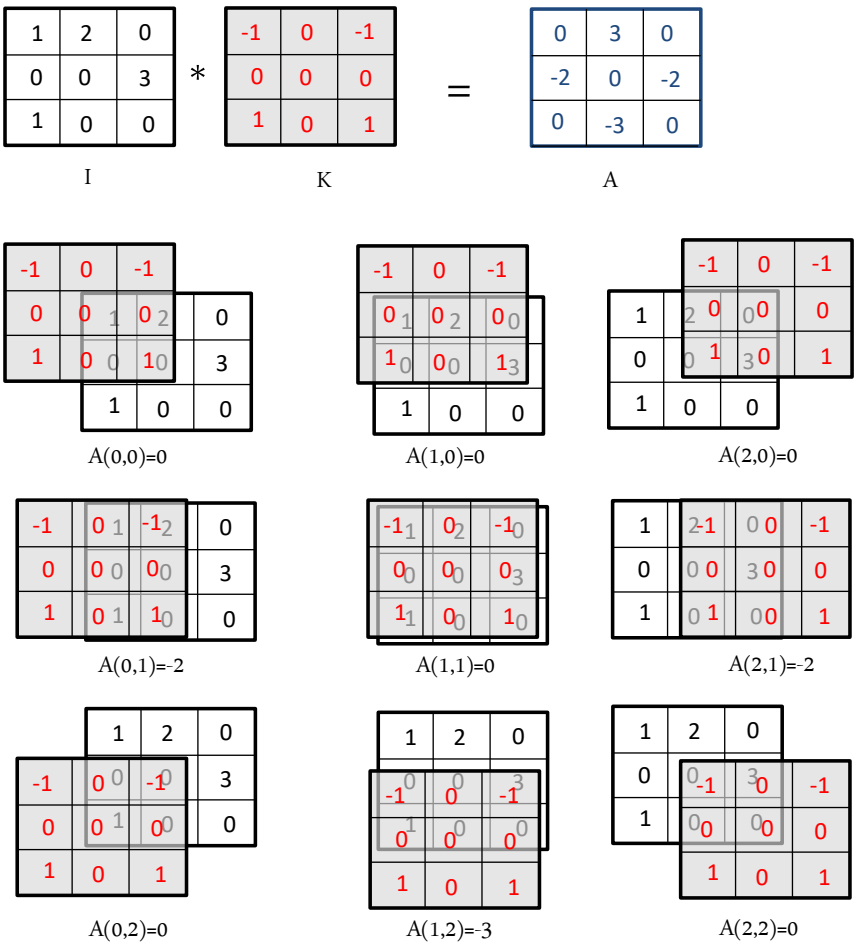


Figure 7: The convolution of a kernel K with an image I to obtain a new image A.

where K_x and K_y are the Sobel operators in x and y directions respectively. Figure 8(a) shows the original colored image. To convolve the image with Sobel kernel, we converted it into gray-scale level as clarified in Figures 8(b). Figure 8(c) and (d) present the convolving results with K_x and K_y separately. Figure 8(e) and (f) show the gradient magnitudes and the directions of pixels in the pepper image computed as given in Equations (10). Figure 8 presents that the gradient of image is computed after converting the image into gray-scale level i.e. the color channels are discard. The gradient features are used in the steps of blob detection in Section 2.3.3 and keypoint features in Sections 2.3.4, 2.3.5 and 2.3.6.

2.3.3 Blob Detection

Blobs detectors find specific regions in an image where all of its pixels have a significant property than their surroundings. The extracted blobs from an image are different in form, number, and properties depending on the used blob detector

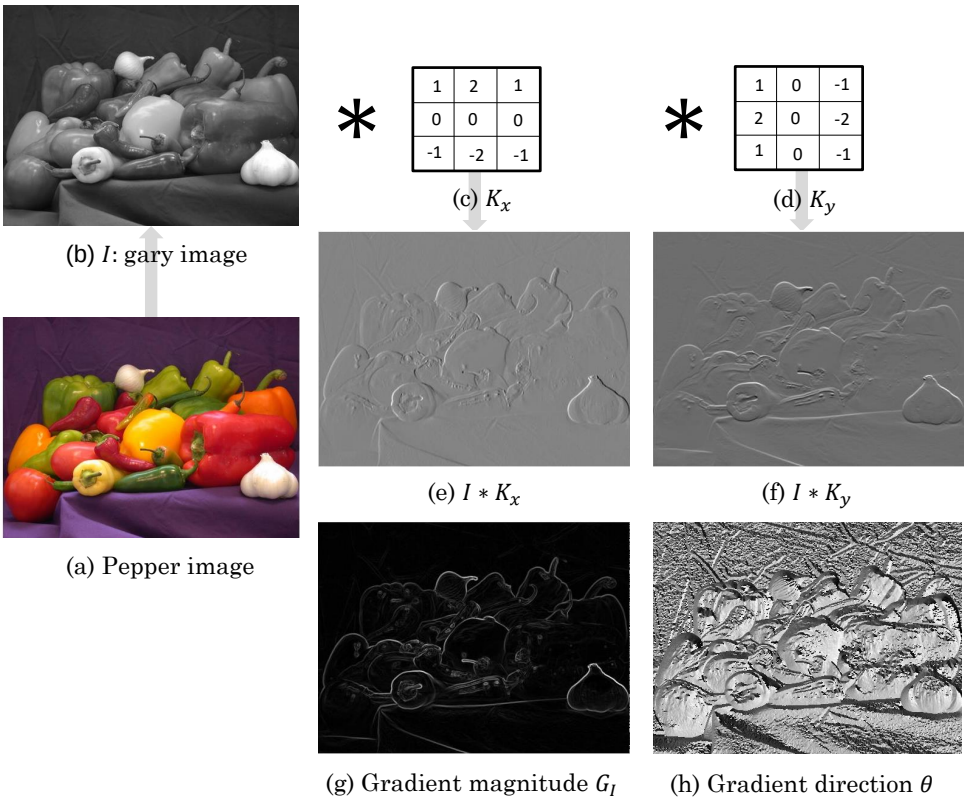


Figure 8: The convolution of a pepper image (a) with the Sobel kernels K_x (b) and K_y (c). (d) and (e) presents the result of convolution. (f) displays the gradient magnitude, and (g) shows gradient direction.

(Laplacian of Gaussian, the difference of Gaussian, determinant of Hessian [128], salient region [94] and maximally stable extremal regions [123]). These blob detectors are compared on different kinds of image affine transformation, changes in scale, viewpoint, and blurring. However, it has been shown [130] that maximally stable extremal regions detector obtains the highest performance.

Maximally Stable Extremal Regions MSER MSER is a method to detect blobs in images [123]. The detected blobs are invariant to affine transformation described in Section 2.2. MSER detects the blobs in a gray-scale image. As shown in Figure 9 left, MSERs look like a component tree. Each level in this tree indicates the arising of new blobs. The nodes in the same branch are produced by expanding the size of regions. To build the component tree, a sequence of thresholds is applied to generate thresholded images. The blobs of thresholded images expand and merge depending on the value of used thresholds. The extremal regions are the set of all components in the tree [123]. The steps of building the MSER blobs are summarized as follows:

1. Given an input image, it is converted first to the gray-scale space. Thus the

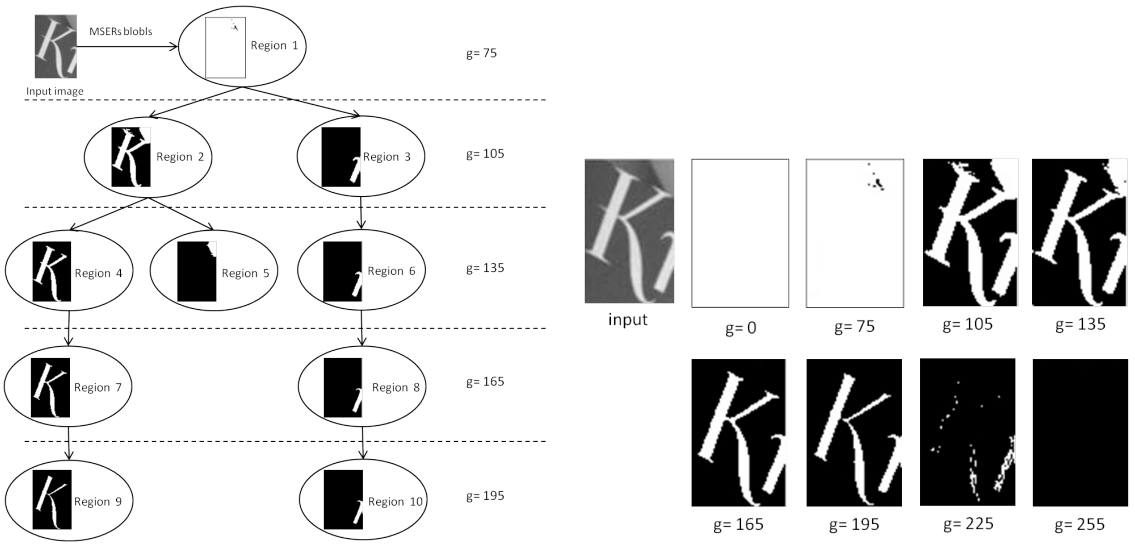


Figure 9: Blobs detected by MSERs. The component tree of nested regions is shown on the left side. The subsequent frames generated at different thresholds are presented on the right side.

pixels have gray values in a range $\{0, 255\}$.

2. Determine a range of thresholds. For each threshold, the black value is assigned to the pixels that have intensities above this threshold and the white to the others.

3. By using a sequence of thresholds, thresholded images are subsequently generated with frames corresponding to the thresholds. As clarified in Figure 9 right, the first image in the sequence is white. In the next sequences, black blobs appear.

4. By increase the thresholds, the blobs expand to obtain at the end a black image.

5. The black blobs in the sequence of images express the extremal regions. The word *extremal* indicates that all pixels inside one region have either higher or lower intensity than the surrounding pixels [123].

6. Extremal region Q_{i^*} is maximally stable iff $q(i) = |Q_{i+\Delta} - Q_{i-\Delta}| \,/\, |Q_i|$ has a local minimum at i^*, where $Q_1, ..., Q_{i-1}, Q_i, ...$ are sequence of nested extremal regions i.e., $Q_i \subset Q_{i+1}$ and $\Delta \in \{0, ..., 255\}$ [123].

2.3.4 Scale Invariant Feature Transformation Algorithm (SIFT)

The SIFT algorithm is very popular due to the robustness of its features against some kinds of deformation such as scale change, illumination change, JPEG compression,

change in 3D viewpoint, and a particular amount of noise. In the following, we detail the main steps of the SIFT algorithm [116]:

Construct image scale-space (pyramid) The first step to build image scale-space is generating a set of octaves. The first octave is the input image itself. The second octave is produced by half-sampling the input images. Subsequently, each octave is created by half-sampling the previous one. In this way, the primary layers of the scale-space are constructed in a pyramid form. The number of octaves is limited by the size of the input image and calculated as $O = log_2(\min(W, H)) - 2$, where W and H denote the width and height of the input image respectively. Due to the massive loss of image details across octaves, the top two octaves are discarded. To complete the construction of the scale-space, the primary layer of each octave is convolved with a multiple-scale Gaussian filter as follows:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{12}$$

where $I(x, y)$ denotes the input image or one of the down-sampled images, $*$ is the convolution operation in the position (x, y) and $G(x, y, \sigma)$ is Gaussian filter (employed to build Gaussian kernel) defined as:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{13}$$

Each generated image of the convolution forms one "*scale*" in the scale space, and is presented by the value of σ (Figure 10(a)). The standard number of scales in each octave is determined in the implementations to be $S = 6$. Figure 10 summarizes the method of building Gaussian pyramid (in Figure 10(a)) and difference of Gaussian pyramid Figure 10(b). To clarify the concept of image scale-space, we constructed the Gaussian pyramid for a beaver image in Figure 11(a) and the difference of Gaussian pyramid in Figure 11(b). These pyramids present the distinct areas of images that are employed to produce the keypoints.

Image down-sampling and smoothing with Gaussian filter cause loss of peaks, where the highest spatial frequencies occur. Therefore, Low [116] suggested expanding the input image by a factor of 2 using bi-linear interpolation [147] before constructing image scale-space. This step guarantees to process whole features across the octaves of scale space. Experiments show that using an input image without expanding its size decreases the number of robust keypoints by a factor of 4.

Difference of Gaussian DoG pyramid Construction Low [116] construct the DoG pyramid by subtracting the adjacent scaled images in each octave.

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{14}$$

in [127] it has been proven that the Laplacian of Gaussian $\sigma\nabla^2 G$ detects the most robust areas in images. However, its computation is expensive comparing with

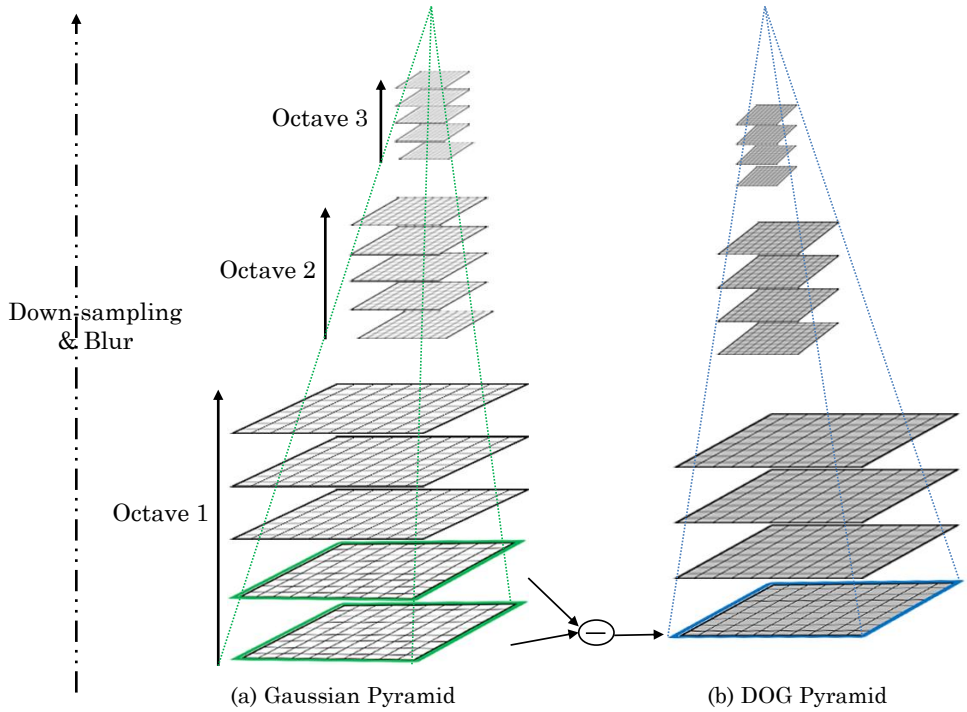


Figure 10: The construction of the Scale-space pyramid. (a) The Gaussian pyramid contains three octaves, each has five layers. (b) The difference of Gaussian pyramid includes three octaves, each has four layers.

the DoG. However, in [109], it has been reported that the difference of Gaussian approximates the scale-normalized Laplacian of Gaussian as [116]:

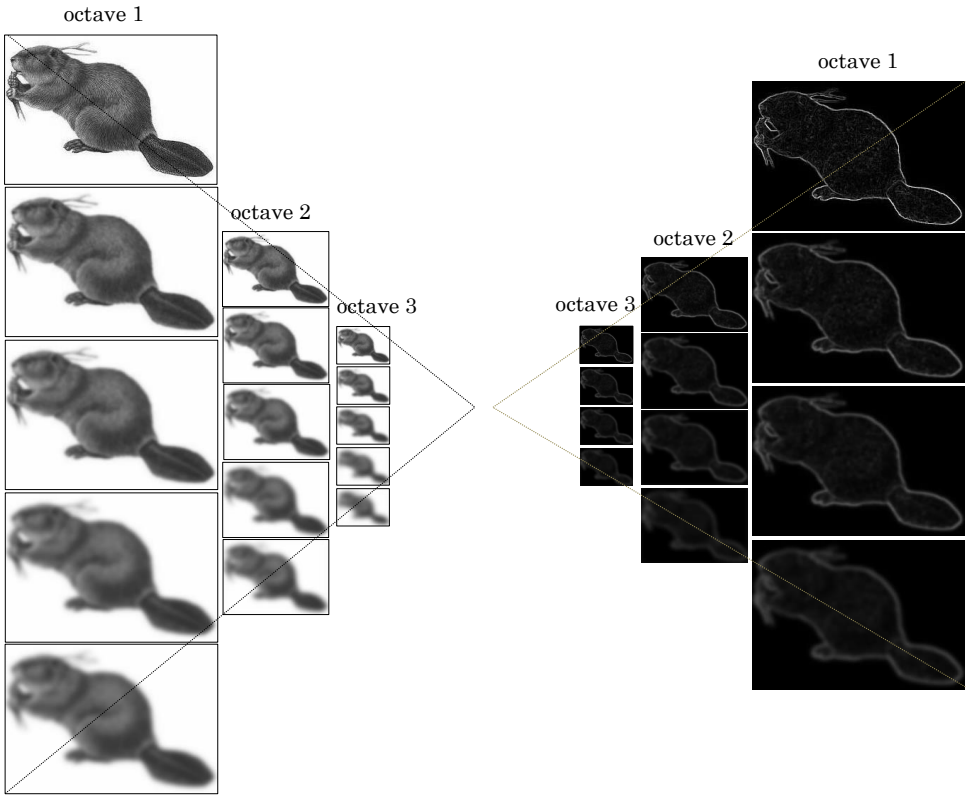
$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma} \tag{15}$$

The rearranging of Equation (15) gives:

$$(k-1)\sigma^2\nabla^2 G \approx G(x, y, k\sigma) - G(x, y, \sigma) \tag{16}$$

The multiplicative factor $k-1$ is constant overall scales therefore, it has no impact on the extrema detection and localization. Hence Low employed the proposed approximation and constructed the DoG instead of computing the Laplacian of Gaussian. Figures 10 and 11 present the construction of difference of Gaussian pyramid. They clarify that the octaves of DoG contain lesser layers than those in the Gaussian pyramid. Accordingly, if S is the number of scales in an octave, the corresponding one in the DoG pyramid contains $S-1$ layers.

Keypoints Identification The candidate SIFT keypoints are justified through three steps:



(a) Gaussian Pyramid (b) DOG Pyramid

Figure 11: Example of Scale-space pyramid construction employing the image of beaver. (a) The Gaussian pyramid of the beaver image. (b) The difference of Gaussian pyramid of the beaver image.

- Extract extreme points: The local maxima and minima are specified in the DoG pyramid by comparing each pixel with its eight neighbors of the current scale and its nine neighbors in the layers above and below. As shown in Figure 12(A), the pixel marked with $\times$ is accepted as a keypoint candidate if its intensity value is bigger or smaller than all shown neighbors.

- Keypoints localization: To determine the accurate positions of keypoints Low [116] proposed using a model in terms of second-order Taylor expansion of a scale-space function. This function is given as:

$$D(z) = D + \frac{\partial D^T}{\partial z} z + \frac{1}{2} z^2 \frac{\partial^2 D}{\partial z^2} z \tag{17}$$

where $z = (x, y, \sigma)$ is the offset of a sample keypoint. To identify the extreme location of a keypoint, the derivative of this function concerning z is set to zero. So the position is given as:

$$\hat{x} = \frac{\partial^2 D^{-1}}{\partial z^2} \frac{\partial D}{\partial z} \tag{18}$$

- Reject unstable keypoints (flats and edges): Once locations of keypoints are determined, their stability is verified against contrast change and edge response. The contrast of each keypoint is calculated using the relation:

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D^T}{\partial x}\hat{x} \tag{19}$$

which comes from the substituting of Equation (18) in Equation (17). Keypoints, with a value of contrast lesser than a specified threshold, are rejected. The contrast threshold belongs to the range $[0.04, 0.20]$. The values closer to 0.04 produce more keypoints than the ones in the neighboring of 0.20. Due to the strong response of DoG function along edges, the edge response of keypoints is checked to get more robust keypoints.

Orientation Assignment The orientation of a keypoint is calculated in the closest scaled image where a keypoint is detected. To give keypoints more robustness against rotation change, a 36 bins gradient-orientation histogram is constructed, which presents the orientation in the range $[0°, 360°]$. The neighbor pixels around keypoints contribute their gradients and orientations to construct this histogram. Intensity difference is employed to compute the gradient magnitude $m(x, y)$ and the orientation $\theta(x, y)$, as follows:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{20}$$

$$\theta(x,y) = arctan\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)} \tag{21}$$

The orientation determines the histogram bin where a sample will affect. The

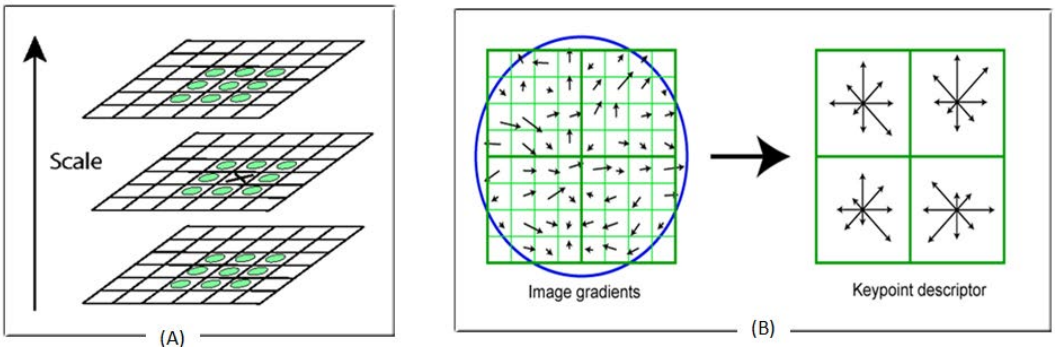


Figure 12: (A) maxima and minima in the difference-of-Gaussian pyramid. (B) a keypoint descriptor is created by first compute the gradient magnitude and orientation at each image sample point in a region around the keypoint location. This figure shows a 2×2 descriptor array computed from 8×8 set of samples, whereas the experiments in [116] use 4×4 descriptors computed form a 16×16 sample array [116].

magnitude increases the assigned value to that bin. Before adding magnitude values

to the orientation bins, they are weighted using a Gaussian-circular window of a size 1.5 times the keypoint scale. The highest bin in the histogram determines the orientation of keypoint. To improve the stability of keypoints, the bins of the histogram within 80% of the most powerful one, are considered to create new keypoints with those orientations. Hence, keypoints of various orientations and descriptor vectors may share the location.

Keypoint Descriptors After extracting keypoints and assigning their scales and orientations, the properties of patches around keypoints are computed to form the descriptors. A descriptor is constructed by first rotate the region around a keypoint employing the computed orientation. The radius of a sampled region depends on the layer (i.e. the keypoint scale) in which a keypoint is detected. To avoid the sudden change in a descriptor caused by intensity change when a small change in location happens, the patches around keypoints are smoothed employing a Gaussian weighting function with σ equal to one half the width of the descriptor window. This also helps to give less importance to the samples that their gradients strongly vary of the descriptor center. After that, a 4×4 orientation histogram is created over the sample patch to allow the gradient-shifting in four directions. For each direction, 8 orientations are assigned, so that the descriptor contains 3 dimension and $4 \times 4 \times 8 = 128$ elements (Figure 12(B)). To smooth the descriptor, a tri-linear interpolation is applied by multiplying each added sample with a weight of $(1 - d)$, where d is the distance of a sample from the center of the keypoint. The weights are presented in Figure 12(B) by the overlaid circle. Finally, the descriptors are normalized to a unit length to decrease the effects of illumination change.

In case of linear brightness change or contrast change, the normalization helps to eliminate their effects. But if non-linear illumination change occurs, the gradient magnitude is mainly affected. However, the gradient orientation is less likely changed therefore, to reduce the impact of magnitude change, the gradient magnitudes are checked to be no larger than 0.2 next the vector is re-normalized to a unit length.

2.3.5 Speed Up Robust Feature Detector and Descriptor (SURF)

The SIFT algorithm is to extract features of images, that are invariant to scale and rotation changes and robust against various image transformations like blurring, adding noise, and viewpoint change. However, to determine the extreme regions, a comparison process is frequently iterated for all pixels of all layers in the DoG pyramid, which is time-consuming. To reduce the time of keypoints extraction, the integral image and filter box concepts have been introduced in the SURF algorithm [27]. The integral image and filter box replace the Gaussian pyramid and the difference of Gaussian pyramid of the SIFT algorithm. The following steps describe the details of the SURF algorithm.

Integral Image and Box Filter Construction The concept of the integral image or so-called summed-area table has been firstly addressed in 1984 to improve texture filtering [50]. The idea of the integral image has been introduced to accelerate the extraction of image features. The integral image at a specific location $\mathbf{x} = (x, y)$ of an image I is defined as the sum of intensities overall upright pixels i.e.:

$$s(x, y) = \sum_{\substack{x_i \leq x \\ y_i \leq y}} I(x_i, y_i) \tag{22}$$

where $I(x_i, y_i)$ is the intensity at location (x_i, y_i).
Figure 13 clarifies that the computation of $s(x, y)$ requires only four values $s(x-1, y)$, $s(x, y-1)$, $s(x-1, y-1)$ and $I(x, y)$ i.e.:

$$s(x, y) = s(x, y-1) + s(x-1, y) - s(x-1, y-1) + I(x, y) \tag{23}$$

Therefore, $s(x, y)$ is independent of the size of the integral image. As shown in

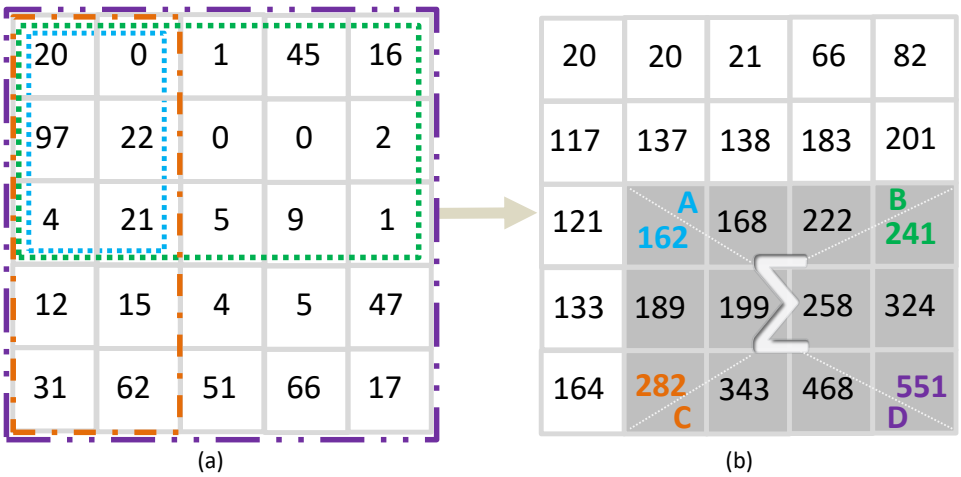


Figure 13: (a) The construction of integral images, the dashed frames present the way of building integral images in four locations A, B, C and D. (b) The box filter Σ is computed as $\Sigma = 551 - 241 - 282 + 162$.

Figure 13, based on integral images, the intensity is computed inside any box filter Σ of any size determined by points A, B, C, D as follows:

$$\Sigma = s(x_D, y_D) - s(x_B, y_B) - s(x_C, y_C) + s(x_A, y_A) \tag{24}$$

The box filters allow the parallel processing of image patches. Therefore, they are used to accelerate the computation of keypoints.

Approximation of the Hessian Matrix with Box Filter To extract the interest points it has been proposed by the SURF algorithm to apply the Hessian

matrix at each pixel $\mathbf{x}$ of an image I. Hessian matrix is the convolution of the Gaussian second-order derivative at the point $\mathbf{x}$ and given as [27]:

$$\begin{bmatrix} \frac{\partial^2 G(\mathbf{x})}{\partial^2 x} & \frac{\partial^2 G(\mathbf{x})}{\partial x \partial y} \\ \frac{\partial^2 G(\mathbf{x})}{\partial x \partial y} & \frac{\partial^2 G(\mathbf{x})}{\partial^2 y} \end{bmatrix} \tag{25}$$

Similar to [173], the SURF algorithm proposed to approximate the Gaussian second-order derivative with the filter box. This is since the filter box has similar effect to the Gaussian second-order derivative (as seen in Figure 14). This approximation accelerates the building of scale-space and extraction of the interest points.

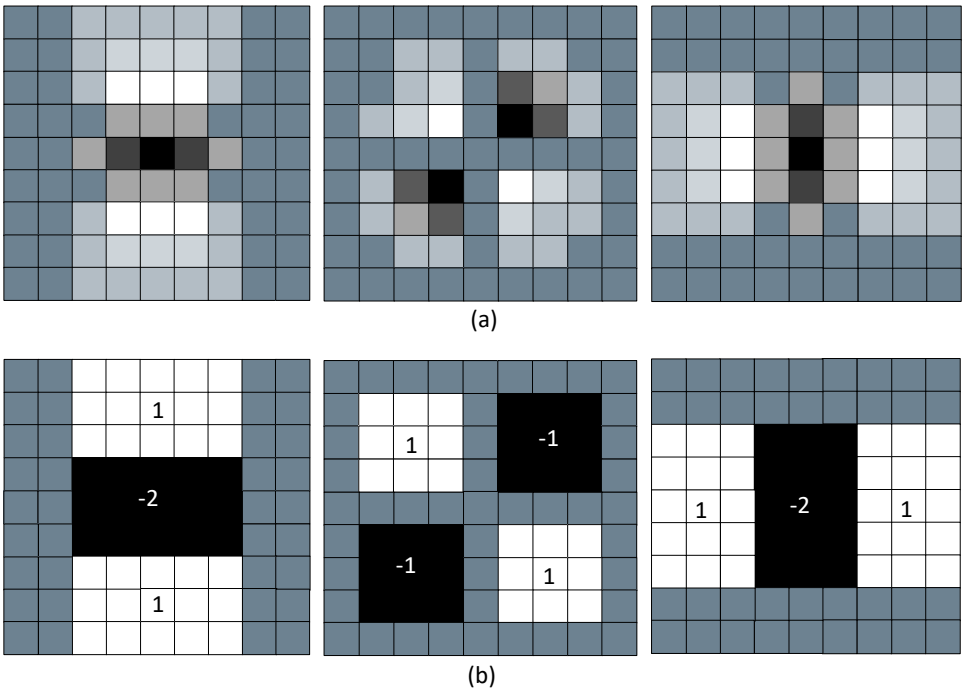


Figure 14: Comparison between (a) the Gaussian second order partial derivatives in y, xy and x directions respectively and (b) the box filter.

Scale-Space Pyramid To build the scale-space, box filters of various sizes are applied to the original image in parallel. Hence, contrary to the SIFT algorithm, the scale space is constructed by increasing the size of the box filter instead of down-sampling the input image. The scale-space consists of octaves, each, in roll, includes a set of layers. The layers differ with the size of box filters but it is constant in each octave. Given $O-1$, $O-2$ and $O-3$ three consecutive octaves, then the difference in the box filter size between $O-2$ and $O-3$ is the double of the difference between $O-1$ and $O-2$. The initial box filter has a size of 9×9, which represents an approximation to the Gaussian second-order derivative with $\sigma = 1.2$. The sizes

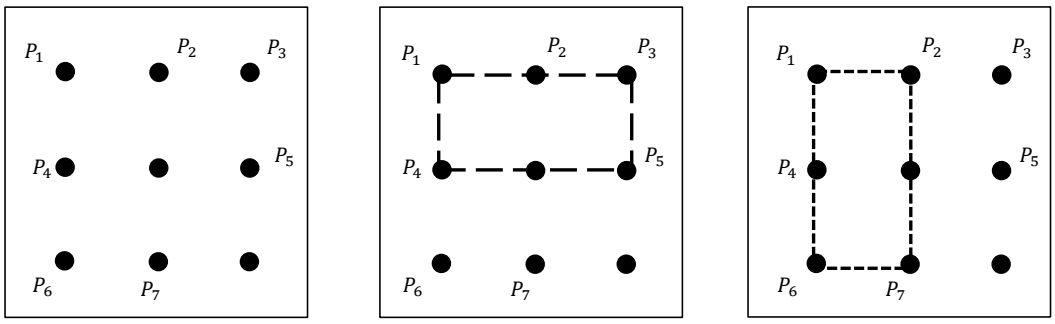
of the next box filters increase by six pixels to get box filters: 15×15, 21×21, and 27×27 in the first octave. For each new octave, the box filter is doubled i.e. 6, 12, 24, and 48 to first, second, third, and fourth octave, respectively. Hence, the filters of the second octave are going from 15×15 to 27×27 to 39×39 to 51×51. The sizes of filters of the third octave are 27×27, 51×51, 75×75, and 99×99. If the size of the processed image still larger than the size of the box filter, the fourth octave is constructed using the sizes 51×51, 99×99, and 147×147. The scale difference between the layer in scale space is quite big i.e. for the first and second layers in the first octave is $\frac{15}{9} = 1,67$. To overcome this problem, in implementation, a scale-space refinement has been recommended by up-scaling the input image to the double size before building the scale space [27], [26]. The up-scaling is completed by applying linear interpolation to the original image. Since the double sizing, the first applied box filter has the size 15×15 instead of 9×9 in the theoretical suggestion. The increase of the filter size between the first and the second octaves is 12 instead of six i.e. the first filter box in the second octave is 27×27 and so on [27], [26].

Keypoints Detection and Localization To determine keypoint candidates, a non-maximum suppression technique is applied in the eight neighboring of the same layer and the 3×3 neighboring in the up and bottom layers of the same octave. After that, keypoints are defined maxima of the determinant of the Hessian matrix. To localize keypoints, in the scale-space, the interpolation technique presented in [173] is utilized. This interpolation is necessary to avoid the impact of the big difference in the size of box filters between the layers of the same octave.

Orientation Assignment The Haar-wavelet response is computed in both horizontal and vertical directions. All pixels within a circle of radius $6s$ join the orientation computation, where s is the scale of the keypoint. To accelerate this process a box filter is again employed to replace the Haar-wavelet at the same scale. Hence, as shown in Figure 15 simply six computations are required to compute the response in vertical (Figure 15(b)) or the horizontal (Figure 15(c)) direction. Once the responses are computed, the orientations within a sector of size $\frac{\pi}{3}$ are summed together. The maximum estimated orientation over all sectors is selected to present the keypoint orientation.

Build Descriptor The descriptor is calculated by applying the Haar-wavelet filter of size $20s$ around interest points. This window is divided into $4x4$ subregions for each, the Haar-wavelet responses are computed. Hence a vector of $v = \Sigma dx, \Sigma dy, \Sigma|dx|, \Sigma|dy|$ is constructed for each of the $4x4$ blocks i.e. the SURF descriptor has 64 elements.

Figure 16 presents the extraction of the SURF keypoints. Comparing Figure 16(a) and (b), we find out that the most robust features are presented in the top 30 features and they present bigger image regions than the other keypoints.



(a) filer box at P_1, ..., P_7 (b) vertical response (c) horizontal response

Figure 15: The approximation of the Haar-wavelet response with box filters. (a) present a part of an image where box filters are computed. (b) approximation of the horizontal wavelet response with a box filer $P_1P_3P_4P_5$. (c) approximation of the horizontal wavelet response with a box filter $P_1P_2P_6P_7$.

(a) Strongest 30 SURF Keypoints (b) Strongest 50 SURF Keypoints

Figure 16: SURF keypoints of a rotated boat image taken form [5]. (a) the top 30 SURF keypoints. (b) the strongest 50 SURF keypoints.

Modifications of the SURF As suggested in [27], [26] in some studies (such as object detection), the rotation invariant of the interest points is not necessary to be justified. Therefore, the upright-SURF (U-SURF) has been proposed, where keypoints are invariant only for scale change and a small rotation change (i.e. lesser than ± 15). In this case, the computation of the dominant orientation is kept out. Hence, U-SURF reduces the computation time.

To preserve the rotation invariant property and at the same time accelerate feature extraction and matching processes, the SURF-36 has been introduced. The regions around interest points are divided into 3×3 sub-regions instead of 4×4 in the original SURF algorithm. However, the employing of a shorter descriptor decreases the performance of matching.

To increase the distinction of interest points, SURF-128 has been presented. To get longer descriptor, for each of the 4×4 sub-regions the responses d_x, $|d_x|$ and

d_y, $|d_y|$ are split up regarding the sign of d_y and d_x respectively. The extension of SURF descriptor to 128 element improve the matching but it increases features computation and matching costs.

2.3.6 Binary Robust Invariant Scalable Keypoints (BRISK)

Like SIFT and SURF, BRISK detects its keypoints by building image scale-space [105]. However, instead of gradient-based descriptors, BRISK constructs binary descriptors which, are faster in building and matching than the gradient descriptors [105]. In the following, we describe the main steps to extract BRISK keypoints.

Scale-Space of BRISK Like the SIFT scale-space, the first step to build the BRISK scale-space is to down-sample the input image. The scale-space has a pyramid form and contains a set of n octaves $c_0, c_1, ..., c_{n-1}$. In addition to octaves the scale-space of BRISK has intra-octaves $d_0, d_2, ..., d_{n-1}$. The octaves are produced by down-sampling of the original image using the scale factor $s_i = 2^i$ where $i = 0, 1, ..., n-1$. The first intra-octaves d_0 is built by down-sample the original image by a factor 1.5. The remains intra-octaves are built subsequently by half-sampling the d_0 or by down-sampling the input image utilizing the scale factor $s_{di} = 2^i.1.5$, where $0, 1, ...n-1$. Figure 17(a) clarifies that the BRISK scale-space contains only one layer in each octave but it has the property that it has intra-octaves.

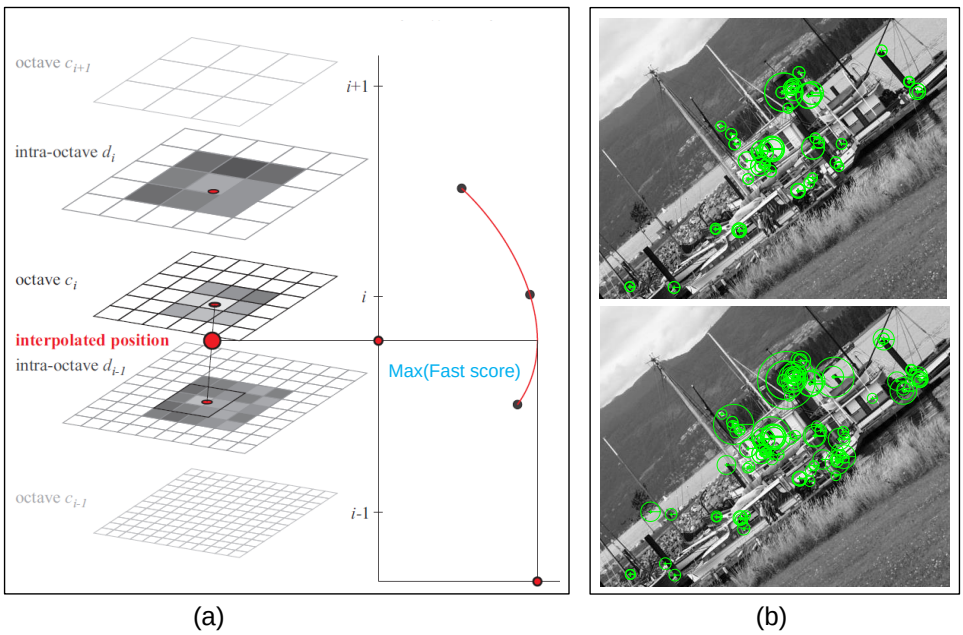


Figure 17: (a) The octaves and intra-octaves of BRISK scale-space [105]. (b) The extraction of BRISK keypoints using two images of a boat but with different rotations [5]. The circle areas present the scale where the keypoints are detected. The radius presents the orientation of a keypoint.

Keypoints Extraction BRISK employs the *FAST* corner detector on each layer in the scale space [154]. *FAST* is the shortcut of *Features from Accelerated Segment Test*. The FAST algorithm accelerates corner detection by comparing each candidate corner p to its 16 on circle surrounding neighbor pixels as shown in Figure 18. If the intensity of p is lesser or greater than the sixteen circle pixels then p is a corner. To accelerate this process, labels with the numbers one to sixteen are assigned to the circle pixels in the clockwise direction. The intensity of p is checked firstly including pixels 1, 5, 9, and 13. If the condition of intensity is satisfied for at least three of them, the comparison with the rest is accomplished, otherwise, the comparison is stopped since p is not anymore a candidate to be a corner. In this way, the keypoint candidates are defined in all octaves and intra-octaves separately. To determine final keypoints through the scale-space, each keypoint candidate is compared with its eight neighbors in the same layer and nine neighbors in the layer above and below. If the score of this candidate p is greater than all neighbors, then the scale of this keypoint is estimated. For this, the maxima in the 3×3 region around p is computed by convolving it with a quadratic function. This process is repeated for neighboring patches above and below. After that, as shown in Figure 17(a), the Maxima through the three layers are fit to $1D$ parabola and the scale s_p of keypoint p is selected as the maximum of this parabola.

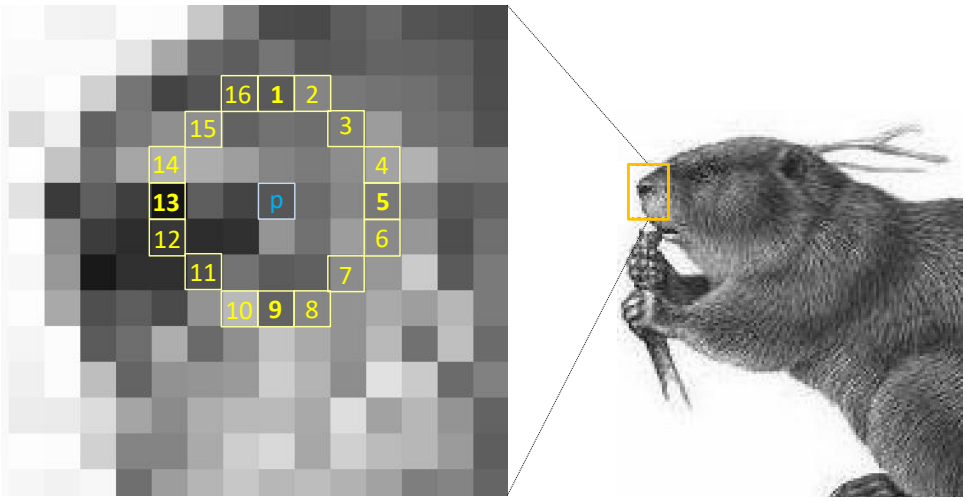


Figure 18: The main idea of the FAST corner detector. The pixel p is compared with the labeled group of pixels. To accelerate the verification, p checked firstly with the pixels with labels 1, 5, 9, and 13.

Rotation Computation To compute the rotation of a keypoint p the Gaussian filters with pre-defined distances are applied on the surrounding region of p. Let p_i and p_j are samples of these N pixels region, with intensity values after convolving with Gaussian filters, $I(p_i, \sigma_i)$, $I(p_j, \sigma_j)$, respectively. The local gradient of this pair

is:

$$g(p_i, p_j) = (p_j - p_i)\frac{I(p_j, \sigma_j) - I(p_i, \sigma_i)}{\|p_j - p_i\|^2} \tag{26}$$

To define the gradient and orientation of p two concepts are introduced: short-distance set S and long-distance set L of all possible pairs A where:

$$A = \left\{(p_i, p_j) \in R^2 \times R^2 | i < N \wedge j < i \wedge i, j \in N\right\}$$

$$S = \{(p_i, p_j) \in A | \, \|p_j - p_i\| < \delta_{max}\} \subseteq A \tag{27}$$

$$L = \{(p_i, p_j) \in A | \, \|p_j - p_i\| > \delta_{min}\} \subseteq A$$

δ_{max} and δ_{min} are the distance thresholds and their values are $\delta_{max} = 9.75 s_p$ and $\delta_{min} = 13.67 s_p$ [105]. Based on the previous details, the gradient of the keypoint p is calculated employing the long-distances [105]:

$$g_p = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \frac{1}{L} \sum_{(p_i, p_j) \in L} g(p_i, p_j) \tag{28}$$

Based on the values of g_x and g_y, the orientation θ is computed as described in Subsection 2.3.2.

Keypoint Descriptor The computed local gradients and orientation θ of keypoint p are employed to build its descriptor. The region around p is rotated by θ after that, only the short-distance pairs that the satisfy Equation (27) are compared to build the descriptor of p. For a pair $(p_i^\theta, p_j^\theta) \in S$ the descriptor element b is calculated as:

$$b = \begin{cases} 1 & \text{if } I(p_j^\theta, \sigma_j) > I(p_i^\theta, \sigma_i) \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

In this way, the binary descriptor of BRISK is built as a string of 512 bit (i.e. 64 byte).

Figure 17(b) presents the extracted BRISK keypoints employing of a boat image [5] (that used in [130], [128], [129]), with different rotation angles. The images above and below show the top 30 and 70 robust keypoints, respectively. By comparing both images, we find out the most identified features in the low level of the scales-space (presented with small circles) do not belong to the top 30 strong features. The robust features are detected over all scales in the pyramid therefore, they are presented with big circles.

2.4 Features Indexing and Matching

Before using the keypoint features in the matching process, most retrieval systems utilize a feature indexing step to structure and quantize the descriptors in a suitable form for further processing steps. The k-dimensional tree *k-d tree* has been used to structure the gradient descriptors in the high dimensional space and speed up their matching process [28], [175]. The main idea of the k-d tree is to split the set of descriptors based on the dimension where the maximum variation occurs. This process is repeated to build the k-d tree of specific depth. *K-means* clustering algorithm [81], [36], [117] has been applied as direct clustering method to quantize SIFT descriptors and splitting them into k groups [108], [191], [75], [76]. In this case, a specific number of clusters is defined and the descriptors are indexed depending on their closest centers. The produced cluster centers form a *bag of visual words*. The descriptors are presented utilizing this bag of features. The hierarchical clustering methods [159], [56], [112], [134] specifically the hierarchical *k-means* clustering method [19] have been employed to build descriptors vocabulary tree [90], [136]. The vocabulary tree is created by applying the $k-$means algorithm on the entire descriptor database which split them into k clusters where each cluster consists of a set of descriptors closest to a particular center. This process is applied recursively on each cluster to build a vocabulary tree of depth L and k^L leaf nodes. The tree nodes present cluster centers and form the *visual words* that represent features as a bag of visual words [90]. The leaf nodes in the tree are represented by inverted files. Each inverted file contains the indexes of the images that they have at least one descriptor in the corresponding leaf node. The inverted files of the leaf nodes are concatenated to build the inverted files of inner and root nodes. These inverted files strongly speed up the matching process.

To increase the robustness and speed up of keypoints matching process, techniques based on the bag of visual words have been proposed. Features pyramid kernel has been constructed by merging the clusters, that the Euclidean distance between their centers is lesser than specific threshold [76], [75]. This merging is applied recursively till having all features belonging to one group. This idea is extended in [104], [188] by considering the spatial information of features in the merging operation. In [189], [193], k-mean clustering with sparse encoding has been used to compute the relation between each feature and all cluster centers. These methods exploit the relationship between the neighboring features to improve image classification and near-duplicate detection.

Hashing functions have been employed to generate keys to index keypoint features [23], [46]. Locality sensitive hashing function, that indexes the similar descriptors with high possibility to the same keys [55], [74], has been used to identify the near-duplicate images [100]. Min-Hash function has been applied to detect the similarity between near-duplicate images [47]. Min-hash function is a locality-sensitive hashing function, which accelerates the hashing process by computing the intersection and union of the compared sets A and B and use the result to determine the mapped hashing keys.

The improvement in feature indexing has been discussed in the previous researches to accelerate the matching and enhance its quality. As shown in Figure 1, feature indexing step is not in our focusing therefore, we apply in Chapters 5, 6 and 7 the k-d tree and the hierarchical k-means clustering techniques but without any modifications in their basics.

2.5 Evaluation

2.5.1 Feature Matching

To measure the similarity between two images the distances between their features are computed. Two features match if the distance between them is smaller than a pre-defined threshold. For gradient keypoints and histograms features the Manhattan (or so-called L1-norm) L1 [156], [72] and Euclidean [49], [97] (or so-called L2-norm) L2 distances are recommended [152]. For histogram features, other methods have been employed such as Chi-Squared [190], Bhattacharyya [95] and histogram intersection [60] distances. For binary keypoint features, Hamming distance is often computed [174].

2.5.1.1 Gradient Keypoints Matching The gradient Keypoints like SIFT [116] and SURF [26], [27] reported in Subsections 2.3.4 and 2.3.5, describe distinct regions in images by their locations, scale, orientation and descriptor vectors. In the gradient descriptors, the distance between two keypoints is computed as the distance between their descriptors. For this the Manhattan L1 [156], [72] or Euclidean [49], [97] L2 distance is employed. Given are two keypoints p and $p^{'}$, which have the descriptors $v = (v_1, v_2, ..., v_n)$ and $v^{'} = (v_1^{'}, v_2^{'}, ..., v_n^{'})$ respectively, where n is the length of the descriptor. The Manhattan distance presents the absolute distance between the corresponding components of two vectors v and $v^{'}$ and is defined as:

$$L1(v, v^{'}) = \sum_{i=1}^{n} |v_i - v_i^{'}| \tag{30}$$

The Euclidean distance represents the shortest distance between vectors and is computed as the root of the sum of all quadratic differences between the corresponding components of two descriptors. The Euclidean distance is clarified as:

$$L2(v, v^{'}) = \sqrt{\sum_{i=1}^{n} (v_i - v_i^{'})^2} \tag{31}$$

2.5.1.2 Binary Keypoints Matching To match the keypoints which have binary descriptors as BRISK keypoints [105] described in Subsection 2.3.6, the location information are ignored too. Only the distance between the binary descriptors

is computed. Since the binary descriptors are strings of binary values i.e. strings of 0 and 1, the Hamming distance between them is computed be means of XOR operator. XOR operator of two binary strings $s = s_1 s_2 ... s_n$ and $s^{'} = s_1^{'} s_2^{'} ... s_n^{'}$ is defined as:

$$sXORs^{'} = s_1 s_2 ... s_n XOR s_1^{'} s_2^{'} ... s_n^{'} = xb_1 xb_2 ... xb_n \tag{32}$$

where:

$$xb_i = \begin{cases} 0 & \text{if } s_i = s_i^{`} \\ 1 & \text{otherwise} \end{cases} \tag{33}$$

To present the distance between two binary descriptors, Hamming distance counts the values 1 in the $xb_1 xb_2 ... xb_n$ i.e.:

$$HammingDistance(s, s^{'}) = \sum_{i=1}^{n} xb_i \tag{34}$$

2.5.1.3 Histogram Matching To compute the similarity between two histograms various measures have been proposed [121]. Considering $H = b_1, ..., b_n$ and $H^{'} = b_1^{'}, ..., b_n^{'}$ are two histograms, the similarity between them can be computed employing one of the following methods:

- **Chi-Squared Distance** [190]**:** It computes the difference of the corresponding bins divided by their sum. So that, Chi-Squared distance is zero if the histograms are identical. Its value becomes larger when the differences between the corresponding bins increase. The Chi-Squared distance is expressed as [121]:

$$Chi - SquaredDistance(H, H^{'}) = \sum_{i=1}^{n} \frac{(b_i - b_i^{'})^2}{(b_i - b_i^{'})} \tag{35}$$

- **Bhattacharyya Distance** [95]**:** It computes the distance between two histograms as the multiplication of their corresponding normalized bins. It is given as [121]:

$$BhattacharyyaDistance(H, H^{'}) = 1 - \sum_{i=1}^{n} \frac{(b_i . b_i^{'})^2}{\sum b_i . \sum b_i^{'}} \tag{36}$$

  So that, its values are in the range $[0, 1]$. The value zero describes that the histograms have identical corresponding bins.

- **Histogram Intersection** [60]**:** It measures the distance between two histograms based on the minimum value of the corresponding bins. The histogram intersection is given as:

$$HistogramIntersection(H, H^{'}) = \sum_{i=1}^{n} min(b_i, b_i^{'}) \tag{37}$$

- **Histogram Correlation** [37]**:** Instead of bins, the variation of bins is computed to measure the distance between two histograms. The correlation is calculated as follow:

$$HistogramCorrelation(H, H^{'}) = \frac{\sum_{i=1}^{n}(b_i - \overline{b_i})(b_i^{'} - \overline{b_i^{'}})}{\sqrt{\sum_{i=1}^{n}(b_i - \overline{b_i})^2(b_i^{'} - \overline{b_i^{'}})^2}} \tag{38}$$

where:

$$\overline{b_i} = \frac{1}{n}\sum_{i=1}^{n} b_i \qquad\qquad \overline{b_i^{'}} = \frac{1}{n}\sum_{i=1}^{n} b_i^{'} \tag{39}$$

2.5.2 Evaluation Measures of a Retrieval System

After computing the feature matches between two images I and $I^{'}$, we measure the similarity between them as:

$$S(I, I^{'}) = \frac{M_{II'}}{F_I.F_{I'}} \tag{40}$$

Where $M_{II'}$ is the feature matches between I and $I^{'}$ and F_I, $F_{I'}$ are the amount of extracted features in I and $I^{'}$ respectively.

The performance of any retrieval system needs to be evaluated to measure the quality of results and the usage of time and memory. The standard evaluation measures for retrieval systems are the precision, recall, mean average precision, and variance of recall [120].

To define these measures for a dataset D and query image set Q. Considering that, for a query image q a set $q_{relevant} \subset D$ of relevant images is determined. Let $q_{retrieved} \subset D$ is the set of retrieved images by a system for the query q. We compute the evaluation measures as follows:

2.5.2.1 Mean Recall MR The *recall* presents the amount of relevant and successfully retrieved images by a system and is computed as:

$$Recall(q) = \frac{q_{relevant} \bigcap q_{retrieved}}{q_{relevant}} \tag{41}$$

To compute the recall over a set Q of query images, we calculate the *Mean Recall* (MR) as follows:

$$MR = \frac{1}{Q}\sum_{q=1}^{Q} Recall(q) \tag{42}$$

The recall is very important evaluation measure since it presents how much of the relevant images is retrieved by a given system. To present the distribution of recall

values of individual query images around the mean recall, we compute the variance of the recall (VR) as follow [196], [176], [31]:

$$VR = \frac{1}{Q}\sum_{q=1}^{Q}(Recall(q) - MR)^2 \tag{43}$$

2.5.2.2 Mean Average Precision (MAP) Precision presents the number of relevant images in the retrieved set and is computed as:

$$Precision = \frac{q_{relevant} \bigcap q_{retrieved}}{q_{retrieved}} \tag{44}$$

To compute the precision over Q of query images, we calculate the Mean Precision (MP) as:

$$MP = \frac{1}{Q}\sum_{q=1}^{Q} Precision(q) \tag{45}$$

To present the positions and amount of relevant images in the set of retrieved images, we compute the *Mean Average Precision* (MAP) as:

$$MAP = \sum_{q=1}^{Q}\frac{Ap(q)}{Q} \tag{46}$$

where $Ap(q)$ is the average precision for image q and is given as:

$$AP(q) = \frac{1}{J}\sum_{i=1}^{J} Precision(i) \times r(i) \tag{47}$$

where $r(i) = 1$ if the i^{th} retrieved image is one of the relevant images otherwise $r(i) = 0$, $Precision(i)$ is the precision at the i^{th} position, J is the number of retrieved results.

2.6 Summary

In this chapter, the important aspects of the thesis were explained. Since we deal with content of images, we clarified first the concept of image features and their types. After that, we detailed the employed algorithms in this thesis to extract these features. To use the extracted features in solving image retrieval and near-duplicate retrieval tasks, we introduced feature matching and indexing methods. The compared features belong to different images but have the same type. Based on the amount of the correct feature matches, the similarity between images is computed. To evaluate the performance of a retrieval system, specific measures have been proposed in the information retrieval field. We introduced only the metrics that we applied through our research.

3 Related Work

In this thesis, we analyze and improve the near- and partial-duplicate retrieval of images by improving a method for feature extraction. We propose an approach to utilize the benefits of keypoints and color features to accelerate and enhance the detection of ND-images. Moreover, we speed up and improve the estimation of the spatial transformation between two near- or partial-duplicate images based on the spatial correlation identification between the feature matches.

Accordingly, we discuss in this chapter the current works that are proposed to enhance near-duplicate image retrieval. The traditional approaches and convolutional neural network techniques have been employed to detect and retrieve the near-duplicate images (as presented in Figure 19). Traditional techniques support image understanding and enable the user to describe and argue the correlation between the near-duplicate images. Therefore, we employ and improve the traditional approaches to detect and retrieve the near-duplicate images. We present techniques that focus on feature extraction improvement to lower the required costs (time and memory usage) of retrieval tasks or improve the retrieval results. In addition, we review approaches that use global features and combinations of global and local features to solve the retrieval tasks since global features require lower extraction and matching costs than keypoint features. However, keypoint features outperform global features in solving the ND-image retrieval tasks. Therefore, the combination of both types helps to enhance image retrieval.

To detect the spatial correlation between the near-duplicate images, non-deterministic and deterministic techniques have been proposed. The main idea of the non-deterministic methods is to fit all or a subset of feature matches to a model without any pre-processing step to identify the suitable candidates. The deterministic approaches split feature matches into correct and false groups and employ only the correct ones to estimate the possible correlations between images.

More recently, also some of the convolutional neural network approaches have been proposed to solve near-duplicate detection problems. However, most of these approaches have low performance due to the small size of near-duplicate benchmarks. Moreover, their performances decrease in case of viewpoint change or when many types of changes are applied to images.

In the following, we discuss the above-mentioned approaches in more details to motivate techniques that we employed in the evaluation process.

3.1 Advanced Global Features for Image Classification & Retrieval

The first step in image retrieval, classification, and near-duplicate detection is to extract image features. These features form the abstracted representations of images. They supply details about specific properties of images. Recent researches have

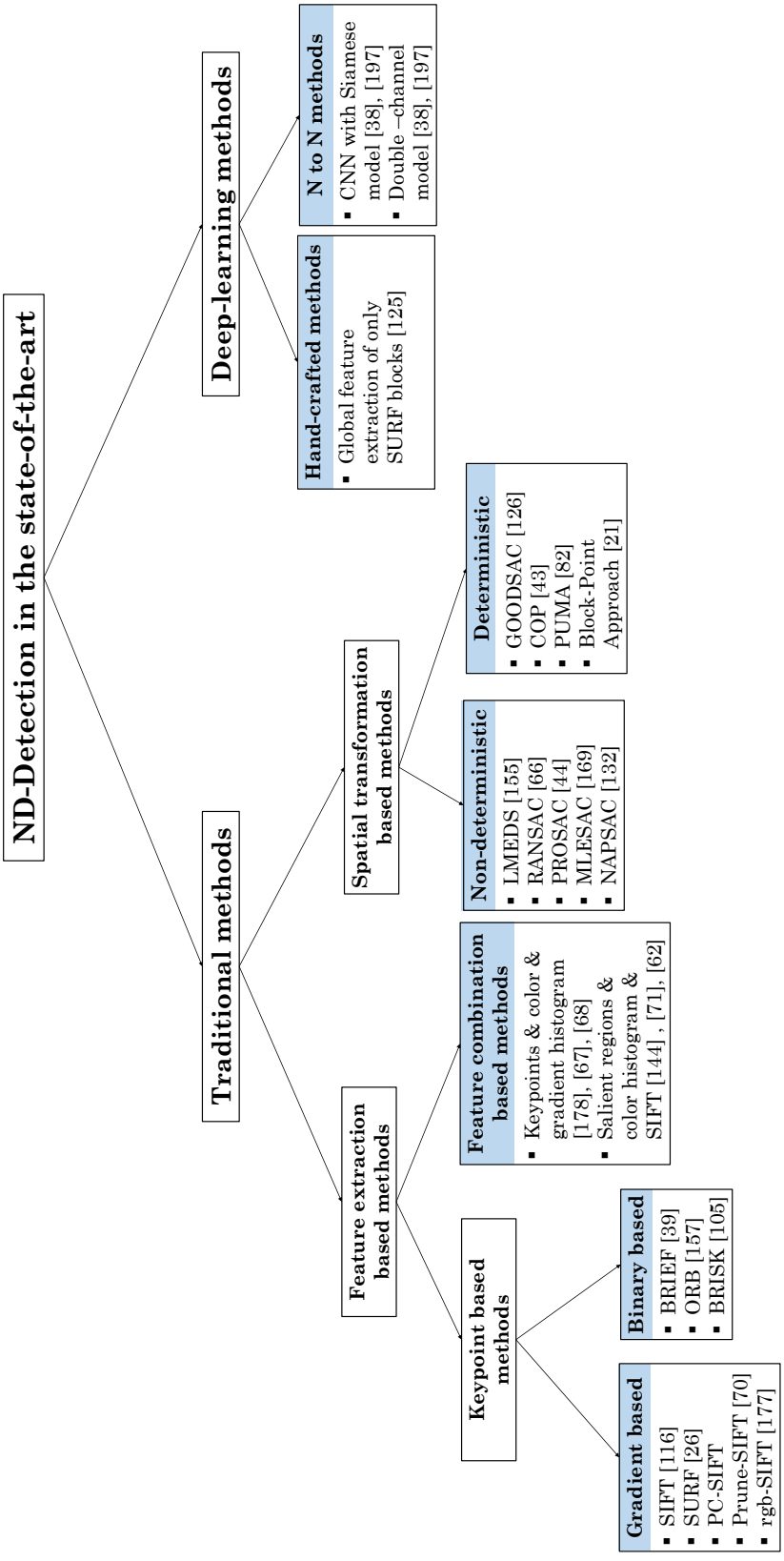


Figure 19: Overview of the state-of-the-art of near-duplicate detection and retrieval.

performed the classification and retrieval tasks by proposing the weighted sub-images gradient vector [138], [58], [198]. They achieved it by dividing the input image into sub-regions of equal sizes and then built the gradient vector for each one. After that, they completed a training stage to learn weights for each class of images. These weights were used later in the classification stage. In this way, images were classified based on their global properties into indoor vs. outdoor, city vs. natural landscape, highways vs. city center streets [139], [140]. Further researches employed the weighted gradient to ranking images based on their ruggedness, openness, roughness, naturalness, and expansion degrees [58], [140]. In [198], a generative model has been introduced based on the weighted gradient to classify images into eight categories i.e. coast, mountain, forest, open country, street, inside city, tall building, and highway.

The hierarchical color histogram with local hashing function has been proposed to identify the near-duplicate images [46]. The idea of the hierarchical color histogram is to divide the input image into four and then 16 sub-images. After that, concatenate the histogram of the image with those of the sub-images.

To classify images based on their color content, color features such as RGB, HSV, and L*a*b* have been employed [165]. The color moments and average color have been employed to improve image classification based on their color features. The concept of the average color is to filter out images that the distance between their color averages and the one of the query is higher than a specific threshold. The main idea of the color moments is to compute the average μ (first moment), the variation σ (second moment), and the skewness ς (third moment) overall color channels as follows:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} p_{ij} \quad , \sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_{ij}-\mu_j)^2} \quad , \varsigma_j = \sqrt[3]{\frac{1}{N}\sum_{i=1}^{N}(p_{ij}-\mu_j)^3}$$

hence the color distribution in an image has been presented as probability distribution [192], [59]. The fuzzy color histogram has been introduced by [80] to build the fuzzy histogram by processing the three channels of the RGB histogram simultaneously. The idea in [80] is to build clusters of colors based on the values of the red, green, and blue channels concurrently. After that, the distance is computed between the color value of each pixel and all constructed clusters. Hence, each pixel contributes various values to all clusters. In [41], [22], the color histogram has been optimized to improve the content-based image reverential. In [32], [98], it has been shown that the usage of the HSV color model outperforms the other color models such as RGB and L*a*b* [32], [98].

3.2 Improved SIFT Keypoints for Near-Duplicate Retrieval

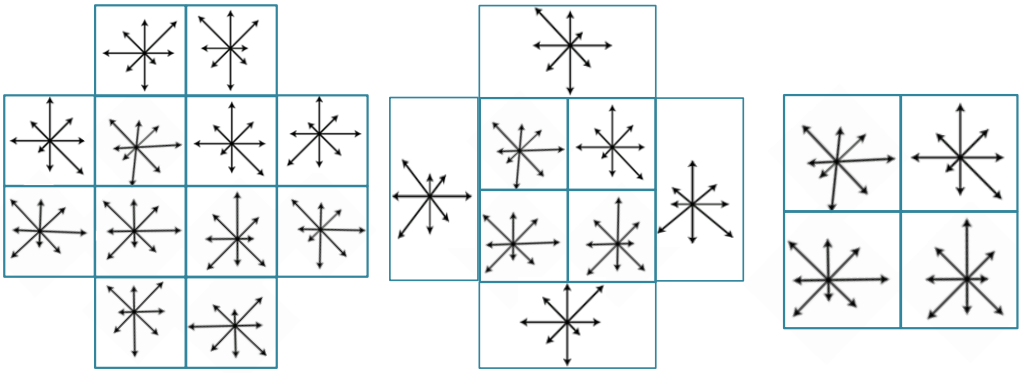
Local features, specifically Keypoint features, described in Section 2.3, have been employed in near-duplicate retrieval fields due to their invariant to affine transformations and robustness to viewpoint change, blur, adding noise and illumination change [116], [27], [105].

The SIFT algorithm described in Subsection 2.3.4 is still the most popular keypoint detector and descriptor since it outperforms most of the gradient and binary keypoint detector and descriptor methods such as SURF, ORB and BRIEF [96], [130]. In [167], it has been shown that the SIFT and BRISK algorithms produce the most invariant features to different types of image transformations. The experiments and results in [79] show that SIFT algorithm performs better than SURF and ORB when images are modified employing noise or fish eye filters. Therefore, many researchers have proposed methods to accelerate the matching process of the SIFT keypoints. However, those methods should preserve the performance quality of the SIFT features. The proposed methods either reduce the dimension of the SIFT descriptor [99], [102], or select a specific amount of SIFT keypoints based on their properties [70]. These properties are the scale, orientation and contrast of the SIFT keypoints described in Subsection 2.3.4.

Reducing the Dimensionality of the SIFT Descriptor The standard SIFT algorithm constructs its descriptors as vectors of oriented magnitudes of the areas around keypoints. Each vector contains $128d$ elements. The indexing and matching processes for a huge amount of such vectors are time and memory-consuming. To accelerate these steps, methods to reduce the length of the SIFT descriptors have been introduced in the recent works. The principal component analysis (PCA) has been employed in [99] to construct the PCA−SIFT. PCA−SIFT projects the high dimensional gradient vectors around keypoints to a lower-dimensional space by computing the principal components. The principal components are the eigenvectors of the covariance matrix of gradient vectors dataset [65], [103]. In [99], it has been proven that a set of top 20 eigenvectors is enough to project the descriptors into a lower space.

A method to reduce the dimensionality of SIFT descriptors to build vectors of 96, 64 or 32 dimensions, without any training stage, has been proposed in [102]. To construct the 96 dimensional descriptor, the four corners of the 4×4 patch around the keypoint are skipped as presented in Figure 20(a). As shown in Figure 20(b) to build the 64 dimensional descriptors the four outside corners are ignored and the rest outside neighboring patches are aggregated. The 32 dimensional descriptor is presented in Figure 20(c) and constructed by employing only the inside patches of the 4×4 patches i.e. a region of only 2×2 patches are regarded in building the SIFT descriptors. However, the original SIFT$-128D$ (D is dimensional) still performs better than the PCA−SIFT in case of adding blur, rotation and scale change [92]. The descriptors of the SIFT$-96D$ performs better than the proposed SIFT$-64D$ and SIFT$-32D$ [102].

Pruning SIFT Keypoints The amount of extracted SIFT keypoints rely on various factors such as the resolution of the input image, the number of octaves in the Gaussian pyramid, the initial value of the Gaussian filter, and the contrast



(a) SIFT 96 D (b) SIFT 64 D (c) SIFT 32 D

Figure 20: Reduce the dimensionality of the SITF descriptor as described in [102]. (a) the $96D$ descriptor is constructed by ignoring the four edges. (b) the $64D$ descriptor is built based on the $96D$ one by averaging the outer regions around a keypoint. (c) the $32D$ descriptor presents only the inner region.

threshold. The change of any of these factors affects the amount of the extracted keypoints. To accelerate the indexing and matching of SIFT keypoints without down-projection of high dimensional descriptors into lower space, a method to prune the extracted keypoints has been introduced in [70]. The contrast property of the SIFT keypoints presents their robustness. Therefore, the idea in [70] is to rank the extracted SIFT keypoints based on their decreasing contrast property. After that, select the set of the top N descriptors to present the input image. Similar results can be found by setting the contrast threshold to a particular value. However, in this case, the SIFT algorithm will not obtain any features for images with a structure of low-intensity change. Therefore, the hypothesis in [70] is to select a set N of keypoints with the highest contrast values and to accept all extracted keypoints in case of obtaining lesser than N keypoints. This idea of selecting a subset of SIFT keypoints has been evaluated to solve the task of near-duplicate image retrieval since, in this case, no need to match all keypoints between images [70].

Colored SIFT Descriptor As described in Subsection 2.3.4, the SIFT keypoints are extracted after transforming the input image into gray-level. To improve the performance of the SIFT algorithm in fields of object detection and image retrieval the colored SIFT descriptors (instead of the gray-scale descriptors) has been introduced in [33], [171], [172], [187]. In [33], the SIFT descriptors have been computed over the hue, saturation and volume channels to build 3×128 dimension descriptors. The constructed HSV-SIFT descriptors are scale- and shift-invariant however, they are not robust to the light change in the hue and saturation channels. Consequently, the 3×128 HSV-SIFT descriptor is not invariant to lighting condition change. The Hue-SIFT has been presented in [172] where the saturation value of pixels are convolved as weights in building the hue descriptors. Consequently, the Hue-SIFT descriptors

are scale and shift-invariant. In [171], the RGB-SIFT has been introduced. To build the RGB-SIFT descriptor, the same idea of building the SIFT descriptor has been applied to each of the RGB channels separately to obtain $3 \times 128 = 384$ dimension descriptors. The rgSIFT has been created by normalizing the RGB color space and computing the r and g values as follows:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} \frac{R}{R+G+B} \\ \frac{G}{R+G+B} \\ \frac{B}{R+G+B} \end{bmatrix}$$

Hence the rgSIFT has a descriptor of $2 \times 128 = 256$ dimensions. It has been described in [171] that the RGB-SIFT and the rgSIFT are more robust than the HSV-SIFT and Hue-SIFT to illumination change. However, all proposed colored descriptors of the SIFT algorithm have higher dimensions than the original SIFT. Accordingly, the construction of such descriptors [33], [171] is time and memory consuming. Therefore, they need a longer time than the original SIFT to complete the matching process.

3.3 Combined Features for Image Similarity Detection

To improve the content-based image retrieval, classification, and near-duplicate image retrieval, methods that employ combinations of features have been discussed

3.3.1 Global Features for Content-Based Image Retrieval

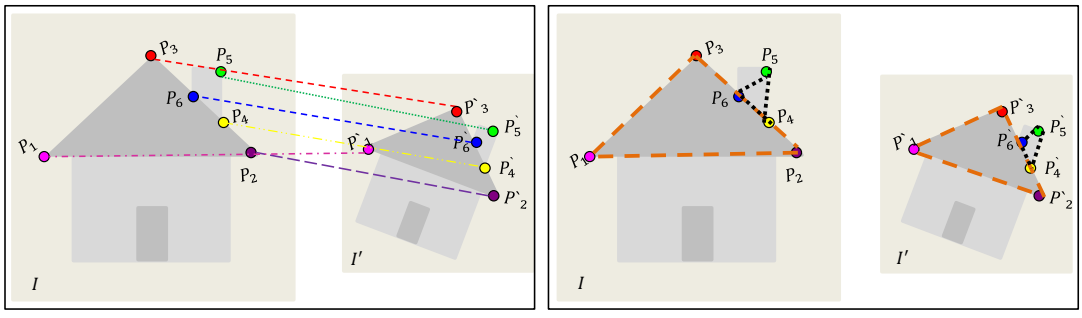
To improve the performance of image retrieval and classification systems, methods to combine more than one type of features have been introduced [141], [8], [133], [7]. The color descriptor and edge histogram of Mpeg7 [119] have been applied to detect similar images in [141]. In [8] content-based image retrieval has been accomplished by first convert the input image to the $YCbCr$ color space. After that, the Canny edges [40] are extracted in the Y space. Next, the edge map of the Y channel is combined with the Cb and Cr channels to build the RGB space. After that, the red, green, and blue histograms of the edge RGB image are constructed and employed to build 3×256 i.e. 768 descriptor vector. To reduce the length of this vector, the Discrete Wavelet Transform (DWT) of the second level is applied on the red Channel and of the third level on the green and blue channels to build a 128 length vector [78], [9]. The combination of the HSV color features, DWT, and edge histogram has been employed in [133] to represent each image with a vector of 310 elements.

The confusion of SURF keypoint features and histogram of the oriented gradient has been introduced in [124] to improve content-based image retrieval. A different idea to combine features has been discussed in [107] by grouping the images of a dataset based on their global features and then refine the groups employing the bag of SIFT features.

However, the methods that are described in [141], [8], [133], [7] and [124] classify images based on their similar content i.e. images are not necessary belong to the same scene. The classified images have similar structures and color distribution. These methods have been introduced to classify images into pre-defined classes such as flowers, beach, horses, elephants...etc.

3.3.2 Color and Keypoint Features for Near-Duplicate Image Detection

The motivation behind the combination of keypoints and global features in solving the near-duplicate retrieval task is that the keypoint matches between two near-duplicate images can be too few [178]. To overcome this problem, a method to combine the benefits of keypoints and gradient and color histograms have been introduced in [178]. After matching the keypoints between two images, the area enclosed within keypoint matches are computed as described in [67], [68]. As shown in Figure 21(a), given a set of keypoints $p_1, p_2, p_3, p_4, p_5, p_6$ in image I and their mapped keypoints $p_1^{'}, p_2^{'}, p_3^{'}, p_4^{'}, p_5^{'}, p_6^{'}$ in image $I^{'}$, the areas S_{123}, S_{456}, $S_{123}^{'}$ and $S_{456}^{'}$ enclosed by $p_1p_2p_3$, $p_4p_5p_6$, $p_1^{'}p_2^{'}p_3^{'}$, and $p_4^{'}p_5^{'}p_6^{'}$ (as shown in Figure 21(b)), respectively are computed. After that, the ratios $R_1 = \frac{S_{123}}{S_{123}^{'}}$ and $R_2 = \frac{S_{123}^{'}}{S_{456}^{'}}$ are computed. If these ratios satisfy $R_1 - R_2 = 0$ i.e. $R_1 = R_2$ then the keypoints are at most correct matches. Based on the spatial information of these matches, regions of interest are constructed in I and $I^{'}$. For these regions the gradient and color histograms are computed. If the histograms of the corresponding regions of interest are similar (the similarity between them is higher than a pre-defined threshold) then the images are near-duplicate even if the amount of keypoint matches is too few. This method can be applied as indicator of non-relevant images. In [144] the combination of color and



(a) matched keypoints (b) constructed regions

Figure 21: Matching keypoints as described [178]. (a) the mapped keypoints that satisfy the condition $R_1 = R_2$. (b) the constructed areas based on the keypoints matches.

SIFT features has been proposed to improve the retrieval of partial-duplicate images. The idea of [144] is to determine the salient regions in images as clarified in [71]. Next extract the SIFT features only within those regions. Finally, build the color histogram of each keypoint as described in [62]. To find the similarity between two

images, the SIFT keypoints are compared. After that, the color histograms of the similar SIFT keypoints are mapped together employing the Chi-Squared Distance described in Section 2.5.1. This technique has been applied to track objects or detect logos in images.

3.4 Estimate Spatial Transformations between Near Duplicate Images

The detection of the spatial correlation between near-duplicate images is very important to align images together and build panorama images, to check whether one image is part of the other and to detect the copyright violation [66], [126], [82], [44], [155]. Various techniques have been proposed to detect the correlation between the ND-images. These methods employ the keypoint matches between two images to estimate the transformation between them. They determine the correlation between images utilizing either non-deterministic or deterministic techniques. The most common non-deterministic basic algorithms are the RANSAC [66] and LMEDS [155] which estimate the transformation between images based on all feature matches i.e. without verifying whether they contain false matches. The deterministic methods filter out the false matches (outliers) and employ only the correct matches (inliers) to compute the correlation between images. To most common deterministic methods are PUMA [82], COP [43] and GOODSAC [126].

3.4.1 Non-Deterministic Methods

The concept of the non-deterministic methods is to find a model that is able to detect the most but not necessarily all of feature matches. In the following paragraphs, we describe the details of the LMEDS [155] and the RANSAC [66] and further improvement of the RANSAC [45], [169].

LMEDS The Least MEdian Squares (LMEDS) algorithm has been applied to estimate the geometrical transformation [155]. The principal concept of LMEDS is to compute a model for a subset of feature matches. Next, fit the rest feature matches into this model and calculate the median square error. Finally, select the model within the least median square error. The advantage of LMEDS is that it requires no knowledge about the distribution of the inliers and outliers. However, LMEDS fails to estimate the transformation between the ND-images when the outliers are more than 50% of the total feature matches [155], [53], [186].

RANSAC The RANdom Sample Consensus (RANSAC) approach [66] has been widely applied to identify the spatial correlation between images and to register the ND-images (i.e. align images and build panoramas). RANSAC is a random selection approach i.e. it employs a randomly selected small sample S_1 of the total feature

matches M to compute the transformation model between images. After that, it defines a set $S^{'} \subset M$ of feature matches that fit the defined model. If the size of $S^{'}$ is greater than a pre-defined threshold t_{max}, then the final model is estimated employing all members of $S^{'}$ with the least error. Otherwise, a different sample of feature matches S_2 is selected to repeat the previous process and compute a new model. If the number of iterations exceeds a specific threshold and still no set is found of size t_{max} or bigger, then the process is terminated and the RANSAC algorithm decides that no plausible transformation occurs between the feature matches.

RANSAC Improvement RANSAC performs very well when almost all feature matches are correct. When the amount of false matches increases the robustness of the RANSAC model decreases. Moreover, it fails in most cases to predict the transformation or estimates a wrong one. In addition, the number of iterations to find a suitable model is not determined. The RANSAC model is not repeatable since it computes the models based on randomly selected groups of feature matches [200], [83] therefore, it may estimate a wrong model when a set of outliers is employed to estimate the model. To improve the performance of RANSAC, extensions of it have been suggested such as NAPSAC [132] PROSAC [44], WaldSAC [45] and MLESAC [169]. The main idea of these extensions is to reduce the number of required iterations to compute the transformation between the images of the same scene.

To guide the RANSAC algorithm in selecting a subset of feature matches, the "*N Adjacent Points SAmple Consensus NAPSAC*" approach have been suggested [132]. The NAPSAC proposes that the correct matches locate, in general, closer to each other than the false ones. This idea has been employed in NAPSAC to select the best candidates to estimate the suitable model of RANSAC. Similar method has been introduced in [183] by bundling the SIFT keypoints to identify the partial-duplicated images. The first step in [183] is to extract the MSER blobs and SIFT keypoints of images. After that, images are matched based on their MSER features and SIFT features separately. Only the SIFT features that belong to mapped MSER blobs are considered to be correct keypoint matches. Based on this hypothesis, the outliers are filtered out, and the retrieved images are re-ranked.

To reduce the number of required iterations, the ''*Maximum Likelihood Estimation by SAmpling Consensus (MLESAC)*'' [169] algorithm has been introduced. MLESAC computes the likelihood for each estimated model and selects the model with maximum likelihood. To enhance the MLESAC, the guided-MLESAC has been suggested in [168]. The guided-MLESAC supplies the MLESAC with the prior probability of matches (i.e. probability of correct or false matches). The prior probability is computed based on the similarity degree between feature matches.

The PROgressive SAmple Consensus algorithm PROSAC [44] performs a pre-processing step by ranking the set feature matches based on decreased similarity. After that, PROSAC utilizes samples of the top-ranked feature matches to determine the transformation between images. The motivation behind this idea is the correct

feature matches have at most higher similarity than the others. The concept of PROSAC reduces the required iterations to fit the feature matches into a model and hence reduces the needed processing time.

3.4.2 Deterministic Methods

The problem of the non-deterministic methods is that, they are sensitive to the amount of the outliers, i.e. when the number of false matches increases they estimate at most wrong models. To overcome this problem, the deterministic models suggested classifying the feature matches into inliers and outliers. After that, they employed only the inliers to compute the transformation between images. In this way, they ensure that the outliers do not affect the quality of the estimated model. Many methods have been proposed in this direction such as GOODSAC [126], PUMA [82] and COP [43]. These approaches have been proposed to trace objects or logos.

PUMA The Putative Match Analysis (PUMA) method is one of the non-random approaches [82]. As presented in Figure 22, main concept of PUMA is to verify the correlation between the feature matches when one of them P_i of image I is selected and translated to its corresponding one $P_i^{'}$ in image $I^{'}$. To avoid the effect of scale and rotation differences, the polar coordinate system is employed and centered at point P_i. The vectors between P_i and all feature matches in I and $I^{'}$ are constructed. After that, PUMA computes the length of vectors and normalize them in the range $(0, 1]$ and compute the cosine of angles between each pair of vectors that start with P_i and end with P_j and $P_j^{'}$ respectively. This process is repeated by translating one of the feature matches at a time. Afterward, the relative polar matrix of the relative length and cosine angles is plotted. PUMA hypothesizes that the inliers build one cluster, whereas the outliers scatter away from this cluster. PUMA removes the outliers and repeats the previous steps until having a cluster where the distance between its centers and elements is smaller than a pr-defined threshold. As result, PUMA guarantees that the selected matches are all correct. However, the computations of PUMA are expensive and it causes skipping a subset of correct matches since the process of removing the outliers is repeated frequently until satisfying specific criteria.

COP To track the presence of a specific object or logo between images, the Combined-Orientation-Position (COP) consistency graph model has been introduced [43]. The COP approach utilizes the spatial location and orientation of feature matches to filter out the outliers. Given two pair of matches $(P_i, P_i^{'})$ and $(P_j, P_j^{'})$. As shown in Figure 23, the idea of COP is to employ the polar coordinate system and locate the original point at point P_i, so that the principle-axis is parallel to the descriptor vector of P_i and has the same direction. The circle around P_i is divide into sectors. Next, COP identifies sector, that P_j belongs to it and the orientation of P_j

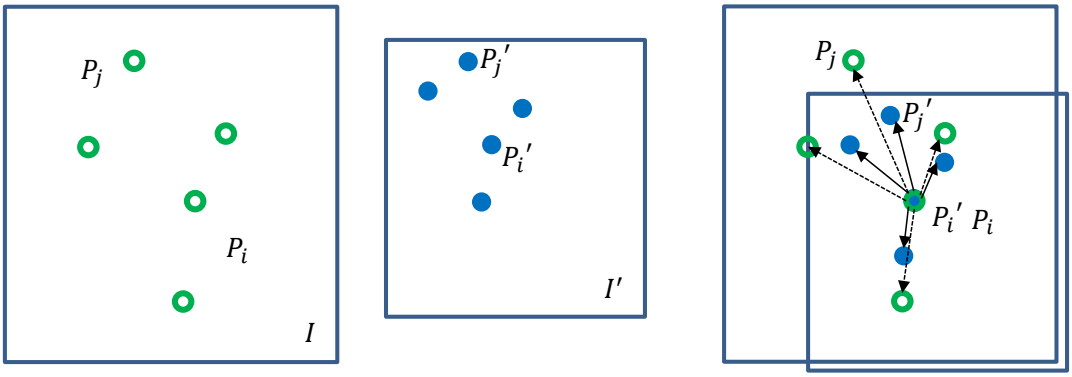


Figure 22: The concept of PUMA model. The keypoint P_i of image I is translated to its corresponding point $P_i^{'}$ of image $I^{'}$. After that, the vectors between P_i, P_j and $P_i^{'} P_j^{'}$ are constructed. This step is repeated for all feature matches between images I and $I^{'}$.

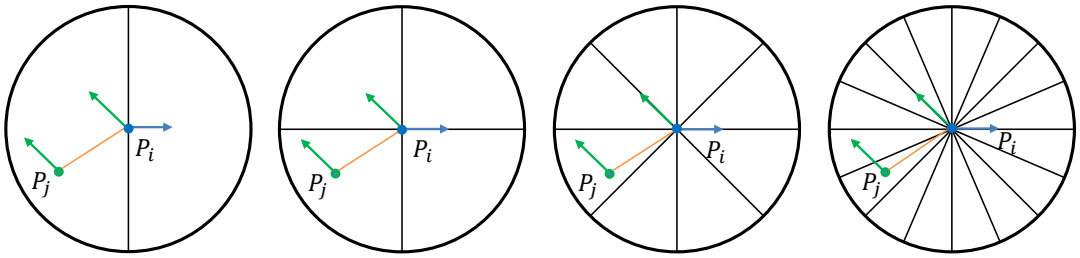


Figure 23: The idea of the COP method to identify outliers. The polar coordination system is centered and oriented at keypoint P_i. The position and orientation of P_j are computed by dividing the circle around P_i into 2, 4, 8 and 16 sectors.

with respect to P_i. The same computations are completed for the keypoints $P_i^{'}, P_j^{'}$. If the keypoint $P_j^{'}$, with respect to $P_i^{'}$, belongs to the same sectors like P_j then the pairs $P_i, P_i^{'}$ and $P^j, P_j^{'}$ are at most correct matches. This process is literately repeated using various number of sectors i.e $2, 4, 8$, and 16 sectors. Based on the output of the previous process and employing the idea of [142], the dominant set (i.e. the correct set) of feature matches is defined. COP filters out the false matches without estimating the kind of spatial correlation between images. Moreover, it is an expensive method since it requires verifying the consistency of two features over four levels. In case of starting with one of the false matches, it is impossible to detect it immediately.

Block-Point Matching Approach This approach has been introduced in [21] to detect copy-moved objects. The idea of this approach is to extract keypoints from an image. The detected keypoints are employed to segment the image into blocks applying the Delaunay triangulation [69], [61]. Each Triangle is represented by its dominant color and inner angles. To eliminate false matches before comparing any two triangles, their areas A and B are computed and the following condition

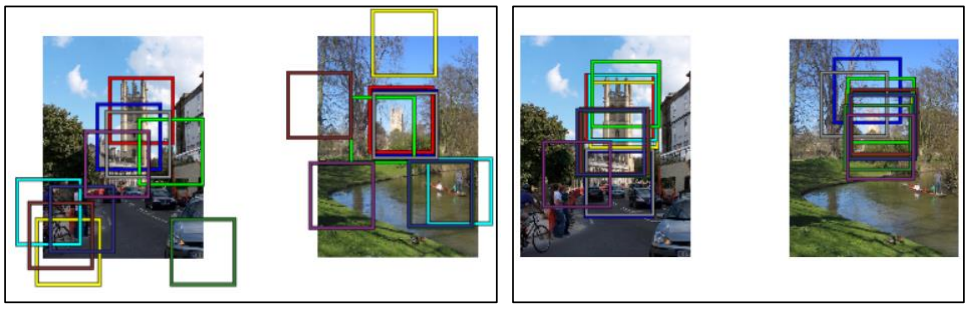
is justified $\frac{min(A,B)}{max(A,B)} \leq 0.25$. Finally, the similarity between the inner angles and dominant colors are computed. The idea of block-point matching approach allows the detection of similarity in case of rotating of the copied-moved object.

3.4.3 ND-Retrieval using CNN Models

Convolutional Neural Networks (CNN) currently exceed the traditional methods in solving image retrieval tasks. However, most CNNs based methods need massive annotated datasets for the training stage, therefore, the training stage is expensive. The performance of CNNs decreases when the target image dataset varies from the trained one hence, for each dataset with a new structure, the training stage should be repeated. The use of training datasets without annotations has been discussed in [149], [64], [150] utilizing the fine-tuning convolutional neural network models. However, the presentation of features that guide the neural network to specify the similarity between images shows many false region matches. Therefore, the scenario of learning the weights based on those features is unclear [149]. Figure 24 shows the related components that are employed to identify the similarity between images by applying the VGG neural network [160]. However, the utilizing of the fine-tuning of VGG [149] presents better-learned components. Convolutional Neural Networks (CNN) currently exceed the traditional methods in solving image retrieval tasks. However, most CNNs based methods need massive annotated datasets for the training stage, therefore, the training stage is expensive. The performance of CNNs decreases when the target image dataset varies from the trained one hence, for each dataset with a new structure, the training stage should be repeated. The use of training datasets without annotations has been discussed in [149], [64], [150] utilizing the fine-tuning convolutional neural network models. However, the presentation of features that guide the neural network to specify the similarity between images shows many false region matches. Therefore, the scenario of learning the weights based on those features is unclear [149]. Figure 24 shows the related components that are employed to identify the similarity between images by applying the VGG neural network [160]. However, the utilizing of the fine-tuning of VGG [149] presents better-learned components.

Deep learning techniques have been employed to identify near-duplicate images. Alexnet and VGG16 networks, which compute global CNNs features, have been applied on the double-channel and Siamese models [38] to classify the near-duplicate images. The evaluation in [197] presents that the double-channel model obtains higher accuracy than the Siamese model. Alexnet with the double-channel model is more robust than VGG16 with double-channel network. The MAP of Alexnet with double channel model exceeds 90% for the image groups with slight change (such as rotation and scale change) [197]. However, the MAP of both methods suggested in [197] decrease to about 60% when changes such as illumination or viewpoint change are involved. However, the hand-crafted method, suggested by extracting the global CNNs features of image blocks where SURF keypoints are located, improves

the performance of near-duplicate classification in the above-discussed cases to about 78% [125]. The discussion in this section implies that CNNs techniques improve the performance NDs-retrieval but they are not robust to viewpoint change (similar to traditional methods) [197].



(a) VGG neural network (b) VGG neural network with fine-tuning

Figure 24: The features that aid the neural networks to detect determine the similarity between images. (a) the comparable features employing the VGG neural network. (b) the matched regions by applying the fine-tuning VGG model. The boxes of the same colors present the corresponding components.

3.5 Discussion

In the field of image near-duplicate retrieval, the main application of neural network techniques is to improve the performance of similarity detection between images belonging to the same scene when big image datasets are employed. However, as shown in Figure 24, most of them do not focus on why two images are similar, i.e. whether the feature matches are true or false. In this work, we focus on image understanding and the efficiency of the extracted features of images. In addition, we focus on the meaning of feature matches, therefore, we filter the list of feature matches before computing the similarity between two images. For our approaches, improving the quantity of performance is not as important as the quality of it. Moreover, most of our approaches do not need a training stage. Therefore, we do not need big image datasets like in neural networks. Regarding the clarified arguments, we do not compare our approaches to deep learning methods. However, in some sections (such as in Section 5.1.9), we gave a qualitative comparison to neural networks. This is to present that deep learning techniques behave similarly to traditional methods, i.e. even when neural networks achieve better performance than traditional methods, their performance drops in same cases where the performance of the transitional approaches decreases.

3.6 Challenges of Near-duplicate Retrieval

In the scope of this thesis, our principal focus is to discuss and improve the retrieval of ND-images regarding the following challenges:

3.6.1 Time Complexity

To solve ND-retrieval tasks, high-dimensional features have been almost extracted from images. The extraction and matching of thousands or millions of these features are time and memory-consuming. Moreover, the stage of correlation detection requires an expensive process to iteratively check the feature matches and fit them into a model. Therefore, the recent works proposed several methods to decrease time complexity. These methods fall under four main categories:

- Methods reduce the dimensionality of feature vectors. This reduction accelerates the matching stage.
- Methods optimize feature matching. Through this optimization no need to compare each feature of one image with all of the others.
- Methods accelerate the correlation detection step.
- Methods filter the extracted features based on specific properties to reduce the number of features which in roll decreases the required time to complete the matching process.

These methods (as described in Sections 3.2, 3.3 and 3.4) reduce time complexity. However, many of them decrease (or at least fail to improve) the performance of solving the ND-retrieval task.

3.6.2 Spatial Correlation between Feature Matches

In content-based image retrieval systems, matching only low-level features, that are extracted of all patches of an image without a pre-defined task, is not enough to overcome the identify the spatial correlation between the image ND-images. Hence non-relevant images may appear on the top of the retrieved images since they have similar colors, textures, or objects. However, in case of ND-retrieval, many of the current methods exploited the properties of features, specifically features that represent the distinct patches of images in terms of vectors, to define the spatial correlation between feature matches. This correlation has been employed to determine images that belong to the same scene or to specify the presence of specific objects. These methods (clarified in Section 3.4) fall into two categories:

- Non-deterministic methods: Their main idea is to find a model that fits the most of feature matches. This is done without justification, whether the feature matches include outliers. Therefore, it produces incorrect models when the

amount of false matches increases. Figure 26 presents an example where the LMEDS, RANSAC and PROSAC methods cannot estimate the correct correlation between images. The outliers in both cases are lesser than 44% of the total matches. We identified the outliers employing our method introduced in Chapter 7.

- Deterministic methods: Contrary to non-deterministic methods, the deterministic systems suggest specific criteria to filter the outliers. The filtering process requires a lot of computations and iterations to return a set of correct matches. This set is applied to estimate the correlation between images.

To overcome these issues, we introduce our solutions in Chapters 5, 6 and 7. In Chapter 5, in the field of near-duplicate image retrieval, we present our methods to compress the dimensionality of the SIFT descriptors to build the *region compressed SIFT* without the need for a training stage and without skipping any region around a keypoint. We justified that our method is invariant to rotation and illumination change and robust to adding noise, viewpoint, and scale changes. To accelerate and improve the matching quality based on the properties of the SIFT keypoints, we involved the scale and contrast of keypoints to prune the set of extracted keypoints. Moreover, we include weights based on scale, contrast, and orientation properties of SIFT keypoints. The details of employing the properties of the SIFT keypoint are in Section5.2.

As discussed in Section 3.3, the building of global features such as color or gradient of images is faster in construction and matching steps than keypoints. On the other hand, Keypoints outperform global features in solving the near-duplicate retrieval tasks. Figure 25 presents a list of query images and their top three retrieved results employing the SIFT algorithm in the first row and the HSVcolor histogram in the second. The results show that the SIFT algorithm performs at most better than the HSV color features. To get the advantages of both kinds of features, we proposed in Chapter 6 our method to accelerate and improve the near- and partial-image retrieval. We achieve this by matching images based on their color features and then re-rank the top retrieved images employing their extracted SIFT features. In this way, we reduce the required time to match the SIFT features between queries and dataset images.

To enhance and accelerate the detection of the spatial correlation between the near- and partial-duplicate images, we introduce in Chapter 7 our models that can identify the outliers even when they form more than 50% of the total matches. Based on the set of inliers, our models predict the spatial correlation between the near- and partial-duplicate images.

3.7 Summary

In this chapter, we described related work to our research. We started in Section 3.2 by discussing methods that focus on reducing the cost of the feature matching step

Figure 25: Samples of queries and their top three retrieved images. For each query, there are three near-duplicate images in the dataset. The first row presents the retrieved images using global features (HSV color histogram). The second row displays the retrieved images employing SIFT keypoints. The green frames show the relevant retrieved images. The red frames present the non-relevant retrieved images.

(a) Queries (b) False matches (c) Correct matches

Figure 26: Samples of queries and their relevant images where LMEDS, RANSAC and PROSAC models fail to estimate the transformation between images. (a) query images. (b) false matches (outliers) in the first example six out 14 and in the second ten out 23 are outliers. (c) the correct matches.

by either projecting the high dimensional keypoint features into lower space or by pruning them based on their properties.

To improve the retrieval of the relevant images in the top results, we presented the combination of more than one kind of feature in Section 3.3. This is done by representing images by vectors that contain color and gradient details. To enhance the retrieval of ND-image, the combination of keypoint and color features was described in Subsection 3.3.2.

Section 3.4 discussed some proposed methods to filter out the false matches and detect the spatial correlation between the ND-images. The introduced methods in the recent works are either not robust to the increased amount of outliers or apply strict conditions to filter them and hence classify part of correct matches as outliers.

The open challenges of the current works were reviewed in Section 3.6. Our proposed methods to solve those issues are presented in the next chapters.

4 Benchmark Datasets

In this thesis, we employed benchmarks of different domains to evaluate the performance of our approaches in Chapters 5, 6, 7 and 8 and in Appendix B and C in solving the near-duplicate retrieval tasks. The reason for using various types of benchmarks is to check the performance of our approaches using images of different contents and structures. Table 1 gives an overview of the datasets that we used and taken from other research. Table 2 displays the datasets, that we generated for our goals. In Table 2, we present only the main dataset that we created, that are Oxford-Zoomed-in, OXB, Panorama, PANO, ATRANS, Aerial, PAIN, Aerial Benchmark, and PAIN. Several minor benchmarks have been created for specific purposes introduced in Chapters 5, 6 and 7.

In the following sections, we describe the main datasets, that we applied in this thesis.

Table 1: Image datasets overview. This table presents ND-images datasets, that have been introduced by other researchers.

Dataset	#Images	Sub-images	Description
Homography Benchmark	48	-	8 different scenes. Employed in [128], [123], [52] and [170]. Used in Appendix B
UKBench	10,200	-	2,550 scenes. Used in[136], [195], [177] and [113]. Used in Chapters 5 and 6 [13], [14], [15]
Caltech	250	-	High resolution images of 50 buildings. Used in [88] and [10]. Utilized in Chapter 5 [15], [14]
Duplicated Objects	920	-	Copy and move objects. Same background with modified copy of an object. Utilized in Chapter 8

4.1 Homography Dataset

This dataset contains 48 images of eight scenes [5]. For each scene, six altered images were created by applying one kind of five modifications. These modifications are viewpoint change, scale change, blur, illumination change, and JPEG compression. For each type of modification, one or two scene were employed, for each one original image *img1* and five transformed images, that are *img2, img3, img4, img5, and img6*. The amount of convolved transformation increases from *img2* to *img6*. The

Table 2: This table presents the datasets, that we generated (e.g. by extracting sub-images created by modifying the sub-images).

Dataset	#Images	Sub-images	Description
Oxford-Zoomed	5062	3×5062	Indoor/outdoor images of 10 buildings. Employed in [143], [20] and [24]. Utilized in Chapters 6 [17] and 7 [12]
OXB	500	$50,000$	Picked up from Oxford Buildings. We generated sub-images and used them in Chapter 7 [12]
Panorama	250	$50,000$	We collected it form [2]. Utilized in Chapter 7 [16], [12]
PANO	250	$40,000$	Collected form [2]. Utilized in Chapter 7 [12]
ATRANS	$60,000$	-	Affine transformed images generated by us using Panorama dataset [2] Utilized in Chapter 7 [12]
Aerial	1000	$20,000$	Picked from AID [185]. Used in [114] and [91]. Utilized in Chapter 7 [12]
PAIN	1000	$36,000$	taken from "Your Paintings" [1]. Used in [51]. Utilized in Chapter 7 [12]

transformations were accomplished by changing the camera settings. The images of Homography dataset are presented in Figure 27 and has been used in many types of research to analyze and compare the affine invariant properties of various types of image features [128], [123], [52] and [170]. We utilized this dataset in Appendix B to compare the performance of various keypoint detector and descriptor algorithms i.e. the SIFT, SURF, BRIEF, ORB, and BRISK employing various threshold.

4.2 UKBench Dataset

UKBench dataset [135] has been used in near-duplicate detection and retrieval researches to present the effect of various kind of feature detectors and descriptors [86], [102] and [166], to evaluate feature structuring methods [136], [195], [177] and [113] and to introduce the role of CNNs in ND-images classification [197]. UKBench contains $10,200$ images of $2,550$ different scenes. The four images of each scene include either slight change (such as rotation or scale change) or significant

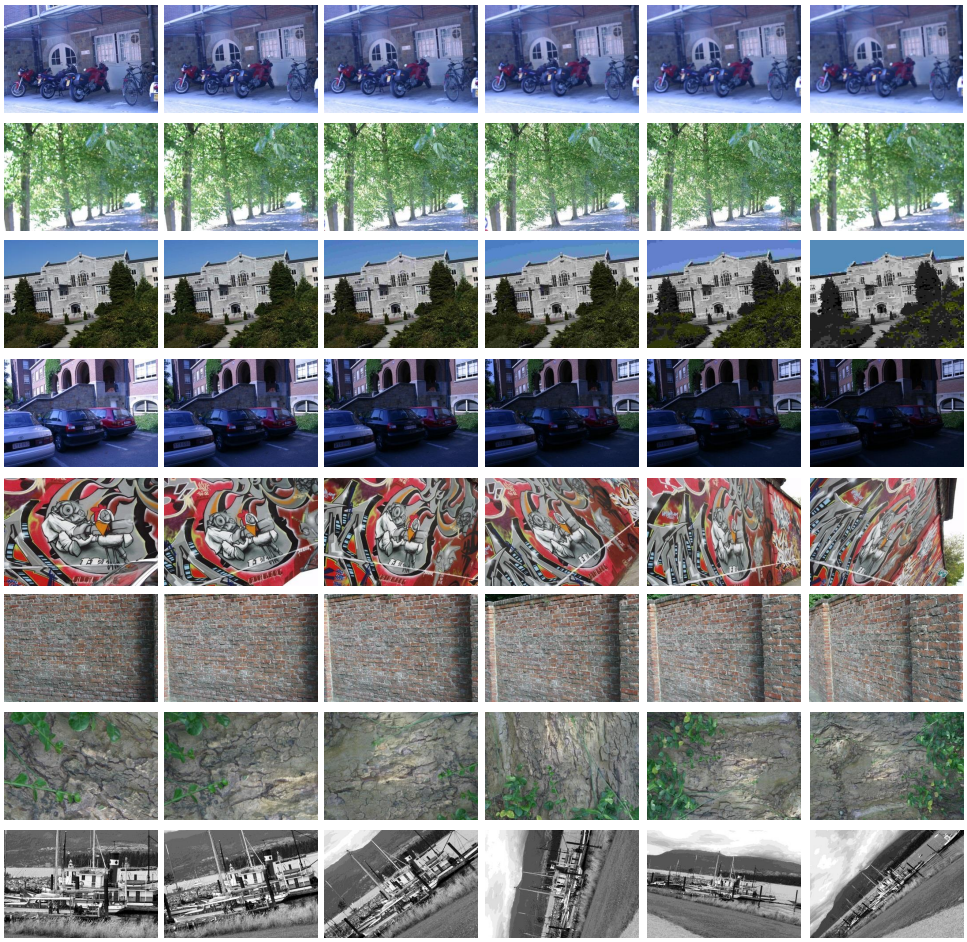


Figure 27: Transformed images taken from [5]. The first column presents the original images (img1). Columns 2 to 6 present the transformed images employing increased values of transformations. The transformations are: blur (first & second rows), JPEG compression (third row), illumination decreases (fourth row),viewpoint change (fifth and sixth rows) and zoom & rotation (seventh and eighth rows).

change such as viewpoint and illumination changes or appearing (disappearing) of objects. The images of UKBench has the size 480×640. We utilized this dataset in Chapter 5 to evaluate the performance of our improved SIFT. In addition, we employed the UKBench dataset in Chapter 6 to analyze the effect of feature combination in solving the near-duplicate retrieval task [13], [15], [14], [17].Figure 28 presents samples of UKBench dataset.

4.3 Caltech Dataset

The Caltech dataset [11] contains 250 images of 50 buildings of the Caltech campus. For each, five photos were captured using different viewpoints and scales. This dataset has higher resolution images than the UKBench. They have the size of

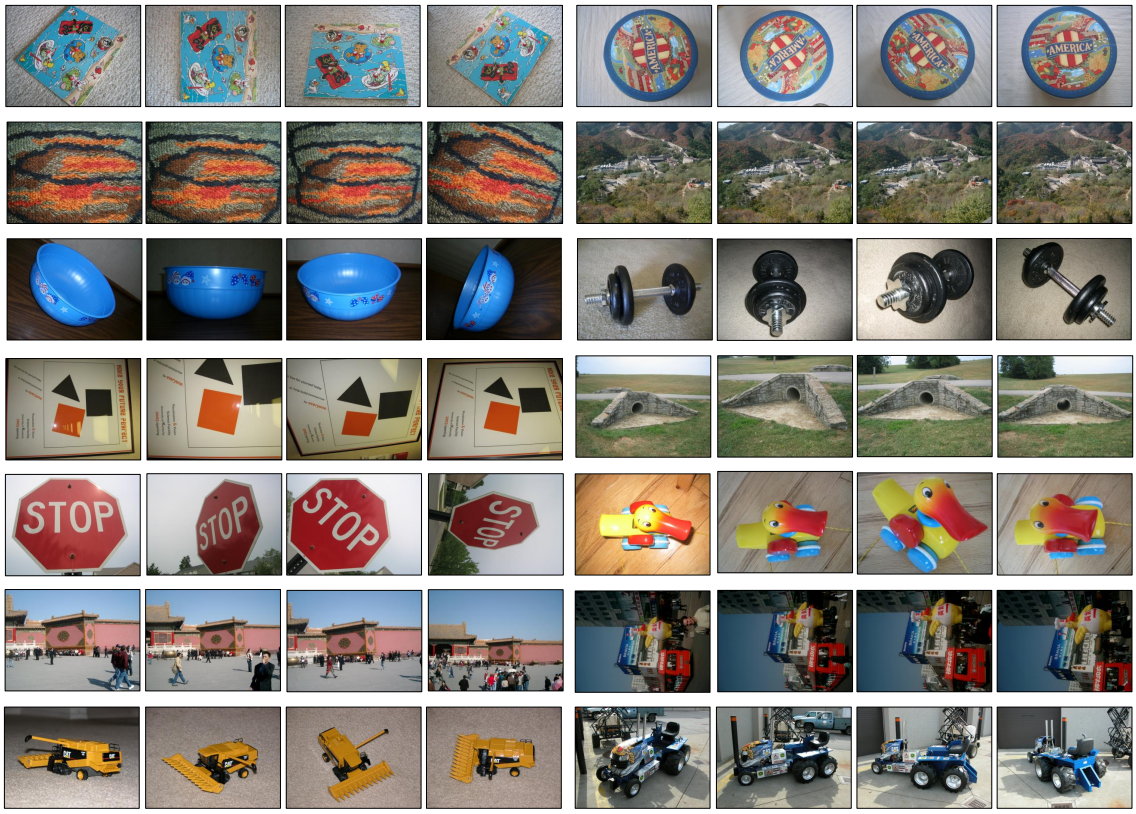


Figure 28: Samples of the UKBench dataset. The first and second rows present ND-images with slight change in scale or rotation. The third and fourth rows show ND-images with viewpoint change. The fifth row displays ND-images with lighting and viewpoint changes. The sixth row presents images with scale change and appearing/disappearing of some objects. The seventh row displays objects of different perspectives.

$2,048 \times 1,536$. Caltech dataset has been utilized to solve ND-retrieval tasks [88] and to compare the performance of global and local features [10]. We used this dataset in Chapter 5 to evaluate the performance of our improved SIFT algorithm [15], [14]. The images of this dataset differ in viewpoint, scale, and lighting condition. Figure 29 shows samples of this dataset.

4.4 The Oxford Buildings Dataset

The Oxford Buildings dataset includes $5,062$ images of 10 different landmarks of Oxford taken from Flicker [6]. As shown in Figure 30 for each landmark indoor/outdoor images were included i.e. the images of the same sight are not near-duplicate. This dataset has been used to solve the tasks of image retrieval based on the existence of specific objects [143], places recognition [20] and image retrieval using neural-networks [24]. We utilized this dataset in Chapter 6 to evaluate the performance of feature combination in solving the ND-retrieval task. For this, we randomly

Figure 29: Samples of the Caltech Building dataset. The first row presents ND-images with slight changes in viewpoint and scale. The second row shows images with a big change in viewpoint. The third row presents images with scale and viewpoint changes. The fourth row displays images with lighting and viewpoint changes.

cropped the Oxford Buildings images to produce sub-images that cover 50%, 25% and 10% of the size of original images. Hence, we built Oxford-Zoomed-in-50, Oxford-Zoomed-in-25, and Oxford-Zoomed-in-10, respectively.

We produced a subset of 500 images of the Oxford buildings dataset and used them in Chapter 7. In this case, we produced the Oxford buildings sub-image dataset (OXB) by cropping 20 sub-images of each image. After that, we scaled the sub-images using the ratios 30%, 50%, 100%, 200% and 300% to produce 100 sub-image for each original image. Hence the OXB sub-image dataset contains $50,000$ images.

4.5 Panorama Dataset

To analyze the geometrical correlation between near-duplicate images, we constructed the "Panorama" sub-image dataset using panorama images downloaded from [2]. Each panorama image consists between six and 30 segments and has a resolution of more than $4,000$ pixels in width or height. They are images of landscapes or sights. We employed ten various ratios to determine the area of panorama images that are covered by the sub-images. These ratios are: 56%, 40%, 35%, 30%, 25%, 20%, 15%, 10%, 5% and 2% of the size of original panoramas. Figure 31 presents two panoramas and their sub-images. For each, four sub-images of different locations are selected randomly. After that, we applied the cubic interpolation model [101] to

Figure 30: Samples of indoor/outdoor Oxford landmarks dataset. The first row presents images of the "All Souls College". The second row shows images of the "Ashmolean Museum". The third row presents images of the "Christ Church". The fourth row displays images of "Magdalen College".

scale-up/down (i.e. increase/decrease the resolution) the sub-images. The scaled-up image has a 150% resolution of the original sub-image. Using the same model, we created four scaled-down sub-images. These scaled-down images have 70%, 50%, 30%, or 15% of the resolution of original sub-images. Based on these settings, we generated 200 sub-images for each panorama. Utilizing 250 panoramas, a sub-dataset of size $50,000$ is built. We introduced and applied this dataset in [16] and in Chapter 7

We generated the "PANO" sub-image dataset using the panorama images too. In this case or each panorama, we created 200 sub-images that differ in size, area, and resolution. For this, we cropped randomly ten sub-images (rectangular regions) of each panorama image that cover areas between 4% and 15% of the size of the original images. For each sub-image, we generated five scaled images i.e. one using the scale factor 100%, two up-scaled images by applying the ratios 200% and 300%, and two down-scaled images using the ratios 50% and 30%. Hence, the PANO sub-image dataset contains $20,000$ sub-images.

For further analysis, we applied different kinds of transformation to create the Mixed Affine Transformation (ATRANS) dataset. The ATRANS includes $60,000$ images generated using the original 250 panorama images. We set in the ATRANS dataset five types of transformations i.e. the original panorama image, ten sub-images

(cover between 4% and 15% of the size of the image), 15 rotated sub-images (using rotation degree in the range $(0^o, 120^o]$), two flipped images (horizontal and vertical). These transformed images are scaled using the factors 30%, 50%, 100%, 200% and 300% to complete the building of this dataset. We applied the PANO and ATRANS datasets in Chapter 7 [12].

We applied further modification on the PANO dataset to check the robustness of our approaches in Chapter 7 to different types of image altering such as adding noise or blur, illumination change, and rotation [12].

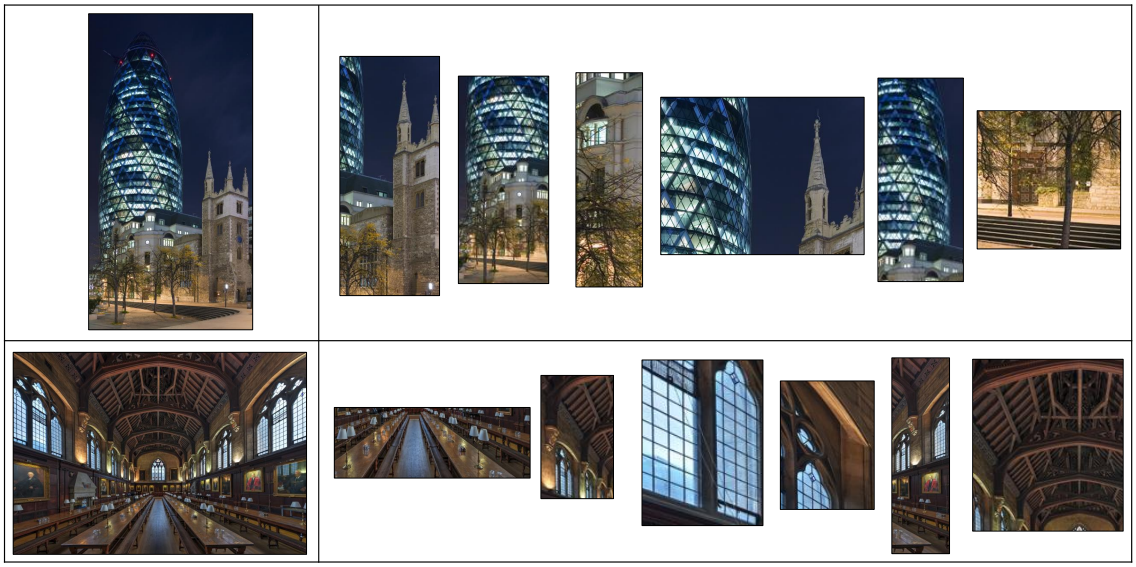


Figure 31: Samples of the Panorama dataset. The first column presents the panorama images. The second column shows their sub-images.

4.6 Aerial Dataset

The Aerial dataset (Aerial) is a set of a published aerial benchmark called "Aerial Image Dataset (AID)" [185]. The AID dataset has been introduced in [51] to solve the problem of aerial scene classification. It has been used in many remote scene understanding and classification research [114], [91]. The AID dataset contains 30 categories of remote scenes. These are: *airport, bare land, baseball field, beach, bridge, center, church, commercial, dense residential, desert, farmland, forest, industrial, meadow, medium residential, mountain, park, parking, playground, pond, port, railway station, resort, river, school, sparse residential, square, stadium, storage tanks, and viaduct* [185]. Each category includes 220 to 420 images of size 600×600. As shown in Figure 32, for our experiment, we selected 200 images of airports and 800 images of buildings (medium residential, industrial, commercial, and school) to produce $20,000$ sub-images. We did not select images of other categories since many

of them contain images of a unique color hence, no keypoints will be exacted from them.

To generate the sub-images, we cropped five sub-images of the original images using the ratios between 4% and 15% of the area of images. After that, we rescaled each sub-image employing five scaling factors 30%, 50%, 100% and 200%. We utilized this dataset in Chapter 7 [12] to predict the geometrical correlation in case of aerial images.

Figure 32: Samples the Aerial dataset. It includes airplane and buildings images. The first column presents the original aerial images and the second displays the sub-images

4.7 Paintings Dataset

We utilized the free accessible dataset "Your Paintings" [1] to create the Paintings dataset (PAIN). This dataset has been used in [51] to discuss the problem of object retrieval in paintings. In our research, we employed it in Chapter 7 [12] to check whether we can estimate the geometrical correlation in case of paintings. To construct our dataset, we picked $2,000$ paintings. Since images of PAIN are not rich with details, we created for each one of its images six sub-images that cover between 4% and 15% of the PAIN images. After that, we down-scaled and up-scaled the sub-images using the ratios 30%,100% and 200% to produce $36,000$ sub-paintings. Figure 33 displays samples of this dataset.

4.8 Duplicated Objects Dataset

Duplicated Objects dataset contains images of 20 different scenes. For each image, 46 modified copies have been generated by selecting a specific object of an image

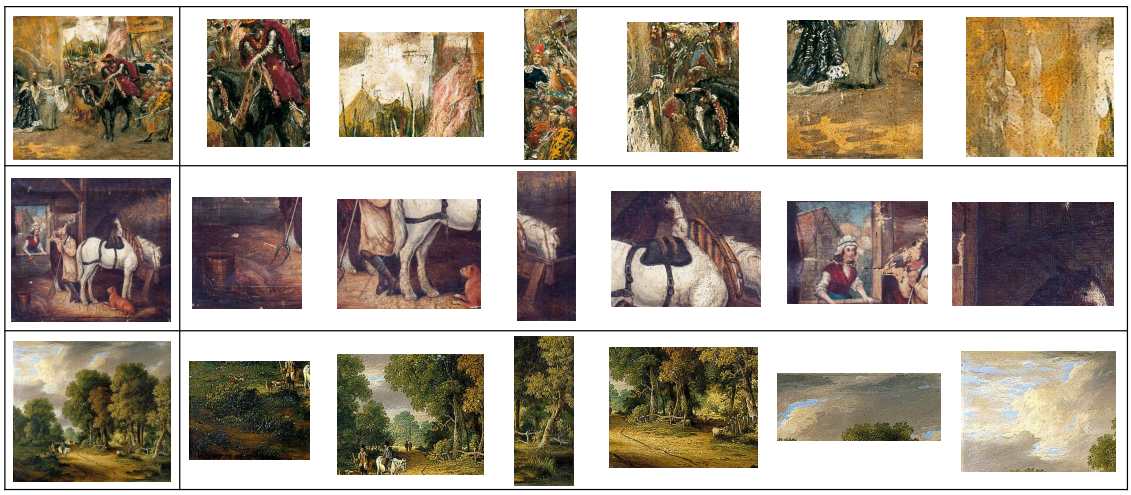


Figure 33: Samples the Painting dataset (PAIN). The first column presents the original paints. The second column shows some of their sub-images.

and duplicating it in the same image. The duplicated object has the same properties as the original one but it has been rotated, scaled-up/down, or flipped using various ratios to produce 46 duplicated objects copies. The scale ratio is set to be between 25% and 200%. This dataset contains 920 images and has been introduced and employed in [21]. We utilized it in Chapter 8 to detect the difference between two images of same scene when only one object is duplicated in one of them. Figure 34 presents samples of the Duplicated Objects dataset. The first row in Figure 34 presents a simple case of one object and background. The second row displays two objects in the foreground and rich with details background. The third shows the case of existing many objects in the foreground.

Figure 34: Samples of the Duplicated-Objects dataset. The first column presents the original images the second shows a mask with the duplicated object. The third displays the modified images with the duplicated object. In the third column, the duplicated objects appear without any change, up-scaled, down-scaled, and rotated with different degrees, respectively.

4.9 Summary

In this chapter, we presented the main datasets that we used to evaluate the performance of our approaches. We utilized six datasets that have been used in other research i.e. Homography, UKBench, Caltech, Oxford Buildings, and Duplicated Objects. Moreover, we created some datasets to evaluate our approaches in solving specific tasks such as sub-image retrieval. Our main generated datasets are Oxford-Zoomed, OXB, Panorama, PANO, ATRANS, Aerial, PAIN, Aerial and PAIN. For some specific tasks in Chapters 6 and 7, we created some small datasets utilizing UKBench and PANO dataset. We described them in Chapters 6 and 7.

*We present in this chapter two approaches to solve **RQ.1**. The first approach (RC-SIFT) aims to improve the SIFT keypoint detector and descriptor by preserving its invariant and robustness properties and reduce the complexity of matching process. The second analyzes the effect of the various SIFT feature properties on the performance of near-duplicate image retrieval. Accordingly, we do not focus on producing high performance instead, we are interested in reducing the complexity of keypoints matching, preserving their invariance and robustness and understanding their properties.*

5 Approaches for Local Features Adaptation

Finding similar images that show the same scene but have been taken with slightly different conditions (i.e. near-duplicate images) is still a very challenging task, even though it is a very fundamental problem in many real-world tasks. Figure 35 presents that the first step in ND-image retrieval system is to extract the features of an image. This step constructs an abstracted representation of images to simplify and accelerate the solving of content-based image retrieval.

Keypoint features are used to detect near-duplicate images due to their invariant and robustness to different kinds of altering. The scale invariant feature transformation algorithm (SIFT) has been designed to detect and characterize local features in images. It is widely applied to find similar regions in affine transformed images, to recognize similar objects, or to retrieve near-duplicates of images since it outperforms the other kinds of keypoint detectors and descriptors as clarified in Section 3.2. In Appendix B, we present the comparison results of the SIFT, SURF, ORB, BRIEF and BRISK algorithms employing various threshold for each. We applied this comparison to the images of the visual geometry group [5]. The comparison results in Appendix B presents that the SIFT algorithm almost outperforms the other competing methods. Therefore, our focusing is to improve the SIFT algorithm for solving the ND-retrieval tasks.

Due to the computational complexity of SIFT based matching operations, several approaches have been proposed in the literature to speed up this process, such as the PCA−SIFT [99] and reduced SIFT$-96D$, $-64D$ and $-32D$ [102], described in Section 3.2. However, most of these approaches lack significant decrease in matching accuracy compared to the original descriptor. We propose an approach, that is optimized for near-duplicate image retrieval tasks by a dimensionality reduction

process that differs from other methods by preserving the information around the keypoints of any region patches of the original descriptor. We called this approach *Region Compressed SIFT* and we introduced it in our publications [13] and [15] to answer **RQ.1**(a) and (b). The computation of the proposed Region Compressed (RC) SIFT$-64D$ descriptors is, therefore, faster and requires less memory for indexing. Most important, the obtained features show at the same time a better retrieval performance and seem to be even more robust. In order to prove this, we provide results of a comparative performance analysis using the original SIFT$-128D$, SIFT$-64D$ [102], SURF$-64D$ [27] and our compressed RC-SIFT versions, in image near-duplicate retrieval using large scale image benchmark databases.

In further discussion in this chapter, we present our study published in [14] to discuss **RQ.1**(c). We tackle two issues: First, especially in ND-image retrieval field the employing of the scale and contrast properties to select a set of keypoints of SIFT$-128D$ or RC-SIFT$-64D$ speeds up the process of image matching, by decreasing the memory and time complexity of the indexing and matching process. Second, the involvement of weights computed based on the scale, contrast, or orientation properties improves the robustness and accuracy of the matching process of the SIFT and RC-SIFT. We evaluate the performance of the proposed hypothesis by conducting extensive experiments using established benchmarks. The experiments show that using of feature properties improves the performance of SIFT and RC-SIFT$-64D$ in solving the ND-image retrieval tasks.

5.1 Region Compressed SIFT Descriptor for NDR

To motivate and describe our suggested modifications of the SIFT descriptor, we will firstly explain briefly the working mechanism of the SIFT detector and descriptor [116].

5.1.1 SIFT$-128D$ Descriptor

The detection of SIFT features can be achieved in four main stages specified as follow: scale-space extrema detection, keypoint localization, orientation computation, and keypoint descriptor computation.

In the first stage, image scale space is built by down-sampling and blurring the input image several times. The blurring is accomplished by convolving the input image with multiple-scale Gaussian filters. After that, the neighbor layers in the scale space are subtracted from each other to form the difference of Gaussian (DoG) images. In the second stage, SIFT keypoints are determined by finding the local maxima and minima in DoG images. The stability of keypoints is verified against contrast change and edge response and the unstable keypoints are rejected. In the third stage, the dominant orientation is determined and assigned to each keypoint.

In the final stage, a highly distinctive descriptor is computed at each keypoint. The SIFT descriptor is extracted from the region around a keypoint which is called

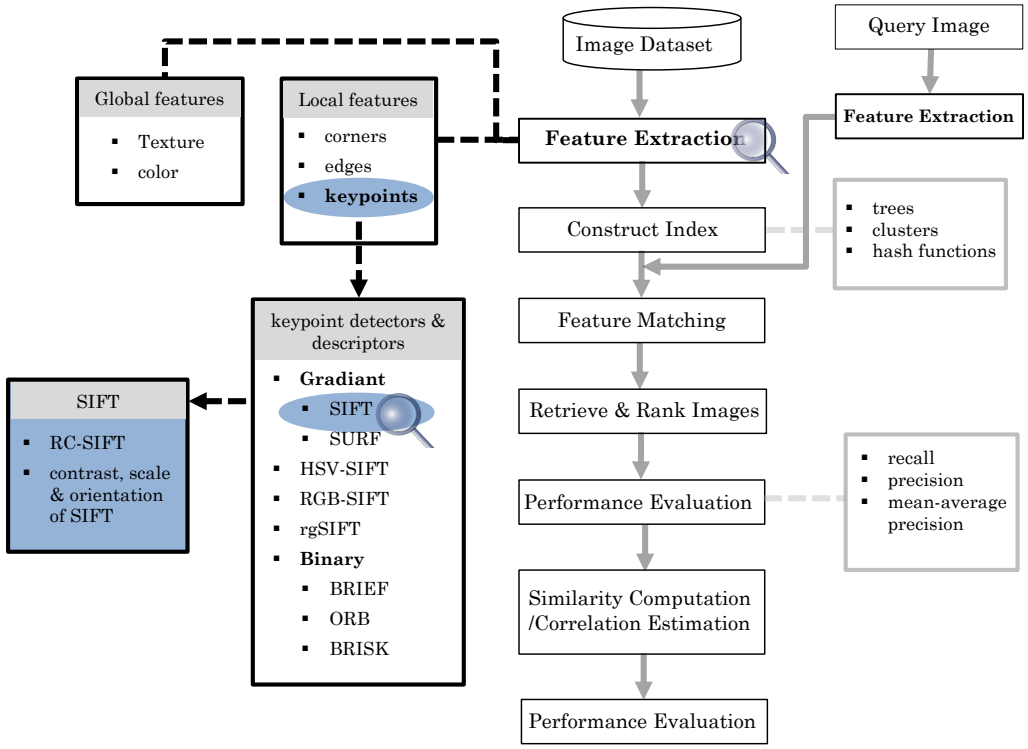


Figure 35: Flowchart of image retrieval systems with specific focusing on the feature extraction step.

region of interest (RoI). The RoI is rotated around a keypoint utilizing the dominant orientation. Afterwards, a $n \times n$ orientation histogram is created over the RoI. For each bin of the histogram, r orientations are assigned, so that the descriptor has three dimensions and $n \times n \times r$ elements. The size of the SIFT descriptor is controlled by the width of the orientation histogram n and the number of orientation bins r. In the original SIFT algorithm [116], it has been shown that the best matching results are reported when $n = 4$ and $r = 8$, i.e. when a descriptor of $4 \times 4 \times 8 = 128$ element is constructed. Figure 36(a) presents the way in which the SIFT$-128D$ descriptor is constructed.

However, the high dimensionality of SIFT descriptor ($128D$) increases the sparsity of descriptors and this may affect the accuracy of descriptor indexing in image ND (for a discussion of problems related to high dimensional data indexing and clustering, see e.g. [3]). Therefore, in this chapter, we compress the dimensionality of the SIFT descriptor. In the next subsection, we explain our approach to compress SIFT descriptor.

5.1.2 Region Compressed SIFT Descriptors (RC-SIFT)

To increase the efficiency of descriptor indexing, i.e. speeding up the process and reducing the amount of stored data in ND-retrieval, we propose a method to compress

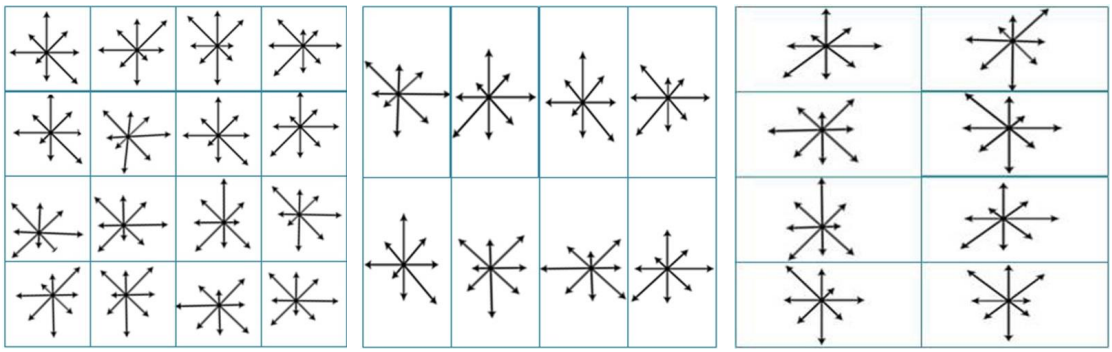
the dimensionality of the SIFT descriptor from $128D$ to $64D$. We achieve this by first extracting the SIFT features in the same way as in the original SIFT algorithm [116] (as described in Subsection 5.1.1). Afterwards, descriptors are computed over all pixels in RoI with specific locations employing their gradients and orientations with respect to the corresponding keypoints. Each descriptor is computed as a $3D$ histogram centered at the keypoint. In the original SIFT algorithm, this descriptor has the dimensions $4 \times 4 \times 8$. The values of these three dimensions indicate how the keypoint shifts to each allowed position in RoI in vertical and horizontal locations that is 4×4 locations. For each location, 8 directions are allowed between 0^o and 360^o. In contrast to the reduction method presented in [102] through ignoring some patches of RoIs, we suggest in this chapter that for each two possible horizontal shiftings in the same direction with respect to the keypoint, only one vertical shifting is allowed. Subsequently, for all possible horizontal shiftings (i.e. four horizontal shiftings) in all directions, only two vertical shiftings exist. For each of these (4×2) locations, eight directions are computed. In this way, we reduce the number of possible changes in the SIFT descriptor when the RoI is modified. Moreover, the number of altered bins in the RoI histogram decreases. As a result we obtain $4 \times 2 \times 8$ histogram i.e. $64D$ SIFT descriptor. We call our method for extracting and compressing the SIFT descriptor "Region Compressed SIFT" (RC-SIFT). The histogram at each keypoint is presented as a triplet of elements H_y, H_x and H_θ where:

$$H_y = y - \frac{N_y - 1}{2} \ , \qquad H_x = x - \frac{N_x - 1}{2} \ , \qquad H_\theta = \frac{2\pi}{N_\theta} \tag{48}$$

where N_y and N_x define the number of bins in H_y and H_x, respectively. The values of y and x are defined as $y = 0, ..., N_y - 1$, and $x = 0, ..., N_x - 1$. N_θ is the number of orientations in each bin of the histogram and θ is defined as $\theta = 0, ..., N_\theta - 1$.

In original SIFT $N_y = 4$, $N_x = 4$ and $N_\theta = 8$ whereas our suggestion is to set $N_y = 2$, $N_x = 4$ and $N_\theta = 8$ to get the descriptor of the form $4 \times 2 \times 8$ or to set $N_y = 4$, $N_x = 2$ and $N_\theta = 8$ to get the descriptor of the form $2 \times 4 \times 8$. We refer to these descriptors as RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) respectively. Figure 36(b) and (c) present the details of building RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) respectively. We applied the experiments using RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) in addition to RC-SIFT$-32D$ (obtained by employing $N_y = 2$, $N_x = 2$ and $N_\theta = 8$) and finally RC-SIFT$-16D$ (produced by suggesting $N_y = 2$, $N_x = 2$ and $N_\theta = 4$). The goal of building RC-SIFT$-32D$ and RC-SIFT$-16D$ is to check whether the performance of our RC-SIFT still stable when the descriptor is compressed to $32D$ or $16D$.

In this way, the RC-SIFT descriptor preserves the size of RoI around a keypoint i.e. contrary to the method described in [102], no region around the keypoint is ignored. In the next step, we evaluated the efficiency of RC-SIFT$-64D$, RC-SIFT$-32D$ and RC-SIFT$-16D$ against the performance of the original SIFT$-128D$, SURF$-64D$ and SIFT$-64D$ suggested in [102] for image near-duplicate retrieval.



(a) Original SIFT 128 D (b) RC-SIFT-64 D (R) (c) RC-SIFT-64 D (C)

Figure 36: Comparison between the gradient orientation descriptors of SIFT$-128D$ and RC-SIFT$-64D$. (a) $4 \times 4 \times 8$ array of SIFT$-128D$. (b) 2×4 array of RC-SIFT$-64D$(R). (c) $4 \times 2 \times 8$ array of RC-SIFT$-64D$(C).

5.1.3 SIFT Descriptors Indexing with a Vocabulary Tree

The straightforward way to match SIFT features is an exhaustive search, which can be achieved by matching each feature of a given query image with all features in the feature database. However, the exhaustive search of SIFT features is extremely time-consuming specifically, for large-scale benchmarks, which produce huge amounts of features. To overcome this problem, hashing functions [23], [47], direct clustering [46], [193] and hierarchical clustering [90], [136], described in Section 2.4, have been adapted to quantize and index SIFT descriptors. In our study, a *vocabulary tree* and inverted files were used as described in [90], [136] to index SIFT descriptors. The construction of the vocabulary tree, described in Section 2.4, is accomplished by splitting the descriptors into $k-$ clusters, employing the $k-$means algorithm. Each cluster is iteratively segmented into k clusters to build a vocabulary tree of depth L and k^L leaf clusters. The clusters in the tree form the set of visual words [90] and they are employed to represent SIFT and RC-SIFT descriptors. The identification keys of the images, that have at least one descriptor belong to a specific leaf cluster, are stored in an inverted file.

In [136] the $L1-$norm and $L2-$norm have been used to compute the similarity between images. The $L1-$norm tends to give better matching results [136]. In our evaluation, we use both the $L1-$norm and $L2-$norm in order to measure the similarity between normalized query and database vectors by traversing each vector in a vocabulary tree as described in Equation (49) [136].

5.1.3.1 Complexity of the Vocabulary Tree for SIFT$-64D$ and SIFT$-128D$

After building the vocabulary tree (see Subsection 5.1.3), the tree was used for image matching. A D dimensional descriptor vocabulary tree of depth L and k^L leaf nodes need a memory of $O(Dk^L)$. Specifically, a $128D$ descriptor tree requires $O(128k^L)$ whereas, a $64D$ descriptor tree requires only $O(64k^L)$. The time complexity of build-

ing a vocabulary tree is affected by the dimensionality of descriptors. Considering the total number of nodes in the vocabulary tree is $\Sigma_L^{i=1} k^i = \frac{k^{L+1}-k}{k-1} \approx k^L$ (see also [136]), the time complexity of building the vocabulary tree for a $D-$ dimensional descriptor database is $O(DNTk^L)$, where T is the iterations and N is the number of all descriptors of a given image database. Based on this, the time complexity of building a tree for a descriptor database of dimensionality $D = 128$ is $O(128NTk^L)$, whereas the time complexity for the RC-SIFT$-64D$ descriptors is $O(64NTk^L)$ (i.e. just a linear decrease). So that, the introduced RC-SIFT$-64D$ descriptors obviously speed up the indexing process and reduce the required memory for processing. To justify this hypothesis, we employed the $UKB10$ dataset described in Subsection 5.1.5 to measure the time required by SIFT$-128D$, reduced SIFT$-64D$, and our RC-SIFT$-64D$ to build the vocabulary trees. Table 3 presents that the indexing time needed by RC-SIFT$-64D$ and SIFT$-64D$ [102] is about the halve time required by SIFT$-128D$. The presented results were computed employing vocabulary trees of depth $L = 6$, 4 and 2 and initial centers $k = 10$. To justify the robustness of these results, we repeated the experiment applying two descriptor datasets of different sizes. The results present that the indexing of SIFT$-64D$ descriptor is faster than our RC-SIFT$-64D$. The reason is that the complexity of SIFT$-64D$ descriptor is lower than RC-SIFT$-64D$ since it produced through the ignoring of some patches around keypoints whereas, the descriptors of RC-SIFT$-64D$ are produced by the compressing of those patches.

5.1.4 Evaluation Measures

We verify the performance of the RC-SIFT$-64D$, RC-SIFT$-32D$ and RC-SIFT$-16D$ against the SIFT$-128D$, the SIFT$-64D$ [102] and the SURF$-64D$ in the image near-duplicate retrieval field. The performance is measured on a large-scale image databases using the vocabulary tree for feature indexing and $L1-$norm to compute the similarity between images. The vocabulary trees are constructed as described in Subsection 5.1.3 for each type of the employed descriptors separately. In our experiment the initial number of clusters is set to $k = 10$.

To perform the ND-retrieval task, the similarity between normalized query vectors q_img and database vectors d_img is computed by traversing each vector in the vocabulary tree and it is given as [136]:

$$s(q\_img, d\_img) = \left\| \frac{q\_img}{\|q\_img\|} - \frac{d\_img}{\|d\_img\|} \right\| \tag{49}$$

The normalization can be done in any desired norm. In our experiment $L1-$norm and $L2-$norm, that are presented in Subsection 2.5.1, were implemented.

In this work, we used our own implementation of SIFT algorithm using some "Opencv" functions. SURF descriptors were computed by means of Opencv functions. The vocabulary tree was constructed using Matlab functions and VLFeat

Table 3: The computation time needed to perform the indexing for SIFT$-128D$, RC-SIFT$-64D$ and SIFT-$64D$ [102] using a standard processor(Intel(R) Core(TM) i7-8700 CPU) and a Matlab implementation.

Leaves of tree	Descriptor datasets	Method	Time(sec)
10^6	$2,789,994$	SIFT-$128D$	2763.26
		SIFT-$64D$	1480.71
		RC-SIFT-$64D$	1520.48
	$2,095,545$	SIFT-$128D$	2033.58
		SIFT-$64D$	1036.75
		RC-SIFT-$64D$	1103.27
10^4	$2,789,994$	SIFT-$128D$	2356.15
		SIFT-$64D$	1230.59
		RC-SIFT-$64D$	1420.10
	$2,095,545$	SIFT-$128D$	1986.43
		SIFT-$64D$	969.09
		RC-SIFT-$64D$	1017.58
10^2	$2,789,994$	SIFT-$128D$	1132.2
		SIFT-$64D$	590.98
		RC-SIFT-$64D$	647.72
	$2,095,545$	SIFT-$128D$	936.59
		SIFT-$64D$	482.14
		RC-SIFT-$64D$	560.59

library. Moreover, we implemented the SIFT$-64D$ described in [102] based on our implementation of SIFT algorithm by ignoring some patches of descriptors as it is described in [102].

The results of the experiments were evaluated by computing the *recall* value. Since we always have a fixed number of relevant images and the comparison is done using a ranked list of fixed lengths (i.e. length of one, three, or ten images as indicated in Tables 4 and 5, respective), we ignored precision since it is directly correlated to the recall values. Rather we computed the values of mean recall MR, mean average precision MAP and variation of recall VR as reported in Section 2.5.2.

5.1.5 Image Datasets

We present an extensive benchmark study to verify the performance of the RC-SIFT$-64D$ in solving near-duplicate retrieval tasks in the following scenarios:

- Various benchmarks: We checked the performance using image databases of various resolutions. We applied our experiments on UKbench [135] [136] and Caltech-Buildings benchmark [11], [10](see Subsection 5.1.6).

- Benchmarks of various sizes: We verified the performance employing benchmarks of various sizes produced from the UKbench benchmark.

- Descriptor databases of various sizes: We evaluated the performance for a variable number of extracted features.

- Benchmark of transformed images: We convolved various kinds of transformation with a set of images (i.e. rotation, blur, noise, illumination change, scale change) to evaluate the robustness of our RC-SIFT$-64D$ to image altering. In addition, we justified the robustness of the RC-SIFT descriptors to the combination of transformation i.e. illumination and rotation changes, illumination change and adding noise, and combination of adding noise and rotation.

The aim of building these datasets is to justify the robustness of our RC-SIFT features to different types of datasets and modifications.

5.1.5.1 Benchmarks of various sizes To build datasets of various sizes, we employed the UKbench dataset described in Section 4.2 [135], [136]. This database contains four different images of $2,550$ different indoor/outdoor scenes i.e. $10,200$ images in total of resolution 640 in width or height. The UKbench dataset contains images with complex alters for some scenes (i.e. different arrangement of objects, appear/disappear of some objects in addition to changes in lightness, contrast, sharpness, scale, and viewpoint conditions). From this benchmark, we constructed four image datasets of different sizes to test the robustness of RC-SIFT descriptors in solving the task of image near-duplicate retrieval. For the experiment, we selected the first image of each scene as a query image, while the remaining three images were used as a basic database for the retrieval task. The constructed benchmarks have the sizes 10200, 6000, 4000 and 2000 images and they are referred as $UKB10$, $UKB6$, $UKB4$ and $UKB2$, respectively.

5.1.5.2 Descriptor Databases of Various Sizes To generate descriptor datasets of various sizes, we used Caltech-Buildings benchmark presented in 4.3. The resolution of images in this benchmark is very high compared to the UKbench benchmark (i.e. they have the resolution 2048 in width or height). Therefore, we used them to generate descriptor datasets of various sizes. The Caltech-Buildings benchmark contains 250 images of 50 different scenes. We picked the first one of them as a query image and the rest four are considered as images in the dataset. We determined three different thresholds to extract 2500, 1000 and 500 SIFT keypoints of images and to build $CB2500$, $CB1000$ and $CB500$ descriptor datasets respectively [11], [10].

5.1.5.3 Benchmark of Transformed Images To verify the robustness of our RC-SIFT against different kind of image transformations in the field of ND-image retrieval, the performance of all proposed descriptors was evaluated against rotation change, illumination increase or decrease, and adding different kinds of noise. Moreover, it was evaluated to the combinations of transformations. To achieve this, we picked the first 500 images of different scenes of the UKbench dataset (presented in 4.2) and named them $UKbench - T$. After that, we applied on the $UKbench - T$ one or more kinds of transformations. In the evaluation process, the original images were used as query images and the transformed ones as the datasets. The settings of transformations are explained later in the results.

5.1.6 Result and Analysis

We extracted keypoints from images using original SIFT$-128D$ [116], SURF$-64D$ [27], SIFT$-64D$ [102], and our RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) [13]. After that, the descriptors of each kind were indexed separately utilizing vocabulary trees of depth $L = 4$ and $k = 10$ initial clusters. To complete the retrieval task, the distances between a query and database images were computed as given in Equation (49) using the $L1-$norm and $L2-$norm. However, in our experiment, $L1-$norm obtains better results than the $L2-$norm. Therefore, we present the results obtained by the $L1-$norm. The retrieval process is successfully completed if the corresponding images with a specific query appear on the top of retrieved images.

5.1.6.1 UKbench Benchmarks To compare the results utilizing benchmarks of various sizes, we employed $UKB10$, $UKB6$, $UKB4$ and $UKB2$ that are described in Paragraph 5.1.5.1. In these datasets, there are three images relevant to each query image. We evaluated the results considering that the system retrieves three, ten, and fifty images. Table 4 summarizes the results of all proposed descriptors using the $UKB10$, $UKB6$, $UKB4$ and $UKB2$ datasets. In this table, a query image is retrieved if its relevant images in the benchmark appear on the top three retrieved images. This table shows that the RC-SIFT$-64D$ obtained slightly better results than SIFT$-128D$. The values of variance are small for all descriptors but the smallest values are found for SURF-$64D$ and SIFT-$64D$ [102]. The best mean average precision is found by the RC-SIFT$-64D$ and then by the SIFT$-128D$ algorithm. Tables 5 and 6 present the performance of the various descriptors when the relevant images appear on the top ten or fifty results, respectively. In both cases, the best performance is given by RC-SIFT$-64D$ and SIFT$-128D$. The results explain that the performance of all methods increases when the size of image datasets decreases i.e. the best performance is found for the $UKB2$ dataset.

The results presented in Tables 4, 5 and 6 show that, if the mean recall increase the variance values increase for both SURF-$64D$ and SIFT-$64D$ [102]. Whereas, for both of RC-SIFT$-64D$ and SIFT$-128D$ the variance of recall decrease as the mean recall value increases. Figure 37 provides a qualitative comparison between

Table 4: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT-64D using benchmarks of various sizes $UKB10$, $UKB6$, $UKB4$ and $UKB2$, each of them contains images of various scenes with groups of four images belong to the same scene. The first image of each scene was used as a query image. The mean recall MR, the variance of recall VR and mean average precision MAP were computed in percent based on the ***top three retrieved images***. The symbols RC-SIFT−64D(R) and RC-SIFT−64D(C) are used to refer to the compression of forms $4 \times 2 \times 8$ and $2 \times 4 \times 8$, respectively.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
UKB10	MR	49.3	27.2	24.3	**50.7**	49.9
	VR	15.1	**11.2**	13.2	14.8	14.6
	MAP	47.5	25.2	22.9	**48.8**	47.9
UKB6	MR	55.3	29.9	26.3	**57.1**	56.3
	VR	14.4	**11.5**	12.3	13.7	14.0
	MAP	53.5	27.9	24.6	**55.2**	54.1
UKB4	MR	53.1	27.1	25.0	**54.5**	53.1
	VR	14.3	**10.9**	11.1	13.6	13.7
	MAP	51.3	25.2	23.4	**52.7**	51.7
UKB2	MR	51.6	25.6	26.1	**54.9**	51.8
	VR	13.4	**10.0**	11.2	12.5	12.8
	MAP	49.7	24.0	25.5	**53.1**	49.7

all proposed descriptors. For this example it shows that the best results are found when the RC-SIFT−64D is used. However, there are of course other examples where the SIFT−128D performs best. Moreover, we note in many cases that despite the equivalent recall results of SIFT−128D and RC-SIFT−64D descriptors, the RC-SIFT−64D obtains better mean average precision values than the SIFT−128D descriptor. Figure 38 presents an example of the results where the performance of SIFT−128D and RC-SIFT−64D is equivalent but the ranking of the results found by RC-SIFT−64D is better than SIFT−128D.

The compression of the dimensionality RC-SIFT to $-32D$ or $-16D$ reduces its mean recall to 36.64% and 24.60% respectively for the $UKB10$ considering the top three results and 45.54% and 28.78% for the top ten retrieved images. However, RC-SIFT−32D still show robust performance in case of viewpoint change. Since RC-SIFT−32D and RC-SIFT−16D perform lower than than the original SIFT−128D and RC-SIFT−64D, we resumed our experiments and analyze without further comparison with RC-SIFT−32D and RC-SIFT−16D.

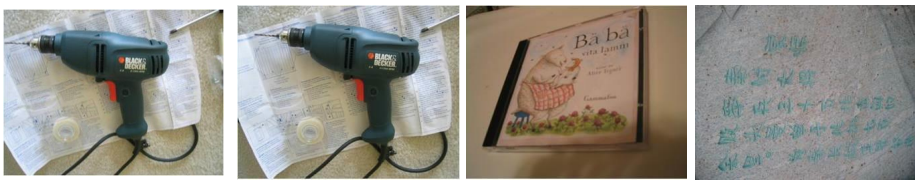
5.1.6.2 Caltech-Buildings Benchmark In this case, we utilized the $CB-2500$, $CB-1000$ and $CB-500$ described in Paragraph 5.1.5.2 to justify the robustness of

Table 5: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D employing benchmarks of various sizes ($UKB10$, $UKB6$, $UKB4$ and $UKB2$). The evaluation was completed based on the ***top ten retrieved images***.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
UKB10	MR	58.7	36.2	30.2	**60.7**	59.2
	VR	15.2	14.0	14.7	14.8	14.9
	MAP	50.1	28.2	23.3	**52.7**	50.6
UKB6	MR	64.8	39.0	34.2	**67.1**	65.1
	VR	13.7	14.3	14.6	13.1	**13.5**
	MAP	57.3	31.2	28.3	**59.4**	58.0
UKB4	MR	62.3	35.4	31.9	**64.6**	62.0
	VR	14.1	13.5	13.7	**13.4**	13.6
	MAP	54.9	28.1	24.2	**57.0**	54.5
UKB2	MR	61.0	30.1	33.7	**64.9**	61.4
	VR	13.3	**12.3**	12.9	12.7	12.8
	MAP	53.3	26.6	28.2	**57.4**	53.5

Table 6: The retrieval performance of the SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT-64D using the benchmarks $UKB10$, $UKB6$, $UKB4$ and $UKB2$. The MR, VR and MAP were computed based on the ***top fifty retrieved images***.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
UKB10	MR	69.4	49.1	45.1	**72.2**	70.2
	VR	13.0	15.1	15.6	**11.8**	13.0
	MAP	51.2	29.4	25.7	**53.9**	52.1
UKB6	MR	75.0	52.0	50.8	**77.6**	75.6
	VR	11.1	14.8	14.9	**9.8**	10.9
	MAP	58.4	32.3	30.0	**60.6**	59.0
UKB4	MR	73.0	47.9	47.0	**75.5**	73.1
	VR	11.5	14.9	15.0	**10.3**	11.3
	MAP	56.0	29.2	26.8	**58.1**	56.2
UKB2	MR	72.4	46.3	47.2	**76.1**	72.7
	VR	11.5	14.1	14.0	**9.6**	11.0
	MAP	54.5	28.0	26.9	**58.6**	54.9



Top three retrieved images by **SIFT-128D**

Top three retrieved images by **SIFT-64D**

Top three retrieved images by **SURF-64D**

Top three retrieved images by ***RC-SIFT-64D***

Figure 37: Performance comparison between SIFT$-128D$, SIFT$-64D$, SURF$-64D$ and RC-SIFT-$64D$ in solving the image near-duplicate retrieval task. The results present that RC-SIFT-$64D$ shows the best performance.

the various descriptors to the amount of the extracted keypoints. In our experiment, we built a vocabulary tree of depth $L = 3$ and initial clusters $k = 10$ for each of the $CB - 2500$, $CB - 1000$ and $CB - 500$ datasets. Table 7 presents the results of all descriptors when the relevant images appear in the top four results. It shows a comparable performance of the RC-SIFT-64 and the SIFT-128 algorithms. However, Tables 8 and 9 present a slight enhancement in the performance of the RC-SIFT-64 compared to the SIFT-128 descriptor. Moreover, by comparing the results in Tables 4, 5, 6 and 8, 9 we find that the performance of the SIFT-64 and the SURF-64 algorithms for the Caltech-Buildings benchmark is better than their performance for the benchmarks constructed based on the UKbench benchmark. Tables 7, 8 and 9 present that the best performance is found for the descriptor dataset $CB - 500$ i.e. for the least amount of features. This concludes that the false feature matches increase by increasing the amount of extracted features, which in role increase the amount of non-relevant images that appear on the top of the retrieved results.

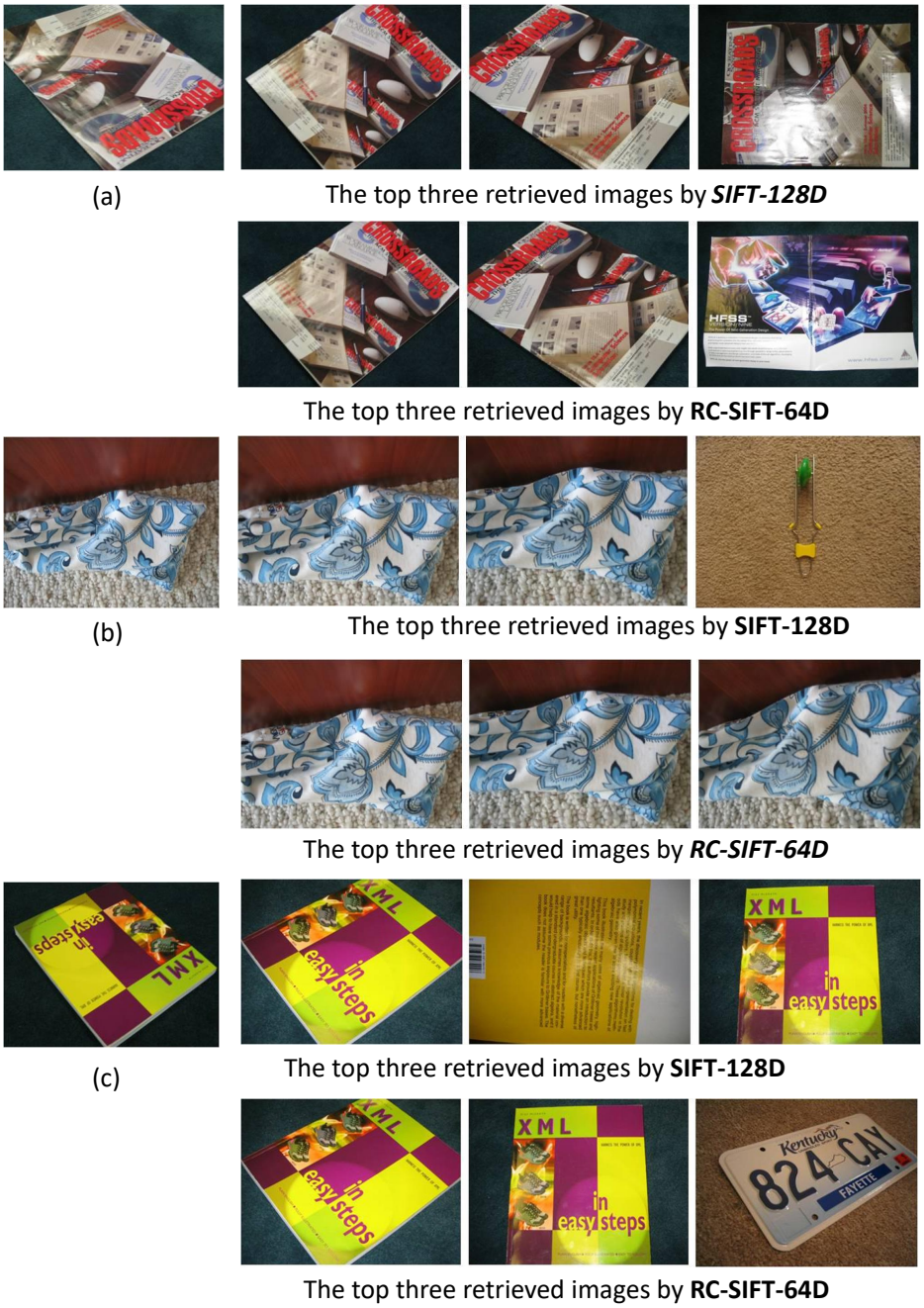


Figure 38: Performance comparison between the original SIFT$-128D$ and RC-SIFT$-64D$ in solving the image near-duplicate retrieval task. (a) presents better retrieval ranking by the original SIFT than the RC-SIFT$-64D$. (b) displays superior performance by our RC-SIFT$-64D$. (c) shows equivalent performance of both SIFT$-128D$ and SIFT descriptors. In this example RC-SIFT$-64D$ presents better raking of the results than SIFT$-128D$.

Table 7: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT-64D when various amount of features were extracted from the Caltech-Buildings images (i.e. 500, 1000 and 2500 features for each image). A query image is retrieved when one or more of its relevant images is obtained in the ***top four results***.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
CB500	MR	**44.0**	39.5	33.2	**44.0**	**44.0**
	VR	8.2	8.3	**7.5**	7.8	7.8
	MAP	39.9	35.4	29.3	**40.3**	40.1
CB1000	MR	43.0	38.2	31.7	42.8	**43.3**
	VR	6.5	6.5	6.7	7.2	7.1
	MAP	40.2	34.1	28.1	39.8	**40.4**
CB2500	MR	**39.5**	36.7	29.8	39.0	39.0
	VR	8.2	7.7	7.2	8.4	8.5
	MAP	**36.7**	31.9	26.1	**36.4**	**36.4**

Table 8: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D and RC-SIFT-64D when the $CB-2500$, $CB-1000$ and $CB-500$ benchmarks are used. A query image is retrieved if one or more of its related images is obtained in the ***top ten results***.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
CB500	MR	57.0	51.0	39.7	**58.2**	57.7
	VR	11.0	13.0	12.8	**10.6**	10.8
	MAP	44.9	39.2	33.0	**45.6**	45.2
CB1000	MR	**53.0**	47.3	34.2	**53.0**	52.8
	VR	11.4	14.1	14.5	**10.4**	10.6
	MAP	43.6	39.1	26.7	**43.8**	43.6
CB2500	MR	48.5	41.7	33.8	**49.0**	**49.0**
	VR	12.6	13.2	12.6	12.3	12.6
	MAP	40.3	36.2	23.8	**40.6**	40.3

5.1.7 Image Transformations

To verify the robustness of RC-SIFT against different types of image transformations in the field of NDimage retrieval, the performance of all proposed descriptors was evaluated against various transformations, described in Paragraph 5.1.5.3, utilizing the $UKbench-T$ dataset. The settings of generating the transformed images are similar to the setting applied in [102]. The descriptors are indexed employing a vocabulary tree of depth $L=3$ and initial centers $k=10$. The similarity is computed

Table 9: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D and RC-SIFT−64D when $CB-2500$, $CB-1000$ and $CB-500$ benchmarks are employed. The performance was verified on the top ***fifty retrieved images***.

Dataset	Results	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
$CB500$	MR	74.0	67.0	50.4	75.0	**75.3**
	VR	8.2	14.6	14.9	**7.0**	**7.0**
	MAP	47.4	42.7	40.3	47.6	**48.0**
$CB1000$	MR	71.5	59.8	48.9	**73.5**	73.2
	VR	9.3	14.3	14.0	8.9	**8.7**
	MAP	45.3	43.5	35.3	**46.0**	44.7
$CB2500$	MR	66.5	53.0	48.5	**67.0**	**67.0**
	VR	**10.4**	14.5	14.9	10.6	11.0
	MAP	43.0	37.0	25.6	**43.1**	43.0

using $L1-$norm. A query image is considered to be found in the database if its corresponding database image appears on the top of the retrieved images. Hence, in this case the mean average precision MAP and the mean recall MR and they present the performance in this case.

5.1.7.1 Rotation Change To verify the rotation invariance of the RC-SIFT−64D, the images of the $UKbench-T$ dataset were rotated by different angles in a clockwise direction to generate 500 database images for each angle. The employed angles are 40^o, 135^o, 215^o and 250^o. Table 10 shows that all proposed descriptors (i.e. SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D) are rotation invariant. For a big rotation change the results clarify that RC-SIFT−64D perform a little bit better than the other proposed descriptors. Since there is only one relevant image in the dataset and we consider only the retrieved result with the highest rank, the mean average precision MAP and the mean recall MR values are equal and present the performance in Table 10.

5.1.7.2 Addition of Noise To test the invariance of the RC-SIFT−64D to adding noise, three types of noise were applied to the $UKbench-T$ benchmark. These are Gaussian noise, salt and pepper noise, and multiplicative noise. The noise was added to images using the following settings: Gaussian white noise (GN) with $\sigma^2 = 0.1$ and $\sigma^2 = 0.2$, salt and pepper noise (SPN) with density of 15% and 35% and multiplicative noise (MPN) with mean 0 and $\sigma^2 = 0.04$. The performances of all used methods (computed as MAP or MR) in this work are presented in Table 11. These results show that performance all proposed descriptors decrease very strongly when the ratio of noise increase. However, in case of using the salt and pepper noise

Table 10: Performance comparison of the SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D(R) and RC-SIFT−64D(C) using the rotated images of the $UKbench-T$ benchmark and the $UKbench-T$ as query images. For each query image we checked if its corresponding database image appears as the first retrieved image in the result. The experiment was repeated for the rotation values: $\{40^o, 135^o, 215^o, 250^o\}$. *MAP and MR are equivalent in this case and the present the performance.*

Rotation	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
$\Theta = 40^o$	**93.41**	92.80	93.30	93.13	92.82
$\Theta = 135^o$	**92.65**	92.41	**92.60**	92.44	92.46
$\Theta = 215^o$	**93.40**	92.68	92.60	93.10	92.82
$\Theta = 250^o$	91.82	**92.80**	92.01	92.43	92.35

the RC-SIFT−64D obtains better results than the other descriptors even when the ratio of noise increases.

Table 11: The performance of SIFT−128D, SIFT−64D, SURF−64D and RC-SIFT−64D using $UKbench-T$ benchmark as ground truth and a set of 500 query noised images, produced employing either GN, SPN or MPN filters. Experiments were repeated for various amounts of noise. For each query image we checked if its corresponding database image appear as the first retrieved image in the result. The performance is presented by the *MAP or MR which have equal values in this case.*

Noise	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
GN $\sigma^2 = 0.1$	**68.41**	64.82	64.41	68.26	67.98
GN $\sigma^2 = 0.2$	**35.60**	29.6	35.22	35.24	35.40
SPN 15%	82.63	82.29	81.22	**83.45**	83.12
SPN 35%	20.22	15.24	14.51	**20.88**	20.50
MPN 0.04	**98.00**	82.20	80.11	97.23	97.05

5.1.7.3 Illumination Change The illumination invariance is verified in cases of increase and decrease the brightness of the $UKbench-T$ benchmark. This is done by adding or subtracting specific values of all pixel's channels (i.e. the channels red, green, and blue of each pixel are incremented equally). The values of color channels are adjusted to be within the range $\{0-255\}$. The brightness effect is tested using the values $\{50, 70, 100, 120\}$ and the darkness effect is test using the values $\{-30, -50, -70, -90\}$. Results of ND-retrieval tasks, summarized in Tables 12 and 13, report that all kinds of used descriptors perform well to illumination increase and decrease.

5.1.7.4 Image Blurring To check the robustness of the compared descriptors against adding blur, three blurred image databases are generated using the

Table 12: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D, RC-SIFT−64D(R), and RC-SIFT−64D(C) using the images of the $UKbench - T$ benchmark as queries, each of them has one brightened image in the database. For each query image we checked if its corresponding database image appear as the first retrieved image in the result. The performance (calculated as MAP or MR) was checked for the following brightness values: $\{50, 70, 100, 120\}$. *Ill-Inc* refer to illumination increase.

Brightness	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Ill-Inc 50	**100**	99.85	99.50	99.50	97.00
Ill-Inc 70	**100**	99.85	99.30	99.50	97.00
Ill-Inc 100	96.80	94.21	93.22	95.61	95.61
Ill-Inc 120	91.20	**90.23**	88.16	**90.23**	**90.23**

Table 13: The retrieval performance of SIFT−128D, SIFT−64D, SURF−64D, RC-SIFT−64D(R) and RC-SIFT−64D(C) using the $UKbench - T$ benchmark as query images, each of them has one darkened image in the database. The performance is presented using MAP or MR and the darkness values: $\{-30, -50, -70, -90\}$. For each query image we checked if its corresponding database image appears as the first retrieved image in the result. *Il-Dec* is a shortcut of illumination decrease.

Darkness	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Il-Dec 30	**100**	99.80	99.71	**100**	99.71
Il-Dec 50	**99.22**	**99.22**	99.20	99.09	98.88
Il-Dec 70	**95.81**	95.60	94.75	95.55	95.55
Il-Dec 90	81.03	80.26	80.20	**82.25**	**82.25**

$UKbench - T$ benchmark and employing three different values of Gaussian filter i.e. $\sigma^2 = 5$, $\sigma^2 = 10$ and $\sigma^2 = 20$. Table 14 shows that the performance (i.e. MAP and MR) degrades most clearly when the ratio of blurring increases (i.e. when the value of σ^2 increases). However, for small amount of blurring, the descriptors seem to be invariant. By increasing the amount of blurring, RC-SIFT−64D is superior in matching images.

Table 14: Comparison of retrieval performance of SIFT−128D, SIFT−64D, SURF−64D, RC-SIFT−64D(R) and RC-SIFT−64D(C) using the $UKbench - T$ benchmark as query images, each of them has one blurred image in the database. For each query image exists only one relevant image, i.e. MAP and MR are equivalent and define the performance. The experiment was repeated for different level of blurring using $\sigma = 5$, $\sigma = 10$ and $\sigma = 20$.

Blur	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
$\sigma = 5$	**92.20**	83.63	85.90	83.05	83.47
$\sigma = 10$	**42.60**	36.80	34.71	36.65	36.82
$\sigma = 20$	35.84	33.02	29.84	38.60	**38.81**

5.1.7.5 Scale Change The robustness of all proposed descriptors was verified against scaling change by selecting 500 different scenes of the benchmark database [136] for which there are two images taken at different scales. Some of the selected images have additional viewpoint change as well. The first image of each scene was used as a query and the second as a database image therefore, MAP and MR are identical. The results of SIFT$-128D$, SIFT$-64D$, SURF$-64D$, RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) are **80.1**, 76.3, 53.3, **80.1** and **80.1** respectively. These results show that both SIFT$-128D$ and RC-SIFT$-64D$ perform consistently in case of scale change. Moreover, it presents that SURF$-64D$ descriptors perform the worst in this case.

5.1.7.6 Perspective Change To test the invariance of descriptors against perspective change. 500 different scenes of the benchmark database [136] were selected for which there are two images taken at different viewpoints. The first image of each scene is used as a query and the other as a database image. The results of SIFT$-128D$, SIFT$-64D$, SURF$-64D$, RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) are 62.61, 60.20, 43.92, **72.05** and 71.71, respectively. The results present that, contrary to the other transformations, the robustness decreases of all proposed descriptors against perspective change. The reason is: the change of viewpoint affects the computation of the dominant orientation, which in role influences the computation of descriptor vectors of keypoints. However, SIFT$-128D$ and RC-SIFT$-64D$ still have the best performance (MAP or MR). This is due to the applied hypothesis to construct the RC-SIFT (described in Subsection 5.1), which increases the robustness of the descriptors the changes in viewpoint. Figure 39 presents samples of the UKBenchmark dataset which include change in viewpoint. It shows that the RC-SIFT$-64D$ outperforms the original SIFT and the RC-SIFT$-32D$. The RC-SIFT$-32D$ still performs well in case of viewpoint change even the dimensionality of its descriptors is highly compressed.

5.1.8 Combination of Image Transformations

We accomplished various experiments to verify the robustness of the original SIFT$-128D$, SIFT$-64D$, SURF$-64D$ and RC-SIFT$-64D$ descriptors against combinations of image transformations in the field of image-ND retrieval. We published the details and results of these comparisons in [13] and [15]. We considered the following kinds of combinations: illumination increase or decrease with rotation change, illumination increase or decrease with adding noise, and finally rotation change with adding noise. To achieve this, we convolved the $UKbench-T$ described in Subsection 5.1.5.3 with a combination of different kinds of transformations. The structuring of descriptors was completed using a vocabulary tree of depth $L = 3$ and initial centers $k = 10$. The similarity was computed using the $L1-$norm. A query image is considered to be retrieved if its corresponding database image appears at

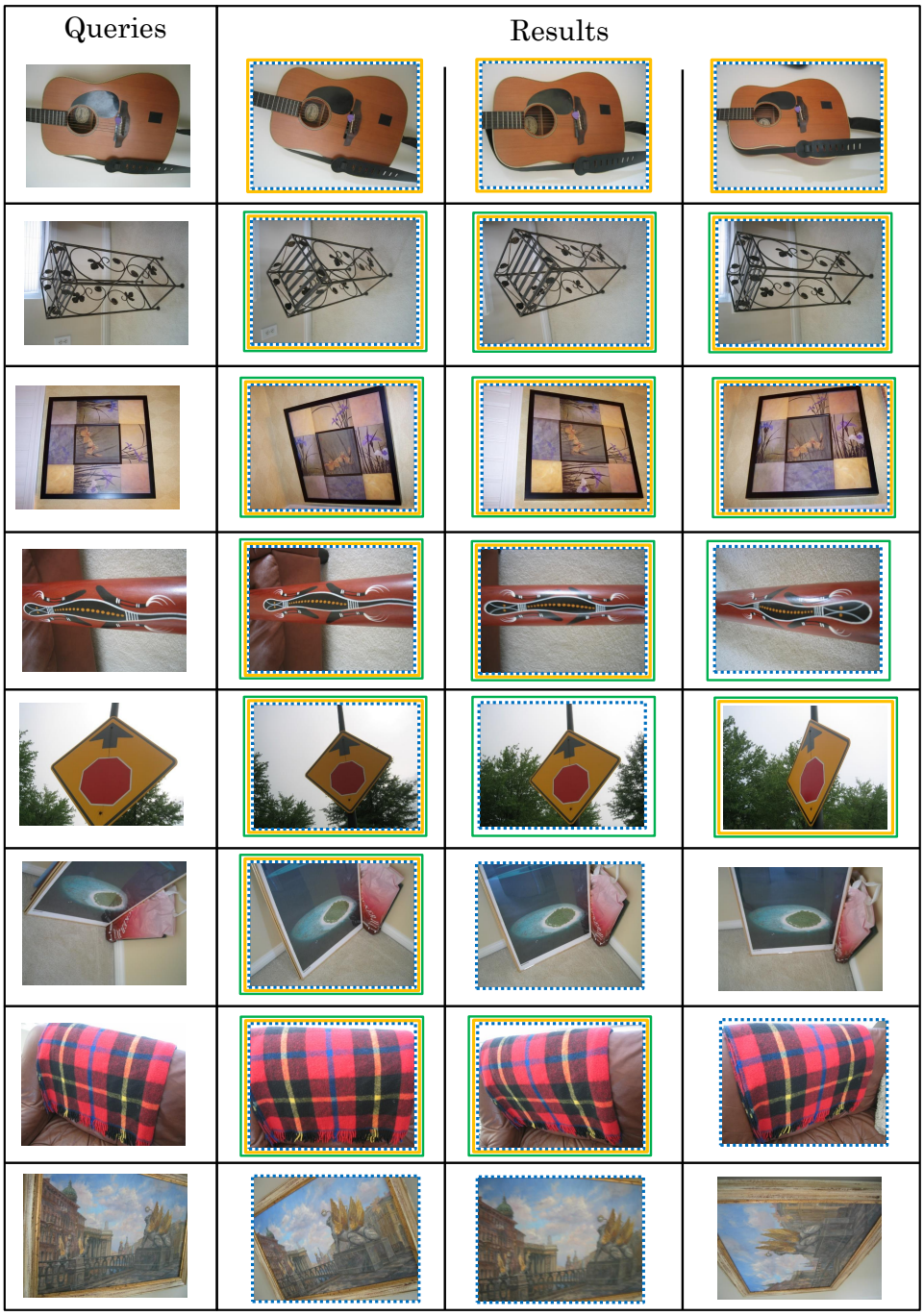


Figure 39: Samples of queries and their relevant images in case of the change of viewpoint. The successfully retrieved images are marked by green boxes for the original SIFT, blue dash box for the RC-SIFT−64D and orange boxes for the RC-SIFT−32D.

the top of the retrieved images. The evaluation was accomplished utilizing the MR or MAP, which have exactly the same values in this case.

5.1.8.1 Combination of Illumination and Rotation To evaluate the robustness of the proposed descriptors to the combination of illumination and rotation change, the illumination of the $UKbench - T$ images was increased using the values $50, 70, 100, 120$ [13], [102]. After that, the illuminated images were rotated at different angles in a clockwise direction (i.e. 40^o, 135^o, 215^o, 250^o, 300^o) to generate 20 benchmarks each of them contains 500. To verify the robustness of the SIFT$-128D$, SIFT$-64D$, SURF$-64D$, and RC-SIFT$-64D$, to illumination decrease and rotation, the values $30, 50, 70, 90$ are subtracted from all channels of the pixels of each image after that the same previous rotation angles are applied to generate 20 benchmarks too. Tables 15 and 16 show robust performance for all used rotation angles when small amount of illumination change is applied to images. However, these tables present a decrease of performance for all rotation angles when the illumination change increase [15]. Hence, we deduce that the increment of illumination combined with various angles lowers the stability of the extracted descriptors. Moreover, the comparison of the Tables 15 and 16 with the results of applying the illumination or the rotation change separately [13] (presented in Tables 10, 12 and 13) clarify that the performance of all proposed descriptors decrease when the rotation and illumination changes are combined (for rotation change the performance is more than 92% and up to 100% for illumination change [13], [136]). However, the results in Tables 15 and 16 present that illumination change affects the performance of SIFT$-128D$, SIFT$-64D$, SURF$-64D$ and RC-SIFT$-64D$, more than rotation change.

Table 15: The comparison of SIFT$-128D$, SIFT$-64D$, SURF$-64D$, and our RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) using a ground truth illuminated and rotated benchmarks (generated from $UKbench - T$). The performance is presented by means of MAP or MR, which are equivalent in this case. The results are presented for two levels of illumination increase (i.e. 50, 120) for each five rotation values are applied: 40^o, 135^o, 215^o, 250^o, 300^o. Θ and $Ill+$ refer to the rotation and illumination increase respectively.

Ill+	Θ	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Ill-Inc 50	40^o	**76.2**	75.6	75.5	**75.9**	75.8
	135^o	**78.0**	76.0	76.9	77.7	**77.8**
	215^o	**78.0**	76.2	76.9	**77.8**	**77.8**
	250^o	**78.0**	76.2	76.9	**78.0**	75.7
	300^o	**76.0**	75.3	**75.8**	**76.0**	75.7
Ill-Inc 120	40^o	**31.0**	29.6	28.9	**31.0**	**30.0**
	135^o	29.4	29.2	29.2	29.6	29.2
	215^o	**30.0**	29.2	29.0	**29.6**	29.5
	250^o	29.4	**29.6**	29.1	**29.6**	29.5
	300^o	29.1	29.0	28.7	**29.3**	**29.2**

Table 16: The performance evaluation of SIFT−128D, SIFT−64D, SURF−64D, RC-SIFT−64D(R) and RC-SIFT−64D(C) in case of combining illumination decrease and rotation. The performance is presented by means of MAP or MR, which are equivalent in this case. The results are presented for two levels of illumination decrease (i.e. 30, 90) for each five rotation values are applied: 40^o, 135^o, 215^o, 250^o, 300^o. Θ and $Ill-$ refer to the rotation and illumination decrease respectively.

Ill-	Θ	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Il-Dec 30	40^o	**79.8**	78.6	78.2	79.5	79.2
	135^o	**75.8**	75.6	74.8	**75.8**	75.5
	215^o	**76.2**	76.0	75.8	76.0	75.8
	250^o	78.6	78.6	78.0	**78.8**	78.0
	300^o	77.3	77.5	77.4	**78.0**	77.6
Il-Dec 90	40^o	52.8	52.6	50.7	**53.1**	52.8
	135^o	49.0	**49.6**	48.7	**49.6**	49.2
	215^o	50.8	**51.2**	50.2	**51.2**	50.3
	250^o	48.8	50.3	50.8	**51.2**	50.6
	300^o	47.3	47.6	47.7	**49.3**	48.7

5.1.8.2 Combination of Noise and Illumination Change To test the robustness of the proposed descriptors to illumination change and added noise, the salt and pepper noise, with densities of 15% and 35%, was applied to $UKbench - T$ images. After that, the brightness of noised images was increased utilizing the values $50, 70, 100, 120$ [13], [102] or decreased by subtracting the values $30, 50, 70, 90$ from all color channels of image pixels. As a result, we obtained 16 benchmarks, each contains a combination of additional noise and illumination change. Tables 17 and 18 present the performance of various descriptors by adding noise and increasing the illumination with values 50 and 120 or decreasing it with the values 30 and 90 separately. These tables show a decrease in the performance of all descriptors i.e, SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D descriptors comparing to the results presented in Tables 11, 12 and 13 where the transformations are applied separately. The performance in case of combination is always lower than the minimum obtained performance by applying the transformations separately. In case of applying the salt and pepper noise with density of 35%, all presented descriptors are not stable anymore [15].

5.1.8.3 Addition of Noise and Rotation Similar to the settings in the previous Paragraph 5.1.8.2, we applied a combination of adding noise and rotation by firstly adding the salt and pepper noise with density of 15% or 35% to the $UKbench - T$ benchmark (the detail of adding noise is described in Paragraph 5.1.7.2 [13]). Secondly, we rotated the noised images at different angles in a clockwise direction (i.e.

Table 17: The performance of SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D when a combination of salt and pepper noise and illumination increase is applied on $UKbench - T$ images. The results are presented for two level of noise densities (i.e. 15% and 35%), for each the illumination increases applying the values 50 and 120. MAP and MR are identical in this case and they present the performance. In this table, SP and $Ill+$ refer to the salt pepper noise and illumination increase, respectively.

Ill+	SP	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Ill-Inc 50	15%	**68.0**	67.6	67.2	67.6	67.1
	35%	5.8	5.0	4.7	**6.2**	5.8
Ill-Inc 120	15%	32.8	32.6	32.3	**34.7**	34.2
	35%	3.00	3.1	2.8	**3.4**	3.1

Table 18: The performance of SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D using a combination of a salt and pepper noise and illumination decrease. The results are presented for two level of noise densities (i.e. 15% and 35%) for each the illumination decreases using the values $Dr = 50$ and $Dr = 120$. The performance is presented by means of MAP or MR. In this table, SP and $Ill-$ refer to the salt pepper noise and illumination decrease, respectively.

Ill-	Noise	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Il-Dec 30	SP 15%	**73.6**	73.0	73.0	73.4	73.1
	SP 35%	8.0	8.0	7.1	**10.2**	9.6
Il-Dec 90	SP 15%	36.7	36.7	35.3	**37.0**	36.6
	SP 35%	6.2	5.8	5.3	**7.0**	6.8

40^o, 135^o, 215^o, 250^o, 300^o) to generate ten benchmarks of noised rotated images. Table 19 describes how the performance (i.e. MAP or MR) of all proposed descriptors decrease very strongly for a fixed rotation angle when the noise density increases. SIFT−128D and RC-SIFT−64D perform better than the other descriptors. However, the results in Table 19 presents that detected features of all detectors and descriptors are not stable anymore when the amount of noise is increased to 35%.

5.1.8.4 Combination of Blur with Rotation, Noise or Illumination To study the effect of image blurring combination with various kinds of image transformations on the performance of the original SIFT−128D, SIFT−64D, SURF−64D and RC-SIFT−64D, the $UKbench - T$ benchmark images were firstly blurred by convolving the image with Gaussian filter using three variations i.e., $\sigma^2 = 5$, $\sigma^2 = 10$ and $\sigma^2 = 15$ (the process of fileting is described in Paragraph 5.1.7.4 [13]). After that, the illumination of the blurred images was increased or decreased using the same values presented in Paragraph 5.1.8.1. The best Performance of near-duplicate retrieval is obtained when the Gaussian filter with $\sigma^2 = 5$ is used. This performance is below 25% for all proposed descriptors. However, by applying the Gaussian filter

Table 19: The performance comparison of SIFT−128D, SIFT−64D, SURF−64D, RC-SIFT−64D(R) and RC-SIFT−64D(C) in case of applying salt and pepper noise and rotation to $UKbench - T$ benchmark. The results are presented for two noise densities (i.e. 15%, 35%) for each five rotation values are applied: 40^o, 135^o, 215^o, 250^o, 300^o. MAP and MR are employed to compute the performance.

Noise	Θ	SIFT-128	SIFT-64	SURF	RC-SIFT(R)	RC-SIFT(C)
Noise 15%	40^o	**32.0**	31.8	30.7	31.7	**32.0**
	135^o	**39.8**	39.5	39.2	39.2	39.3
	215^o	40.0	40.0	40.2	**41.2**	**41.2**
	250^o	43.0	42.0	42.3	**43.8**	43.0
	300^o	37.6	36.8	37.1	**40.0**	**40.0**
Noise 35%	40^o	**6.2**	6.1	5.8	6.0	5.8
	135^o	5.6	5.6	5.5	**6.0**	5.3
	215^o	5.4	5.6	**5.9**	5.5	5.7
	215^o	5.8	5.2	5.8	**6.2**	**6.2**
	300^o	5.4	5.0	5.4	6.0	**6.2**

with $\sigma^2 = 10$ or $\sigma^2 = 15$ the performance, which is computed as MAP or MR, decreases to lesser than 16% or 13%, respectively.

By combining of blur and rotation change (the rotation values are: 40^o, 135^o, 215^o, 250^o, 300^o), the performance in this case is not more than 13% for all applied methods (i.e. SIFT−128D, SIFT−64D, SURF−64D and our RC-SIFT−64D(R) and RC-SIFT−64D(C)) for all rotations when $\sigma^2 = 5$. Whereas, the performance decreases to 8% or 4% when $\sigma^2 = 10$ or $\sigma^2 = 15$, respectively.

A Combination of the Gaussian blur with the salt pepper noise obtain recall lesser than 15% when the density of noise is 15% and the blurring variation is $\sigma^2 = 5$. The number of retrieved images decreases to 10% or 8% when the blurring increases to $\sigma^2 = 10$ or $\sigma^2 = 15$, respectively. The results of combining Gaussian blur with different kinds of image transformations present that the performance of all proposed descriptors decreases strongly and the extracted descriptors become unstable when more blur is convolved to images.

5.1.8.5 Feature Quality by transformations combination The results of convolving one or more types of transformations with images present that the quality of extracted RC-SIFT−64D keypoints decreases when the amount of convolved transformation increases or when combinations of transformations are applied. To clarify the difference of feature quality in these cases, we present samples of the UKbench benchmark with their SIFT keypoints in Figure 40. The first row in Figure 40 clarifies that the illumination increase produces about the same amount

and locations of keypoints as in the original image but the blur produces only two keypoints. However, the combination of them obtains keypoints that they differ in amount, locations and descriptors therefore, matching them with the keypoints of the original image identifies weak similarity. The second row in Figure 40 presents a case where illumination increases and noise are combined together. In this example, the amount of extracted keypoints after the combination is about the same as the one of the original image, but they differ in locations and descriptors. The third row shows that the rotation causes a change in the form of descriptors whereas, illumination-decrease reduces the number of keypoints therefore, the combination of both of them produces the least amount of keypoints, that have different locations and descriptors of the original image. The fourth row presents a case where blur and noise are combined. In this case, blur reduces the amount of extracted keypoints but noise doubles them, therefore, the combination of them obtains about the same amount of keypoints as the original image. However, they differ in locations and descriptors (many features are produced by the noise). From those samples in Figure 40 we conclude that the combination of transformations decreases the robustness of keypoints, specifically when the amount of changes increases.

5.1.9 Conclusion

In the first part of this chapter, we answered **RQ.1**(a) and (b) by presenting our method to adapt the SIFT features and reduce their dimensionality. Our proposed RC-SIFT$-64D$ requires lesser processing time than the original SIFT, but it preserves the robustness of the constructed descriptors. We inspired our initial idea from the fact "*the sparsity of fixed amount of feature increase as their dimensionality increase*" [3]. We verified the performance of the RC-SIFT$-64D$ (for both horizontal and vertical compression), RC-SIFT$-32D$, RC-SIFT$-16D$ against the original SIFT$-128D$ [116], SIFT$-64D$ [102], SURF$-64D$ [27] to solve image ND-retrieval tasks employing a benchmark which contains different kinds of indoor/outdoor images. The experiments showed a slight improvement in matching results when tested on various benchmark databases. However, the RC-SIFT$-64D$ needs shorter indexing time and lesser memory than the original SIFT$-128D$. The RC-SIFT$-32D$ and RC-SIFT$-16D$ showed decreased performance due to the compression of descriptors information in both directions at once.

We presented and evaluated the performance of the RC-SIFT$-64D$ descriptor to solve the near-duplicate retrieval task in two cases: Firstly, for benchmarks of different sizes. Secondly, using the same benchmark but for different amounts of extracted features. The experiments showed a slight improvement in matching results compared to the original SIFT$-128D$ when tested employing various settings.

We also verified the robustness and stability of our suggested RC-SIFT$-64D$ against different types of image modifications such as illumination change, rotation, blurring, scale change, and viewpoint change. However, the results showed that the RC-SIFT$-64D$ descriptors are invariant (like the original SIFT$-128D$ [116],

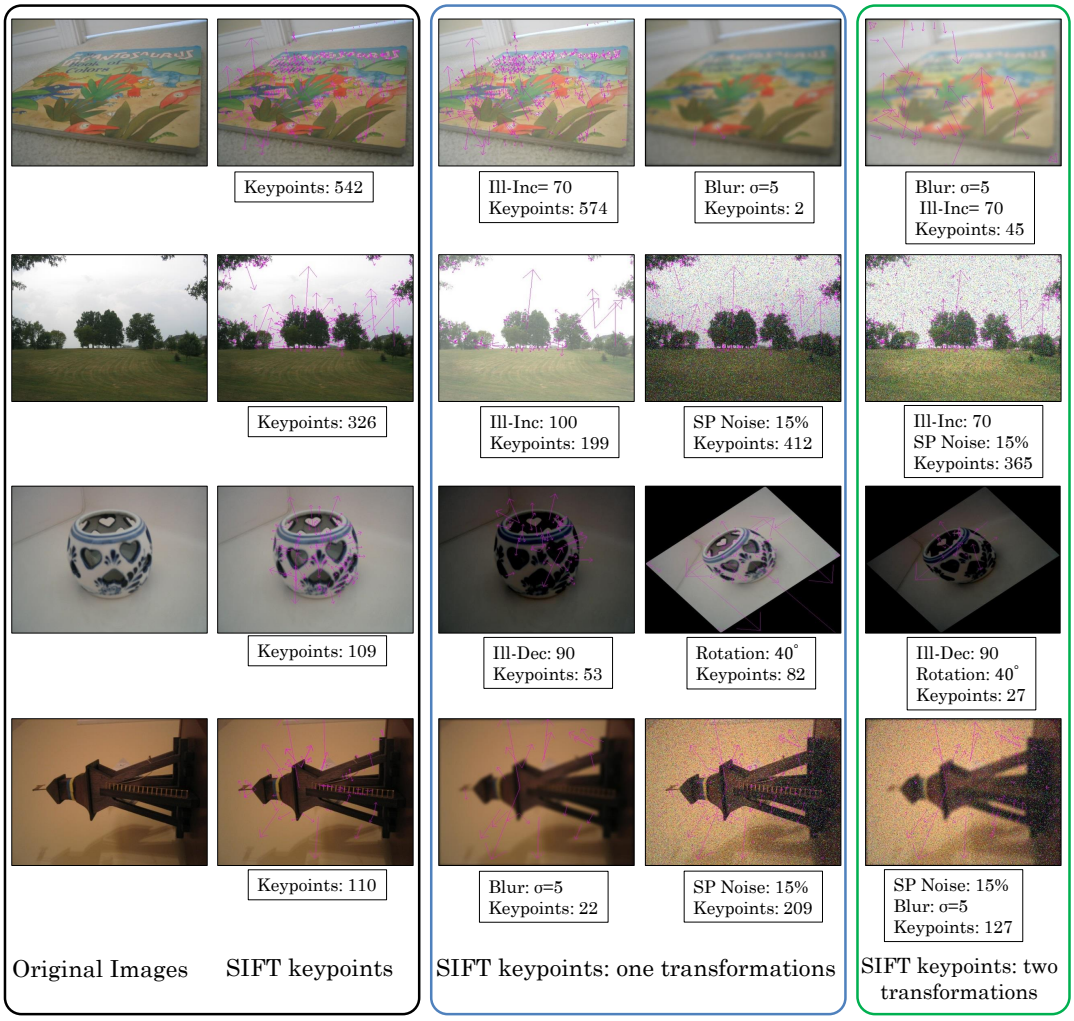


Figure 40: Comparison of amount, location, and descriptors of SIFT keypoints, employing various types of transformations. The first column presents the original images and their keypoints. The second column displays the keypoints when only one type of transformation is applied to images. The third column shows the keypoints when transformation combinations are convolved with images.

SIFT−64D [102], and SURF−64D [27] descriptors) to image transformations in specific ranges of modifications. Moreover, the results presented that RC-SIFT−64D descriptors are more robust than the others to substantial changes in rotation, some kinds of noise, increased blur, and viewpoint change. By comparing of our results with the results of deep learning techniques presented in [197] (explained in Subsection 3.5) we find that both traditional methods, i.e. SIFT−128D [116] SIFT−64D [102], SURF−64D [27] and,RC-SIFT−64D and deep learning techniques such as Alexnet with the double-channel model and VGG16 with double-channel network [197], have similar behavior, e.g. the performance of both of them decreases in case of illumination change and perspective change.

We implemented a special-purpose study to evaluate the robustness and stability of the original SIFT$-128D$, SIFT$-64D$, SURF$-64D$ and RC-SIFT$-64D$ against combinations of image transformations. The results showed that all proposed descriptors are robust to specific amounts of combinations. However, the robustness of descriptors decreases, when the amount of the combined transformations increases specifically, in case of combining noise with other kinds of transformations. When image transformations are combined with blur, the performance of all proposed descriptors decreases very strongly. So, in this case, the extracted descriptors lose their robustness. Noteworthy, the performance of the original SIFT$-128D$ [116], SIFT$-64D$ [102], SURF$-64D$ [27] are more than 90% for each of rotation and illumination changes separately but, by combining the transformations, the performance of all discussed descriptors decreases specifically when the illumination change increases.

An important issue we observed through our experiments is that the extraction of various quantities of features of a specific benchmark affects the performance of the SIFT$-128D$ and RC-SIFT$-64D$. Therefore, in the second part of this chapter, we study the factors that may reduce the amounts of detected features but concurrently enhance the performance of descriptors in solving the near-duplicate retrieval task.

5.2 Approaches for Truncation of SIFT Keypoints

The scale invariant feature transformation algorithm (SIFT) has been widely used for near-duplicate retrieval tasks. Most studies and evaluations published so far focused on increasing retrieval accuracy by improving descriptor properties and similarity measures. Contrast, scale, and orientation properties of the SIFT keypoints were used in computing the SIFT descriptor, but their explicit influence in the feature matching step was not studied. Moreover, it has not been studied yet how to specify an appropriate criterion to extract (almost) the same number of SIFT keypoints of all images in a database. In the following, we study the effects of contrast and scale properties of SIFT keypoints when ranking and truncating the extracted descriptors. In addition, we evaluate if scale, contrast, and orientation features can be used to bias the descriptor matching scores, i.e. if the keypoints are quite similar in these features, we enforce a higher similarity in descriptor matching. We provide results of a benchmark data study using the proposed modifications in the original SIFT$-128D$ and on the region compressed SIFT (RC-SIFT$-64D$).

5.2.1 Truncating the List of Keypoints Based on their Properties

The number of extracted keypoints is not well defined by explicit formal rules neither in the original SIFT$-128D$ nor the RC-SIFT$-64D$. For each image, the amount of extracted keypoints can vary between zero and thousands. The variety of this number produces by using various values of contrast and Gaussian thresholds. To study the influence of truncation methods, we suggest ranking the features based

on either decreasing contrast or scale (depending on the aim of the experiment) and then truncating them using a predefined initial number of accepted keypoints NF. We do not use the orientation property to truncate the keypoints since the orientation does not give any information about the robustness of keypoints (like the contrast property) or their location in the image pyramid (like the scale property). However, the final number of extracted keypoints will be greater than NF. Since the computation of the dominant orientation of each keypoint produces, sometimes, new keypoints that share the locations of the old ones (as described in Section 2.3.4). Hence, after applying this step, the number of the extracted keypoint is lesser than $NF + \epsilon$ where ϵ denotes the number of created keypoint through multiple dominant orientations.

5.2.2 Involving Keypoints Properties in Matching Process

In the previous research, the scale, contrast, and orientation properties of keypoints have not been involved in the matching stage. The matching process of the original SIFT and the RC-SIFT keypoints has been achieved by comparing only the descriptors i.e. without considering the scale, contrast, and orientation properties of keypoints. In this research, after the feature truncation step, we suggested analyzing the effect of using those properties in the matching process. For this, we proposed that the keypoint matches detected at the same scale, contrast, or orientation have higher similarity than those that differ at one or more of these properties. This idea is valuable since the robustness of keypoints depends on their scale, orientation, and contrast properties. Therefore, we analyzed these properties to determine whether they improve the near-duplicate retrieval performance and which of them produces the best (or worst) impact. We published the details of this analysis in [14]

5.2.3 Step of Involving Feature Properties

We start with extracting the SIFT$-128D$ and RC-SIFT$-64D$ keypoints of images. These keypoints are structured using hierarchical $k-$means clustering as described in [13] and Subsection 5.1.3. Based on the hierarchical clustering, a bag of visual words is constructed and employed to represent images in terms of vectors (see also [90] and [136]). To compare a query image with database images the following steps are carried out:

- **Weights definition:** In this step weights related to contrast W_{cont}, scale W_{scl} and orientation W_{ori} properties are defined. These weights are necessary to involve the influence of these properties in feature matching stage. In this research, we define all used weights in terms of unique value W i.e. contrast, scale and orientation are given the same degree of importance.

- Properties criteria for matching: The weights W_{cont}, W_{scl} and W_{ori} are initial-

ized with a value $W_{initial} = 1$. i.e.

$$W_{cont} = W_{initial} \ , \qquad W_{scl} = W_{initial} \ , \qquad W_{ori} = W_{initial} \tag{50}$$

After that, we compute the difference of contrast ($Cont$), scale (Scl) and orientation (Ori) as follows:

$$\Delta Cont = |Cont(f_q) - Cont(f_{db})| \tag{51}$$

$$\Delta Scl = |log(Scl(f_q)) - log(Scl(f_{db}))| \tag{52}$$

$$\Delta Ori = |Ori(f_q) - Ori(f_{db})| \tag{53}$$

If the following conditions are satisfied:

$$\Delta Cont \leq thr_{cont} \ , \qquad \Delta Scl \leq thr_{scl} \ , \qquad \Delta Ori \leq thr_{ori} \tag{54}$$

then a weight $W \in]0, 1[$ is multiplied with W_{cont}, W_{scl} and W_{ori} as follows:

$$W_{cont} = W_{cont} \times W \ , \qquad W_{scl} = W_{scl} \times W \ , \qquad W_{ori} = W_{ori} \times W \tag{55}$$

thr_{cont}, thr_{scl} and thr_{ori} refer to thresholds related to contrast, scale, and orientation respectively. The values of these thresholds and W are determined heuristically. If the relations (53), (51) or (52) are not satisfied , the value of one is assigned to W_{ori}, W_{cont} or W_{scl}.

- Features matching: For each query image vector $v(q) = v_1(q), v_2(q), ..., v_{k^L}(q)$ and database vector $v(db) = v_1(db), v_2(db), ..., v_{k^L}(db)$, (where k^L is the number of leafs in the hierarchical clustering), the distance between of the corresponding components is computed as the average of the following three distances:

$$d_{cont}(v_i(q), v_i(db)) = W_{cont} \, |v_i(q) - v_i(db)| \tag{56}$$

$$d_{scl}(v_i(q), v_i(db)) = W_{scl} \, |v_i(q) - v_i(db)| \tag{57}$$

$$d_{ori}(v_i(q), v_i(db)) = W_{ori} \, |v_i(q) - v_i(db)| \tag{58}$$

- Image matching: Depending on the previous steps the distance between a query vector $v(q)$ and a database vector $v(db)$ is computed as:

$$d(v(q), v(db)) = \frac{1}{N_q N_{db}} \sum_{n=1}^{k^L} Average(d_{cont}, d_{scl}, d_{ori}) \tag{59}$$

where N_q and N_{db} present the number of extracted features of the query and database images respectively. When we consider the effect of only one of these properties then:

$$d(v(q), v(db)) = \frac{1}{N_q N_{db}} \sum_{n=1}^{k^L} d_{attribute}) \tag{60}$$

where *attribute* refers to contrast, scale, or orientation.

The steps of feature truncation and involving properties in the matching step are clarified in Figure 41. Based on these steps the contrast, scale, and orientation properties are used in the matching process. In the following section, we discuss in experiments the influence of using these properties to solve the near-duplicate retrieval task.

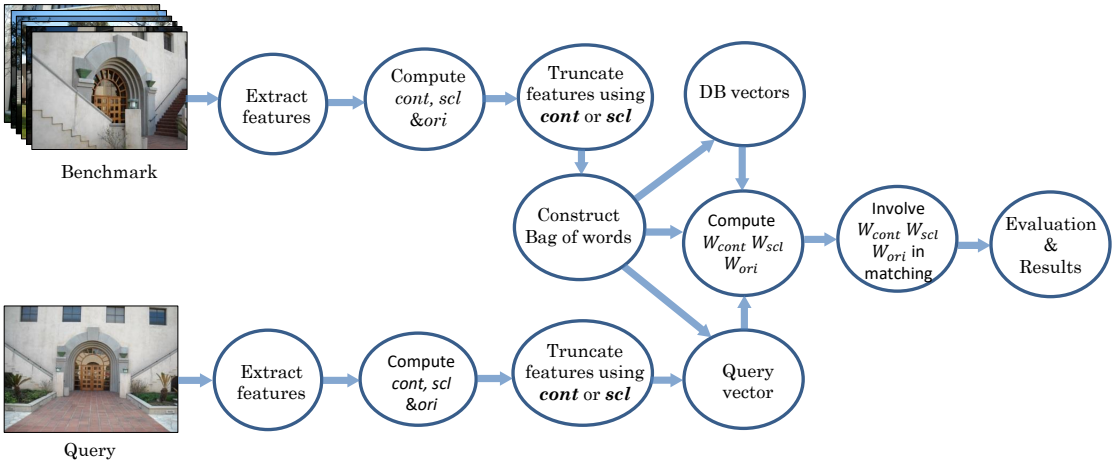


Figure 41: The flowchart of feature truncating and matching when the contrast, scale, and orientation properties of keypoints are used in the matching process.

5.2.4 Evaluation Measures

To evaluate the effect of involving the properties (i.e. scale, contrast and orientation) of the SIFT$-128D$ and the RC-SIFT$-64D$ keypoints in matching process, we extract the original SIFT$-128D$ and the RC-SIFT$-64D$ (we employ only the RC-SIFT$-64D$(R) since both RC-SIFT$-64D$(R) and RC-SIFT$-64D$(C) have equivalent performances) keypoints of two different image benchmarks. After that, we rank and truncate the list of keypoints based on the contrast or scale properties. The descriptors are indexed and the vectors of images are constructed using the hierarchical k-mean clustering. We compute the similarity between a query vector $v(q)$ and database vectors $v(db)$ by applying the relation (59). In case of involving the properties separately, the similarity is computed using the relation (60). The results are evaluated by computing the mean recall value (MR), mean average precision (MAP) and variance of recall (VR) as described in Subsection 2.5.2.

5.2.5 Benchmarks Description

To analyze the influence of involving the scale, contrast, and orientation properties of the SIFT keypoints in feature selection and matching steps, we selected two image benchmarks, i.e. UKbench and Caltech buildings (described in Sections 4.2 and 4.3). We employed two benchmarks to verify whether the content and properties of images

Table 20: The retrieval performance of SIFT−128D RC-SIFT−64D when the lists of features are ranked and truncate based on their ***scale property***. The mean recall is computed based on the top four (MR4) and then top ten (MR10) retrieved images of the *Caltech-Buildings* database.

Descriptors properties			SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MR4	MR10	MR4	MR10
			40.02	49.51	40.70	50.06
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	35.00	47.07	35.00	47.50
ΔScl<0.2			30.46	42.00	30.08.0	41.96
ΔScl<0.2		$\Delta Ori < \frac{\pi}{8}$	36.00	48.41	35.87	47.66
	$\Delta Cont$<0.1		36.52	50.81	37.00	51.13
		$\Delta Ori < \frac{\pi}{8}$	42.50	54.04	43.00	55.10
		$\Delta Ori < \frac{\pi}{10}$	43.71	56.00	44.02	56.23
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**44.21**	**57.22**	**44.85**	**58.00**

affect the retrieval results. These benchmarks contain indoor/ outdoor images of various scenes in groups of four or five images for each scene. The images of each scene differ in viewpoint, scale, lightness, or combination of these conditions. The Caltech-Buildings [11] contains 250 images for 50 different buildings around the Caltech campus. This benchmark has high-resolution images (the resolution of each image is 2048×1536 pixels). The second image database is UKbench [136] (this database can be download from [135]). This image database contains about $10,000$ images of resolution 640×480 pixels.

5.2.6 Result and Analysis

We evaluated the results of the SIFT−128D and the RC-SIFT−64D algorithms using the Caltech-Buildings and UKbench benchmarks in two cases. Firstly when the extracted lists of keypoints are ranked and truncated depending on the scale property. Secondly, when they are ranked and truncated based on the contrast property. In the empirical study, we noticed that the sets of extracted keypoints in both cases are not equivalent when we consider only the top NF extracted features (i.e. the position of features in the ranked list differs). Moreover, we found that the employing of the dominant orientation (see Subsection 5.2.1) creates a set of new keypoints. The size of this set is $\epsilon \leq \frac{NF}{3}$ so that the total number of extracted feature is not more than $NF + \frac{NF}{3}$. We determine the value of NF depending on the resolution of images.

5.2.6.1 Truncation based Scale: Caltech-Buildings Benchmark For the Caltech-Buildings benchmark, due to the high resolution of images of this bench-

Table 21: The *mean average of precision* and the *variance of recall* of SIFT−128D and RC-SIFT−64D when the lists of keypoints are ranked and truncate based on their ***scale property***. The MAP and VR are computed based on the top four retrieved images of the *Caltech-Buildings* database.

Descriptors properties			RC-SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MAP	VR	MAP	VR
			37.50	9.47	37.97	9.49
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	32.12	**8.50**	32.08	**8.6**
ΔScl<0.2			28.20	10.14	29.00.0	10.11
ΔScl<0.2		$\Delta Ori < \frac{\pi}{8}$	32.88	10.48	31.52	10.23
	$\Delta Cont$<0.1		33.00	9.43	34.08	9.37
		$\Delta Ori < \frac{\pi}{8}$	38.62	8.81	39.56	8.23
		$\Delta Ori < \frac{\pi}{10}$	40.21	8.85	41.02	8.14
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**41.02**	8.08	**41.78**	8.06

mark [11], huge amount of keypoints may be extracted from them therefore, we determined $NF = 1600$ to be the number of accepted keypoints. In case of ranking the keypoints based on their decreasing scale values, Table 20 presents the mean recall of the SIFT−128 and the RC-SIFT−64 algorithms receptively. The first row of this table presents the results after truncating the features but without including any weights. This table shows that the best performance of both SIFT−128 and RC-SIFT−64 is achieved by considering the orientation and contrast properties and ignoring the scale, specifically, when $\Delta Cont < 0.1$ and $\Delta Ori < \frac{\pi}{10}$. By employing only the orientation weights the SIFT−128 and the RC-SIFT−64 show the second-best performance when $\Delta Ori < \frac{\pi}{10}$. The worst results were obtained when all three properties were included in the matching process. We determined the orientation threshold to be $thr_{ori} \leq \frac{\pi}{8}$ or $thr_{ori} \leq \frac{\pi}{10}$. We checked other values of thr_{ori} but then the performance decreased. For the scale and contrast properties, we experimented different values too but the best performance of the SIFT−128 and the RC-SIFT−64 was found when $thr_{scl} \leq 0.1$ and $thr_{cont} \leq 0.1$. In case of satisfying one of the relations (51), (52) or (53), the value $W = 0.9$ is assigned to W_{cont}, W_{scl} or W_{ori} respectively. We checked other values for the weights in the range $]0, 1[$ but we got the best performance when the value 0.9 was used. Table 21 presents the mean average precision and the variance of recall of the SIFT−128 and the RC-SIFT−64 respectively. It describes that the best mean average of precision was obtained for the same thresholds where the best mean recall was found. Tables 20 and 21 describe how the variance of recall decreases when the mean of recall increases.

Table 22: The performance of SIFT−128D and RC-SIFT−64D when the lists of features are ranked and truncated based on their ***scale property***. The mean recall is computed based on the top three (MR3) and then top ten (MR10) retrieved images of the UKbench database.

Descriptors properties			SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MR4	MR10	MR4	MR10
			49.30	58.70	50.73	60.70
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	44.03	53.05	46.00	55.05
ΔScl<0.2			41.06	52.20	41.22	53.16
ΔScl<0.2		$\Delta Ori < \frac{\pi}{8}$	44.03	56.72	45.00	57.06
	$\Delta Cont$<0.1		44.56	57.30	45.40	57.82
		$\Delta Ori < \frac{\pi}{8}$	54.00	66.14	54.80	66.38
		$\Delta Ori < \frac{\pi}{10}$	54.82	67.08	55.00	67.20
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**55.12**	**68.00**	**55.41**	**68.20**

5.2.7 Truncation based Scale: UKbench Benchmark

The resolution of images in the UKbench [135] database is not high (it is only 640×480) therefore, we extracted $NF = 500$ keypoints of each image. After that, we ranked and truncated the keypoints based on the decreasing value of the scale property. The truncated keypoints were indexed utilizing hierarchical k-means of depth four and ten leaf nodes. Table 22 presents the performance of SIFT−128 and RC-SIFT−64 descriptors employing the UKbench database. It displays that the best mean recall for the SIFT−128 and the RC-SIFT−64 are obtained when the scale and contrast properties are neglected or when only the scale property is skipped. Table 23 shows the mean average of precision and variance of recall for this benchmark which are equivalent to the results presented in Table 21. The comparison of Tables 20, 21, 22 and 23 explains that in case of truncate the keypoints based on their scale property the best performance depends on the involved properties in the weights and is independent of the types of images and their resolution.

5.2.8 Truncation based Contrast: Caltech-Buildings Benchmark

In case of keypoints truncation based on the contrast property, we used the Caltech-Buildings Benchmark and we set $NF = 1600$ to compare the results with those found when the features were truncated based on the scale property. Tables 24 and 25 present the results for both SIFT−128 and RC-SIFT−64 when the Caltech-Buildings database is used. The performances of the SIFT−128 and the RC-SIFT−64 increase when the scale and orientation properties are skipped and then when only the scale property is ignored. The SIFT−128 and the RC-SIFT−64 perform the best when $\Delta Cont$<0.1 and $\Delta Ori < \frac{\pi}{10}$ and the scale is neglected. Equivalent results are

Table 23: The performance of SIFT−128D and RC-SIFT−64D when the lists of features are ranked and truncated based on their ***scale property***. The mean average precision and variance of recall of the UKbench benchmark.

Descriptors properties			SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MAP	VR	MAP	VR
			46.00	11.13	47.03	10.70
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	41.19	11.00	43.30	11.07
ΔScl<0.2			37.82	10.03	38.02	9.18
ΔScl<0.2		$\Delta Ori < \frac{\pi}{8}$	39.70	9.97	40.06	9.25
	$\Delta Cont$<0.1		40.21	11.10	41.62	10.32
		$\Delta Ori < \frac{\pi}{8}$	50.94	10.58	51.17	10.03
		$\Delta Ori < \frac{\pi}{10}$	51.30	10.25	51.93	9.84
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**52.00**	**9.21**	**52.37**	**8.96**

Table 24: The retrieval performance of SIFT−128D and RC-SIFT−64D when the *Caltech-Buildings* database is used. The lists of features are ranked and truncated based on their ***contrast property***.

Descriptors properties			SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MR4	MR10	MR4	MR10
			37.00	48.50	37.80	49.00
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	36.50	46.00	36.70	46.62
	$\Delta Cont$<0.1		37.58	47.50	39.03	50.80
		$\Delta Ori < \frac{\pi}{8}$	37.00	48.50	37.90	49.00
		$\Delta Ori < \frac{\pi}{10}$	37.89	49.60	38.30	50.00
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**39.00**	**49.82**	**39.19**	**50.60**

obtained for the UKbench database considering the contrast property.

5.2.9 Conclusion

We reviewed **RQ.1**(c) by analyzing the role of the scale, contrast, and orientation properties in improving the performance of the original SIFT and the RC-SIFT algorithms and reduce the amount of the extracted features. This is important since high-resolution images produce at most many keypoints. Moreover, by using pre-defined parameters for the SIFT and the RC-SIFT approaches, the number of the extracted keypoints varies between zero and thousands. The matching and processing of these keypoints are time and memory-consuming. Therefore, we introduced our method to specify the amount of the extracted keypoints of images.

Table 25: The retrieval performance of SIFT−128D and RC-SIFT−64D when the *Caltech-Buildings* database is used. The lists of features are ranked and truncated based on their ***contrast property***. The mean average precision and variance of recall of the Caltech-Buildings benchmark.

Descriptors properties			SIFT−128D		RC-SIFT−64	
Scale	Contrast	Orientation	MAP	VR	MAP	VR
			35.20	8.82	35.95	8.32
ΔScl<0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{8}$	34.37	9.07	34.90	8.69
	$\Delta Cont$<0.1		35.51	8.91	35.73	8.63
		$\Delta Ori < \frac{\pi}{8}$	35.07	9.05	35.64	8.72
		$\Delta Ori < \frac{\pi}{10}$	35.71	9.73	35.92	9.00
	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**36.70**	**8.60**	**37.05**	**8.14**

We suggested determining a fixed set of accepted keypoints based on the scale or contrast properties [14]. We achieved this by ranking and truncating the obtained lists of features based on their decreasing scale or contrast properties. The number of accepted features depends on the resolution of images in a benchmark and is determined utilizing a region adaptive approach. We found out the inversion ranking of keypoints i.e. ranking them based on decreasing scale or contrast after that, truncates them affects negatively the performance of the original SIFT and the RC-SIFT algorithms.

We studied the effect of involving the properties (i.e. scale, contrast, and orientation) of the truncated keypoints in improving the matching process. We found out, in case of truncating based scale, involving the orientation and contrast properties (with specific thresholds) improves the performance of the SIFT and the RC-SIFT in solving image near-duplicate tasks. Our benchmark studies indicated that using contrast and orientation of keypoints enhances the mean recall. Moreover, we showed that utilizing just the orientation or contrast property obtains the next best performance. When the keypoints are ranked based on the contrast property involving only the contrast property or both contrast and orientation improves the performance. The involvement of scale property decreases the performance of the the original SIFT and the RC-SIFT−64.

The results of keypoints truncation and utilizing weights present that employing only 30% or 20% of the default detected keypoints of the UKBench and Caltech-Buildings benchmarks, respectively are enough to improve the retrieved results and reduce computations costs and memory.

5.3 Summary

In this chapter we reported **RQ.1**(a), (b) and (c) by analyzing and describing two methods to reduce the usage of time and memory of keypoints extraction and matching and at the same time preserve the robustness and the performance of keypoints against various kinds of image transformations. We completed the discussion of **RQ.1** by employing two ideas. The first is to compress the dimension of the SIFT keypoints to produce the RC-SIFT keypoints that have shorter descriptors and therefore, requires shorter indexing and matching time but preserve the quality of keypoints [13], [15]. The second is to truncate the keypoints based on their scale and contrast properties and then involving weights in the matching process. These weights depend on the scale, contrast, and orientation difference between the keypoints [14]. Involving keypoint properties in the truncation and matching stages reduces the time of keypoints matching and improves the quality of matching results.

*We introduce in this chapter our suggested answers to **RQ.2** by presenting two hybrid approaches (F-HS-SIFT and FP-HS-SIFT) to improve and accelerate the retrieval of near-duplicate images. These approaches combine the advantages of global and local features. Our focus in this chapter is on presenting the role of global features in accelerating and improving the retrieval process of ND-images.*

6 Combination of Global and Local Features

Most existing near-duplicate image retrieval systems use high dimensional image vectors based on local features such as SIFT keypoints to represent images. The extraction and matching of these vectors to detect near-duplicates are time and memory-consuming. Global features such as color histograms strongly reduce the dimensionality of vectors and significantly accelerate the matching process. On the other hand, they strongly decrease the quality of the retrieval process. In this chapter, we discuss the possible methods to solve **RQ.2** by proposing two hybrid approaches (F-HS-SIFT and FP-HS-SIFT) to improve the retrieval quality and reduce the computation time by applying a robust filtering process using global features optimized for recall followed by a re-ranking process optimized for precision [17]. For efficient filtering, we propose a "fuzzy partition HS histogram" to retrieve a subset of near-duplicate candidate images. After that, we re-rank the top retrieved results by extracting their SIFT features. To evaluate the performance and quality of our hybrid approaches, we provide results of a comparative performance analysis using the original SIFT-128D, the HS color histogram, the fuzzy HS model (F-HS) and the proposed fuzzy partition HS model (FP-HS) and our hybrid approaches F-HS-SIFT and FP-HS-SIFT using large scale image benchmark databases. The results show that applying the hybrid model FP-HS-SIFT, i.e. the F-HS HS model and re-rank the top results (only 6%) of the retrieved images using the SIFT algorithm, significantly outperforms the use of the individual methods.

6.1 Limitation of the Recent Work

As explained in Section 2.3, keypoints detectors and descriptors extract the features after transforming images to gray-scale space. Therefore, the idea of building color descriptor has been proposed in [33], [171] to improve the performance of the detected keypoints (SIFT) in object detection and image retrieval tasks. We detailed the HSV colored SIFT descriptor [33] and the RGB-SIFT [171] in Subsection 3.2. However, the building of the colored SIFT descriptors is time and memory-consuming since

the most methods produce 3×128 dimensional descriptors, which require longer time than the SIFT descriptor to complete the matching process.

6.2 Purpose of Combining Keypoint and Color Features

Keypoint features, specifically SIFT keypoints, have been introduced in most image near-duplicate retrieval researches due to their invariant properties against various kinds of image transformation and their robustness to blurring and adding noise to images (as presented in Appendix B). It has been proven that the SIFT descriptors outperform several kinds of local low dimensional descriptors [130], [129], [128]. The SIFT features have high dimensional descriptors (the dimensionality of each descriptor is 128). The extraction and matching of such high dimensional features for large scale image benchmarks are time and memory-consuming. The suggested methods to build shorter descriptors (i.e. SIFT $96D$ [102], $64D$, PCA$-$SIFT [99] and RC-SIFT [13]) and accelerate the matching process still time-consuming comparing with low dimensional features. In addition, these methods perform at most like original SIFT [102], [99], [13], [15].

On the other hand, color spaces provide low dimensional features. Specifically, the Hue, saturation and value (HSV) color space outperforms various color models such as RGB and L*a*b* [32], [98]. The HSV color space simulates the function of the receptive field in the retina [29], [158] and is produced by combining the three channels of the RGB color space. The histograms produced by color spaces require lesser time and memory than the keypoints to complete the extraction and matching process. However, they reduce the performance in the set of top retrieved results.

From the previous discussion, we inspired our hybrid approaches F-HS-SIFT and FP-HS-SIFT. We suggest to combine the advantages of HSV color space and SIFT keypoints. Our hybrid approaches reduce the required memory and expedites the process of feature extraction and matching. Moreover, they improve the performance of near-duplicate and zoomed-in image retrieval tasks. The idea of our hybrid approach F-HS-SIFT is to construct the fuzzy HSV color model for all benchmark and query images. After that, retrieve images based on the fuzzy HSV color model. Finally, apply the SIFT algorithm to re-rank the top subset of results.

Since color spaces are easily affected by any image changes (i.e. change in viewpoint, illumination, noise, and blur), we propose the hybrid FP-HS-SIFT approach by dividing each image into sub-images before applying the fuzzy HSV model. After that, we utilize the fuzzy HSV model on each sub-image separately. The motivation for this dividing is to improve the retrieval performance of the fuzzy HS color model. To complete the building of our hybrid approach, we re-rank the top set of the retrieved images using SIFT keypoints. Hence, no need to match the SIFT features of a query with all once of a benchmark. We compare the SIFT keypoints only with the top N retrieved images and reduce the memory and time for SIFT keypoints matching step. The reasons for applying the fuzzy HS color model first

are to speed up the process of ND-image retrieval and at the same time retrieve the most near-duplicate images in the top N results [17]. The reverse combination, i.e. applying the SIFT algorithm first and then the proposed fuzzy HSV model, decelerates the feature extraction and matching processes, since we need to compare the SIFT keypoints of each query image with all keypoints of a benchmark.

6.3 Fuzzy HSV & Fuzzy Partition HSV Histograms

Color features have been used in many researches to improve image retrieval performance. In [80] the fuzzy color histogram has been introduced and built utilizing the three channels of the RGB histogram. The idea in [80] is to compute the distance between the color value of each pixel and the centers of histogram bins. Hence, each pixel contributes to all histogram bins.

We build the fuzzy color histogram employing the triangular interpolation described in [115]. We select the triangular interpolation since it outperforms the crisp, cosine and spline interpolations [115]. The following subsections detail our method in building the fuzzy HSV histogram (F-HSV) for all pixels in an image (Subsections 6.3.1 and 6.3.3), improve the retrieval performance by dividing each image into sub-images and construct the fuzzy partition HSV histogram (FP-HSV) (Subsection 6.3.2).

6.3.1 Fuzzy HSV Histogram (F-HSV)

The HSV color space is produced by merging the three channels of RGB color space to get hue, saturation, and value channels (HSV). Hue defines the type of color and it belongs to the range $[0^o, 360^o]$. Saturation describes the pureness of color and value describes the amount of light in a color. The values of saturation and value belong to the range $[0, 255]$ [32]. To build the $3D$ HSV color histogram, the values of each channel are clustered using fix number of clusters. These clusters build the bins of the HSV color histogram. Hence, the hue channel is clustered into 30 bins and each of Saturation and value channels into 32 bins. This way of building clusters is called crisp interpolation i.e., each sample color belongs to only one bin in each channel. Consider c_k; $k = 1, ..., L$ are the centers of clusters, where L is the number of clusters, and r the radius of cluster. The crisp interpolation clustering is defined as:

$$p(x|c_k) = \begin{cases} 1, & \text{if } |x - c_k| \leq r \\ 0, & \text{otherwise} \end{cases} \tag{61}$$

Since the clusters here stand for the bin of the HSV histogram, all clusters have the same radius r. To build the fuzzy clusters, we propose the following steps to involve the triangular interpolation:

- for each sample x, determine the cluster where the sample belongs to it using the crisp clustering.

- compute the absolute distance between the sample and all centers c_k; $k = 1, ..., L$.

- assign the probability $1 - \frac{d_k}{2r}$ to the cluster c_k where $d_k = |x - c_k| \leq r$. Based on the location of a sample to the center of cluster, assign the probability $\frac{d}{2r}$ to c_{k-1} or c_{k+1} i.e.:

$$\begin{aligned} p(x|c_k) &= 1 - \frac{d}{2r} \\ \text{if } x \leq c_k\text{: } p(x|c_{k-1}) &= 1 - \frac{1-d}{2r} \\ \text{else if } x > c_k\text{: } p(x|c_{k+1}) &= 1 - \frac{1-d}{2r} \end{aligned} \tag{62}$$

- for samples that belong to the first cluster and satisfy $d_1 = |x - c_1| \leq r$, we assign the both probability computed in Equation (62) to the first cluster.

- the same thing for the last cluster i.e. the contributions assign only to the last center for samples that belong to the last cluster and satisfy $d_L = |x - c_L| \geq r$.

- repeat the previous steps for all pixels of an image and all channels of the $3D$ HSV histogram.

- normalize the clusters of F-HSV using the area of the input image.

Figure 42 explains the difference between crisp clustering and fuzzy clustering in the case of having four clusters.

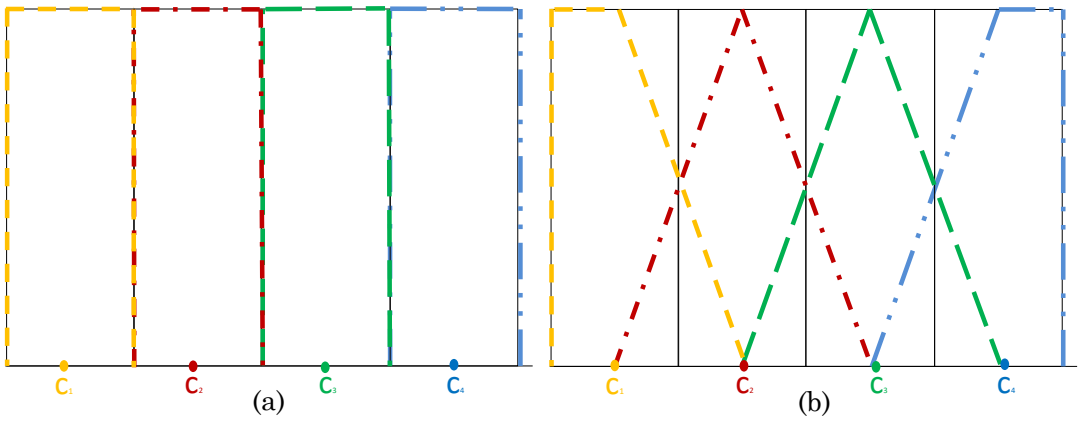


Figure 42: Comparison of crisp clusters (crisp interpolation) and fuzzy clusters (triangular interpolation) of histogram bins. (a) presents the crisp histogram and (b) shows the fuzzy histogram.

6.3.2 Fuzzy Partition HSV Histogram (FP-HSV)

To improve ND and zoomed-in image retrieval by employing the F-HS histogram, we suggest dividing each image into a set of sub-images P. After that, we compute the F-HS histogram for the whole image and for each sub-image as described in Subsection 6.3.1. The set of all F-HS histograms for all sub-images builds the fuzzy partition HSV histogram (FP-HSV). The FP-HSV improves the performance of image ND and zoomed-in image retrieval because it presents additional information about the distribution of colors in images. Moreover, FP-HSV minimizes the retrieval of non-relevant images produced by a similar ratio of colors (but different distribution) between images.

6.3.3 Construction of $2D$ Fuzzy Hue-Saturation Histogram (F-HS)

In our practice study, we noticed that the third dimension of HSV histogram, that is the value dimension decreases the performance of ND-image retrieval. The value dimension measures the amount of lightness in color so that, any change in the lightness of colors causes a big change in the bins of value dimension. Therefore, using only hue and saturation dimensions to build $2D$ histogram is more robust to lightness change than the $3D$ HSV histogram. Based on this notification, we consider only the hue and saturation dimensions when we construct the color histogram i.e., we construct only the fuzzy hue-saturation histogram F-HS and the fuzzy partition hue-saturation histogram FP-HS histograms instead of F-HSV and FP-HSV. We present in Section 6.6.1.1 (Table 27) the comparison between F-HSV and F-HS histograms. The comparison results show that the performance of F-HS is better than the performance of F-HSV in solving the ND-retrieval tasks.

6.3.4 Histogram Similarity Measures

Many methods have been proposed to measure the similarity between two color histograms such as intersection, Chi Square, correlation, and Earth moverś distance [121]. The basic idea of correlation measure is to compare the distribution of two histograms instead of the bin to bin comparison. Therefore, we use in this work the correlation measure by computing the mean μ and standard deviation σ over all bins. After that, we employ the values of μ and σ to compare two histograms. The correlation $Corr$ between two histograms H_1 and H_2 is defined as [121]:

$$\begin{aligned} Corr(H_1, H_2) &= \frac{\sum_{i=1}^{L}(H_{1,i}-\mu_1)(H_{2,i}-\mu_2)}{\sqrt{\sum_{i=1}^{L}(H_{1,i}-\mu_1)^2 \sum_{i=1}^{L}(H_{2,i}-\mu_2)^2}} \\ \text{where } \mu &= \frac{1}{L}\sum_{i=1}^{L} H_i \end{aligned} \tag{63}$$

The values of $Corr$ belong to the rang $[-1, +1]$. The value of -1 means that there is no correlation between the histograms. Whereas the value $+1$ indicates that the histograms are identical. The complexity of this measure is $O(n)$ [121].
The correlation between two fuzzy partitions HS histograms of two near-duplicate images FP-HS_1 and FP-HS_2 is defined as:

$$Corr(FP\text{-}HS_1, FP\text{-}HS_2) = \frac{Corr(F\text{-}HS_1, F\text{-}HS_2)}{P+1} + \frac{\sum_{i=1}^{P} Corr(F\text{-}HS_{1i}, F\text{-}HS_{2i})}{P+1} \tag{64}$$

In the case of zoomed-in / whole scene retrieval, we compute the correlation between the F-HS_z of the zoomed-in image and both F-HS_w and the set FP-HS_{wi}; $\{i = 1, ..., P\}$ of a whole scene. After that, we measure the correlation between the zoomed-in and whole scene images as the average of the highest two correlations. Equation(65) describes the average correlation in case of zoomed-in / whole scene retrieval, where max_{zw1}, max_{zw2} are the biggest two correlations.

$$\begin{aligned} avg\text{-}Corr(z, w) &= \frac{avg(max_{zw1}, max_{zw2})}{2} \\ &\text{where } \{max_{zw1}, max_{zw2}\} = max\{Corr(F\text{-}HS_z, F\text{-}HS_w), \\ &\quad \text{and } \{Corr(F\text{-}HS_z, F\text{-}HS_{wi});\ i = 1, ..., P\}\} \end{aligned} \tag{65}$$

6.3.5 Complexity of F-HS and FP-HS

We compared the computation time of the traditional HS, F-HS and FP-HS to build the color histograms using the Ukbench benchmark (the details of this benchmark are described in Subsection 6.5.1). Table 26 shows that the F-HS and FP-HS require longer time to generate their histograms than the crisp HS. However, F-HS and FP-HS significantly improve retrieval task (see Subsection 6.6.1.1) comparing to the HS histogram. Moreover, F-HS and FP-HS still too faster than the SIFT algorithm (which needs hours to complete the features extraction for the same image dataset). In addition, F-HS and FP-HS produce lesser amount of features than the SIFT algorithm. Hence, they accelerate the matching process too.

Table 26: Time computation of HS, F-HS and FP-HS histograms employing the Ukbench dataset which contains four near-duplicate images for each scene and 10200 images. The consuming time of FP-HS is presented when each image is divided into three sub-images ($P = 3$) or into nine sub-image ($P = 9$).

Method	HS	F-HS	FP-HS	
Sub-images	-	-	$P = 3$	$P = 9$
Duration (Sec.)	151	273	381	530

6.4 Hybrid Approaches

To accelerate and improve the retrieval performance of ND- and zoomed-in images, we proposed our hybrid approaches F-HS-SIFT and FP-HS-SIFT by applying the fuzzy color histogram first. Afterward, re-rank the results utilizing their SIFT keypoints. The following Subsections details our proposed method that are published in [17].

6.4.1 SIFT Features Extraction

In this work, we present the effect of using the F-HS and FP-HS in improving the performance of near-duplicate and zoomed-in image retrieval. Therefore, in the step of extracting SIFT keypoints, we do not discuss the optimized SIFT methods [15], [14], [13], [102] rather, we apply the original SIFT algorithm [116] to extract the keypoints. The keypoints are extracted practicing gray-scale color space and have 128 dimensions. To match the keypoints, we utilize the k-d tree and the best-bin-first algorithm as described in [116]. However, this method obtains duplicate matches i.e. a keypoint of one image may match with several keypoints of the second one. To overcome this problem, we eliminate all duplicate matches except the one which has the best matching score (which is computed employing Euclidean distance as described in Subsection 2.5.1). This filtering of keypoint matches is important to reduce the number of mismatched features. Further discussion to filter the matched features has been presented in Section 5.2 [16].

6.4.2 Re-ranking the Top N Results

To optimize the ND-retrieval results obtained by F-HS and FP-HS, we applied the SIFT algorithm on the top N retrieved results to build F-HS-SIFT and FP-HS-SIFT, respectively. Hence, no need to compare the SIFT keypoints of a query image with all ones of a benchmark. Alternatively, we compared the keypoints of a query image with only the top N results, where $size(N) << size(Dataset)$. In the Section 6.5, we discuss the suitable values for the top N retrieved results. Figure 43 presents the retrieval system steps with the focus on the steps, that we improve in this chapter. Figure 44 details our hybrid approach FP-HS-SIFT [17].

6.5 Benchmark and Evaluation Measures

We compared the performance of the F-HS and FP-HS to the original HSV and HS color models to solve the near-duplicate and zoomed-in image retrieval tasks. After that, we applied the SIFT algorithm on the top N retrieved results to improve the ranking of the retrieved results. For our experiments, suitable benchmarks to solve image near-duplicate and zoomed-in retrieval tasks are employed and described in Subsection 6.5.1. The evaluation measures are discussed in Subsection 6.5.2.

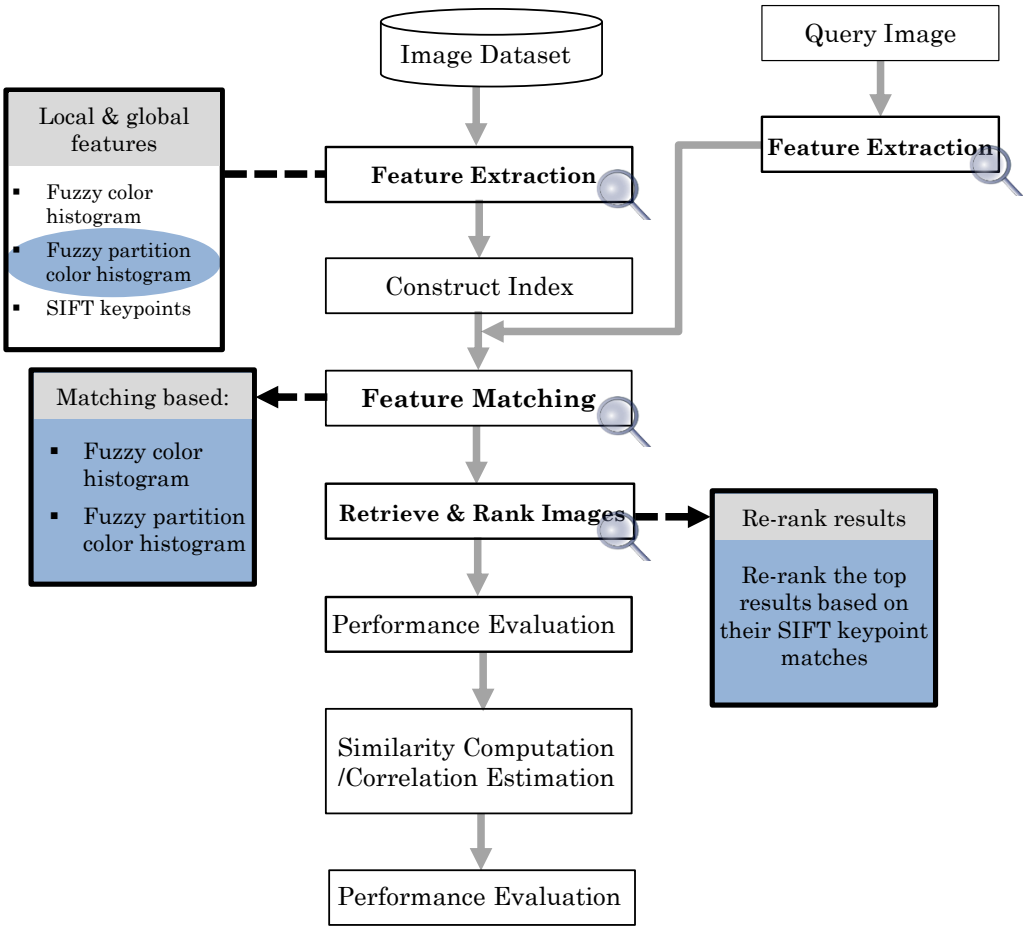


Figure 43: Combine local and global features and re-rank to improve near-duplicate retrieval. The zoomed stages present our focus in this chapter.

6.5.1 Benchmarks

In this work, since we solve two tasks (i.e. near-duplicate and zoomed-in image retrieval), we selected two image suitable Benchmarks. The first is the UKbench [136] benchmark, which fits the ND-retrieval tasks. The second is the Oxford building Benchmark, which is employed to solve the zoomed-in image retrieval task [17].

6.5.1.1 UKbench Benchmark This Benchmark is suitable to solve the ND-image retrieval task. It contains 10200 images of 2550 various scenes i.e. for each, four near-duplicate images. To use this Benchmark, we picked the first image of each scene as a query and kept the rest three in the Benchmark to be retrieved. Hence, we obtain 2550 query images. The details of the UKbench dataset are described in Section 4.2

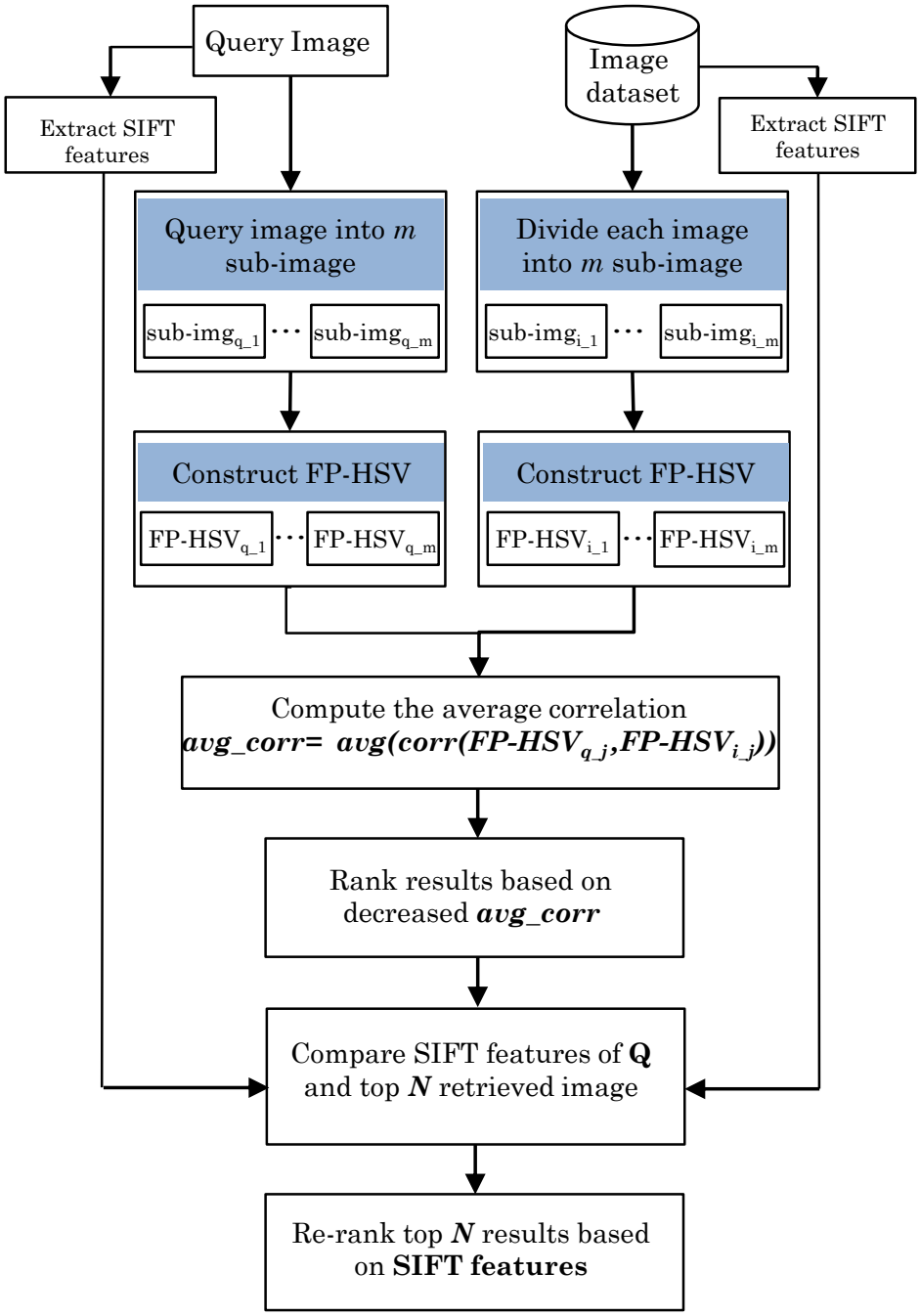


Figure 44: Flowchart of our proposed hybrid model to reduce the required time and memory of feature matching step. Furthermore, to improve the performance of near-duplicate image retrieval.

6.5.1.2 Oxford Buildings Benchmark This Benchmark [143] contains images of the same sight but not necessarily the same scene i.e. they present scenes of inside and outside (details are given in Section 4.4). To use this Benchmark to

solve the zoomed-in image retrieval task, we generate three sets of zoomed-in images by cropping and rescaling the oxford building Benchmarks. These Benchmarks are Oxford-Zoomed-in-50, Oxford-Zoomed-in-25, and Oxford-Zoomed-in-10. The zoomed-in images cover 50%, 25% and, 10% of the original scene, respectively. We use these three constructed Benchmarks as query images to solve the task of whole scene retrieval (original images of oxford buildings).

6.5.2 Evaluation Measures

To evaluate the performance of the proposed F-HS, FP-HS histogram and re-ranked results by the SIFT algorithm, we computed the mean recall (MR) to determine the relevant and retrieved images. To specify the positions of relevance in the retrieved results, we calculated the mean average precision (MAP). To check the distribution of the individual recalls around the average recall we computed the variance of recall (VR) as described in Section 2.5.2 and in [13] and [16] and [17].

6.6 Results and Analysis

6.6.1 Results for Near-duplicate Retrieval Task

We evaluated the performance of the HSV, HS, SIFT and the proposed F-HS, FP-HS and our hybrid approaches F-HS-SIFT and FP-HS-SIFT to solve near-duplicate retrieval tasks. We published most of the results in [17]. In the following subsections, we discuss the results.

6.6.1.1 Comparison of F-HSV & F-HS with Original HSV & HS As mentioned in Subsection 6.3.3, using $2D$ HS histogram performs better than the $3D$ HSV histogram. To justify this idea, we estimated the performance of the crisp HSV and the F-HSV against the crisp HS and the F-HS in solving ND-retrieval tasks. We applied this experiment on the Ukbench Benchmark (see Subsection 6.5.1). We constructed the crisp F-HS histogram as given in Equation (61) and the correlation between two crisp F-HS histograms as given in Equation (63). We built the F-HS as clarified in Equation (62). After that, we computed the correlation as defined in Equation (63). We computed the mean recall, mean average precision and variance of recall using the top three results instead of the first result because the Ukbench benchmark contains three near-duplicate images for each query. Table 27 shows that the crisp HS and F-HS perform better than the crisp HSV and F-HSV in solving the ND-image retrieval task. It presents that the crisp HS and the F-HS obtain better mean average precision than the crisp HSV and the F-HSV models. Moreover, it describes that the variance of recall produced by crisp HS and the F-HS is smaller than the variance of recall obtained by the crisp HSV and the F-HSV.

Table 27: Comparison of the retrieval performance of crisp HSV, crisp HS, F-HSV and F-HS methods using the Ukbench Benchmark. The comparison is done by computing MR, MAP and VR considering the top 3, 10 and 500 results for MR and only the top 3 and 10 for MAP and VR.

Method	MR3%	MR10%	MR500%	MAP3	MAP10	VR3	VR10
HSV	34.87	44.44	82.38	31.40	33.52	11.65	16.02
HS	38.49	48.01	87.52	35.57	36.47	12.01	13.81
F-HSV	37.09	47.76	86.38	33.40	35.02	11.19	15.65
F-HS	41.87	51.62	87.52	37.32	40.39	12.07	13.28

6.6.1.2 Results of the F-HS and FHS-SIFT Approaches As shown in Table 27, the F-HS outperforms the crisp HSV, the F-HSV and the HS models. Therefore, we re-ranked the results of the F-HS utilizing their SIFT features. We suggest re-ranking the top 500 retrieved images because F-HS retrieves about 87% of the relevant images within the best 500 retrieved results (see Table 27).

Table 28 presents that the hybrid approach F-HS-SIFT (i.e. applying F-HS and then re-rank the top 500 results) obtains better results than the extraction and matching of only SIFT features to solve ND-image retrieval task. Moreover, our hybrid approach accelerates the matching process since it compares the SIFT keypoints of a query image with only the top 500 results (i.e. with only 6.5% of the total Benchmark size). Whereas applying only the SIFT algorithm requires matching each query with all benchmark images. Table 28 describes that the hybrid model obtains better mean average precision than the SIFT algorithm. The variance of recall of the hybrid model is lesser than the one of the SIFT algorithm.

Table 28: The retrieval performance of the SIFT algorithm and the hybrid model using the Ukbench Benchmark. The comparison is presented by computing the MR, MAP and VR considering the top 3, 10 results and 50 results for MR.

Method	MR3%	MR10%	MR50%	MAP3	MAP10	VR3	VR10
SIFT	49.32	54.31	58.70	47.46	51.07	15.08	15.17
F-HS-SIFT	**53.41**	**58.22**	**61.12**	**51.70**	**54.80**	**15.12**	**14.57**

6.6.1.3 Results of the FP-HS Approach To evaluate the performance of the FP-HS in solving the ND-retrieval task, we constructed the FP-HS and F-HS for all images in the Ukbench benchmark. After that, we computed the correlation between query and benchmark images using Equation (64). Table 29 presents the performance of FP-HS to solve the ND-retrieval task when both query and benchmark images were segmented into three ($P = 3$) and then nine ($P = 9$) sub-images. The results explain that the use of nine sub-image improves the mean recall and mean average precision of the FP-HS. Moreover, as presented in Table 29, the FP-HS produces

small variance of recall, when nine sub-images are used. Hence, applying the hybrid model to re-rank the retrieved images outperforms the utilizing of the F-HS, SIFT or FP-HS separately. Table 29 describes that the mean recall obtained using nine sub-images is around 80% at the top 50 results and more than 90% at the 500 best results. Therefore, we can improve the performance of image near-duplicate retrieval by re-ranking the top 50 (i.e. 0.65%) or top 500 results (i.e. 6.5%).

Table 29: The retrieval performance of the FP-HSV histogram using the Ukbench benchmark. The results are presented using three ($P = 3$) and nine sub-images ($P = 9$). MR, MAP and VR are displayed for the top 3, 10, 50 and (500 only for MR) retrieved images.

FP-HS	MR3	MR10	MR50	MR500	MAP3	MAP10	VR3	VR10
$P = 3$	52.27	59.32	73.53	89.32	46.74	50.39	13.42	15.73
$P = 9$	**59.71**	**66.22**	**78.44**	**91.32**	**54.68**	**58.14**	**13.17**	**14.33**

6.6.1.4 Results of the FP-HS-SIFT Approach After applying the FP-HS model to retrieve the near-duplicate images, we improved the results ranking by applying the hybrid approach FP-HS-SIFT at the top 300 images. Table 30 introduces the re-ranked results of FP-HS using the SIFT keypoints. The hybrid model FP-HS-SIFT obtains the best mean recall, average precisions and variance of recall when nine sub-images are utilized to construct the FP-HS. The FP-HS-SIFT model using nine sub-images improves the retrieval of ND-images by 22% more than the F-HS-SIFT model (i.e. without segmentation), 30% more than F-HS (see Tables 28, 29 and 30) and 4% more than the FP-HS-SIFT model using three sub-region. The partition of images into smaller sub-images is once again time and memory consuming. Therefore, we did not resume the evaluation for more sub-images. Figure 45 presents a comparison between the SIFT algorithm, the F-HS model, the FP-HS-SIFT approach. The SIFT algorithm and the F-HS model retrieve only one of the three relevant images (in the UKbench benchmark there are three near-duplicate images for each query one) in the top three results. The FP-HS retrieves two of the relevant images at top results. However, re-ranking the results employing the SIFT keypoints (i.e. the hybrid FP-HS-SIFT approach) obtains all relevant results at the top three results.

6.6.2 Results for Zoomed-in Image Retrieval

Zoomed-in image retrieval is part of near-duplicate image retrieval when the zoomed-in image covers the most important in the whole scene. The whole scene / zoomed-in image correlation identification is difficult even for the human visual system when the zoomed-in image covers only a small part of the whole scene and has a different resolution. Therefore, we discuss the case of zoomed-in image retrieval separately. We utilized the Oxford buildings benchmark described in Subsection 6.5.1 to compare

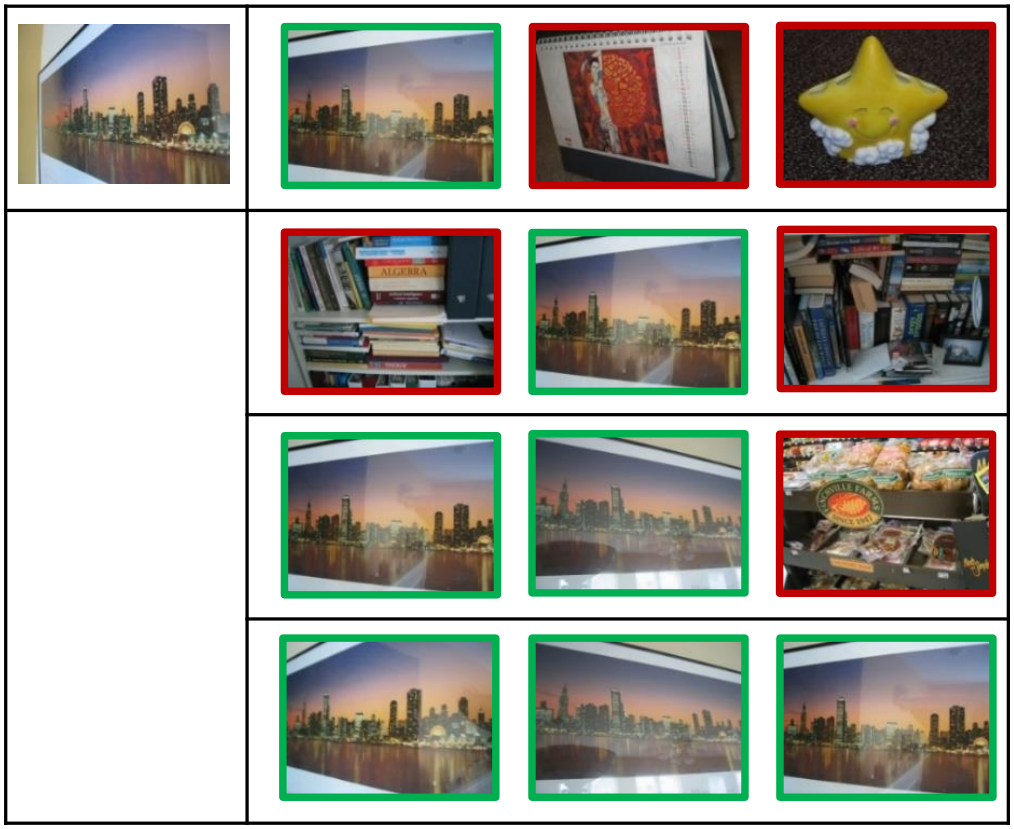


Figure 45: Top three retrieved images for a given query image in the first column. The first row presents the results of the SIFT algorithm, the second of the F-HS model, the third of the FP-HS model and the fourth of the hybrid model. The best results of each method are ranked from left to right. The relevant and retrieved images are marked using green frames. The non-relevant and retrieved results are listed using red frames. The images are from the UKbench benchmark.

Table 30: The re-ranked results of FP-HSV after applying the SIFT algorithm on the top 300 retrieved images. The comparison is presented when three and nine sub-images are used to generate the FP-HS. MR is shown for the top 3, 10 and 50 results.

Hybrid approach: FP-HS−SIFT							
Partitions	MR3	MR10	MR50	MAP3	MAP10	VR3	VR10
$P = 3$	68.52	73.73	75.44	67.12	70.57	13.11	12.05
$P = 9$	**72.77**	**79.04**	**81.79**	**71.06**	**74.41**	**12.64**	**10.82**

the performance of the F-HS, the SIFT algorithm, the FP-HS model, and our hybrid model. In the following paragraphs, all comparisons are detailed to improve the image zoomed-in retrieval task.

6.6.2.1 F-HS for Zoomed-in Image Retrieval We generated the F-HS for the Oxford buildings benchmark and each of Oxford-Zoomed-in-50, Oxford-Zoomed-in-25, and Oxford-Zoomed-in-10 benchmarks as given in Equation (62). After that, the correlation between the zoomed-in images and Oxford buildings dataset was computed as clarified in Equation (63). Table 31 explains that the best performance of F-HS is obtained when for the Oxford-Zoomed-in-50. When the zoomed-in images cover only 10% of the whole scene, the mean recall and the mean average precision of the F-HS model decrease very strongly since the color distribution in the zoomed-in images differs from the one in the whole scene. We do not present the mean average precision for the first result since it is equal to its mean recall. We present the variance of recall and the mean average precision of the top 5 and 50 results.

Table 31: Comparison of the retrieval performance of the F-HS model using the benchmarks: Oxford-Zoomed-in-50, Oxford-Zoomed-in-25, and Oxford-Zoomed-in-10.

Oxford-	F-HS						
Zoomed-in-	MR1	MR5	MR50	MAP5	MAP50	VR5	VR50
50%	**77.63**	**83.86**	**95.41**	**70.93**	**72.64**	**11.41**	**12.41**
25%	49.17	57.13	78.49	37.25	39.35	11.83	13.21
10%	29.93	36.79	59.63	19.58	21.46	14.64	15.21

6.6.2.2 FP-HS for Zoomed-in Image Retrieval We constructed the FP-HS only for the Oxford buildings benchmark using the Equation (62) and three ($P = 3$) and nine ($P = 9$) sub-images. The correlation between the F-HS of zoomed-in images and the F-HS and the FP-HS of whole scenes is computed as defined in Equation (65). We present the performance FP-HS only when $P = 9$ because we focus on comparison for various zoomed-in ratios. Table 32 shows that of FP-HS model obtains mean recall 2%, 14% and 24% better than the F-HS model for all Oxford-Zoomed-in-50, Oxford-Zoomed-in-25 and Oxford-Zoomed-in-10, respectively. As described in Table 32, the best mean recall and mean average precision are obtained for the Oxford-Zoomed-in-50. The variance of recall is small for all benchmarks.

6.6.2.3 Hybrid approach (FP-HS-SIFT) for Zoomed-in Retrieval To improve the retrieval of zoomed-in images, we applied the FP-HS-SIFT approach by re-ranking the top 100 results of the FP-HS model using their SIFT keypoints. The results in Table 33 present an improvement in the performance even when the zoomed-in image covers a small part (i.e. 10%) of the whole scene. The invariance of results is small for all zoomed-in benchmarks. Table 33 shows that the most relevant images appear in the first place (in the case of Oxford-zoomed-in-50 and Oxford-zoomed-in-25) and in the top five results in case of Oxford-zoomed-in-10. The variance of recall produced by the hybrid model is really small when zoomed-in

Table 32: Comparison of the FP-HS model to solve image zoomed-in retrieval task. The number of sub-images is $P = 9$. The MR, MAP, and VR are presented for the top 1, 5 and 50 results using Oxford-Zoomed-in-50, Oxford-Zoomed-in-25 and Oxford-Zoomed-in-10 datasets.

Oxford-	FP-HS						
Zoomed-in-	MR1	MR5	MR50	MAP5	MAP50	VR5	VR50
50%	**79.10**	**85.40**	**96.20**	**66.39**	**66.98**	**16.92**	**3.65**
25%	63.48	73.77	92.02	48.52	50.84	23.94	7.33
10%	53.10	61.44	82.39	39.80	41.99	25.81	14.50

images cover 50% or 25% of whole scenes. Table 33 describes that the mean average precision for the top five and ten results is about the same for all zoomed-in benchmarks.

Table 33: Performance of the hybrid model FP-HS-SIFT to solve the image zoomed-in retrieval task. The number of sub-images is $P = 9$. The MR, MAP, and VR are presented for the top 1, 5, and 10 results of the Oxford-Zoomed-in-50, Oxford-Zoomed-in-25 and Oxford-Zoomed-in-10 datasets.

Oxford-	Hybrid approach: FP-HS−SIFT						
Zoomed-in-	MR1	MR5	MR50	MAP5	MAP50	VR5	VR50
50%	**95.92**	**96.24**	**96.24**	**96.08**	**96.08**	**3.91**	**3.61**
25%	91.56	92.07	92.07	91.80	91.80	7.29	7.02
10%	79.66	81.55	82.06	80.48	80.55	16.20	15.04

6.7 Summary

In this chapter, we proposed our methods to accelerate the extraction and matching of features to improve the process of near-duplicate and zoomed-in image retrieval and hence we solved **RQ.2**. To accelerate the extraction of features and reduce the memory usage, we proposed the F-HS and FP-HS models [17]. The idea of F-HS and FP-HS is to construct the fuzzy $2D$ hue and saturation histograms of an image. To improve the performance of the F-HS and FP-HS models, we proposed our hybrid approaches F-HS-SIFT and FP-HS-SIFT by re-ranking the top N retrieved results using their SIFT keypoints (which are high dimensional features). Hence, we avoid the comparison of the SIFT keypoints of each query with whole benchmarks images. The results characterized that the hybrid approach FP-HS-SIFT (i.e. the combination of the FP-HS and SIFT keypoints) obtains the best performance to solve image near-duplicate and zoomed-in image retrieval tasks. We did not test the reverse combination (i.e. applying the SIFT algorithm first and then re-rank

the results utilizing the F-HS or FP-HS model) because the matching of the SIFT keypoints for all images in a benchmark is time and memory-consuming comparing to the F-HS and FP-HS models.

The focus of this chapter is to describe the correlation between two near-duplicate images without any prior knowledge about their contents. For this, we introduce two deterministic approaches. The first approach (COTA) aims to detect the correlation and filter out false feature matches when one image is a sub-image of another one. The second approach is ECOTA aims to accelerate COTA and increase its capability to detect the correlation between two images in cases such as scale-up / down, rotation, flipping, and overlapping. Regarding these approaches, we figure out the issues presented in ***RQ.3****.*

7 Localization and Transformation Reconstruction of Image Regions

The detection and explanation of the exact correlation between two images is a very important stage in near-duplicate retrieval systems to clarify the ambiguity about the ranking of results. It helps the user to get a better understanding of the process of the retrieval system since it explains and presents the decision with visual arguments. Correlation detection and explanation has many applications such as defining the correlation between an image of a whole landscape and a specific small detailed image of it (this sub-image may be modified too), stitching images into a panorama, determine the transformation between images of the same sights or cities and detection of a copyright infringement cases (i.e. when an image (or only part of an image) is altered and used illegally). The most current approaches to predict the affine transformation between images are RANSAC and PROSAC, which are non-deterministic approaches [126]. These approaches attempt to fit a model to matched features. Therefore, they are biased to the increased amount of false matches and very often estimate wrong transformations by 50% of false matches [53], [30] and their performances start strongly to decrease by 30% of false matches [16], [12]. The random selection of the initial set of matches increases the possibility to estimate variable transformations by repeating the experiment on the same sample, specifically when the initial set includes many false matches. On the other hand, the deterministic approaches proposed in the previous works [82] suffer from rejecting lots of correct matches and consider them as false matches. Due to the huge amount of required iterations by the non-deterministic and deterministic methods, they suffer from time-consuming issues too. Therefore, we introduce in this chapter two approaches, i.e. Congruent Triangle Approach (COTA) and Extended

COTA (ECOTA), that are robust to the number of false feature matches and perform in parallel correlation estimation and false matches filtering, hence, they reduce the required time and memory to assess the correlation between image candidates. Figure 46 presents a flowchart of the main steps of our proposed method to detect the correlation between ND-images. We published our initial approach to estimate the transformation between two ND-images in [16]. In [12] we extended our approach to predict cases such as rotation, flipping and overlapping. The suggested approaches in this chapter address the introduced issues in **RQ.3**.

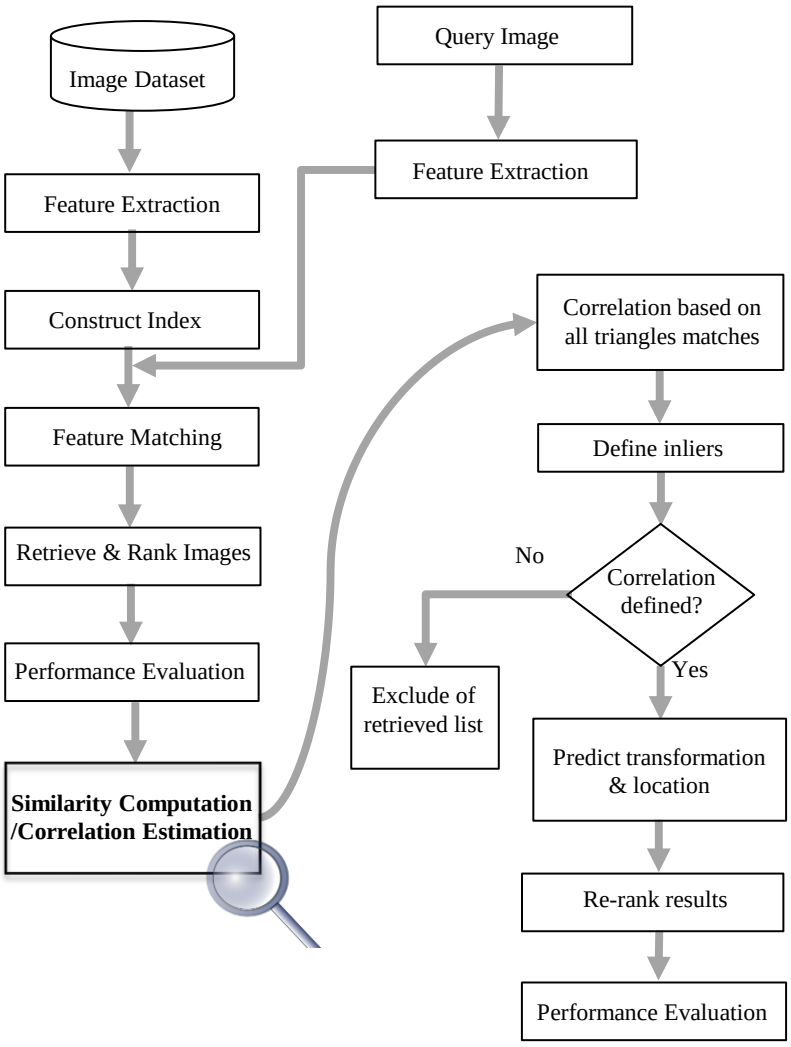


Figure 46: Correlation identification and exclude non-relevant but retrieved images.

7.1 Challenges in Correlation Identification between ND-Images

In order to develop near-duplicate retrieval system the recent researches, three main challenges should be addressed , that are:

- **Amount of Features:** Existing a big amount of not matched features either in both or at least one of the near-duplicate images. These features may influence the correlation detection between images. They are produced by resolution altering, blurring, adding noise, or since one of them is a sub-image of the other one.
- **False Matches:** The substantial increment of false matches affects the correlation detection between images.
- **Too Few Matches:** The correlation detection in many cases impossible when the number of matches decrements dramatically (lesser than five matches) since most of the related methods require more matches to estimate the correlation between two images.

To clarify and explain the correlation between near-duplicate images and overcome the limitation of the state of the arts, we apply the following steps:

- Based on mathematical theories, analyze the relationship between the locations of matched features.
- Employ the detected relationship to split matches into correct (inliers) and false matches (outlier).
- Utilize only the set of inliers to define which of the retrieved images are correlated (relevant) to the query image. Based on these details, exclude the non-relevant images from the list of retrieved images.
- The exact spatial correlation and affine transformation are computed including only the set of inliers.

7.2 Detection & Localization of ND-Image Region

To clarify our hypothesis about correlation detection and localization of near-duplicate images, we carry out a two-stage analysis. In the first stage, we limit our research to the case of sub-image whole scene retrieval. This is a special field of ND-retrieval. It allows determining the sight, panorama, or landscape image where a sub-image belongs to it. To the best of our knowledge, only few researches have been done on this problem using supervised learning techniques. Based on our hypothesis, we aim to detect such relationships without any further details about them and any training phase. In the second stage, we extend our approach to identify the correlation in cases: image / sub-image, flipping, rotation, overlapping, or scale-up/down cases. In addition to the combination of two or three kinds of transformations such as: sub-image & scale-up/down, sub-image & flipping, sub-image & rotation, sub-image

& scale-up/down & rotation or sub-image& scale-up/down & flipping , overlapping & scaled-up/down & rotation, overlapping & scaled-up/down & flipping images. The main contributions of our approaches are computing a set of correct matches based on a robust mathematical theorem, determine the kinds of transformations between ND-images, and identify which of the retrieved images are non-relevant. Since our approaches depend on deterministic basics, they are appropriate even when more than half of the matches are outliers or when only a few matches (lesser than six) are obtained, which is impossible in most of the previous works. Moreover, the extension approach, which we propose, reduces the processing time (comparing to several deterministic and non-deterministic previous approaches) to detect the correlation between feature matches and affine transformation between ND-images.

7.3 Congruent Triangle Approach (COTA)

We introduce a method to retrieve and localize sub-images concerning the whole scene based on correlating groups of feature matches. Matching grouping is employed to filter features that do not contribute to identifying relations between images (outliers). The remaining features (inliers) are employed to estimate the scale altering and location of the sub-image concerning the whole scene. The goal of our approach is to improve the retrieval and localization of sub-images even when a lot of feature matches are predicted as outliers or when only a few matches are detected. Therefore, we demand to identify the correlating group of feature matches and use this group to filter outliers and estimate the scale and location of the sub-image in the whole scene. For this task, the computation costs of the RANSAC and the LMEDS methods are very expensive. Moreover, they fail in estimating the correct relationship or are not qualified to determine the relation when the feature matches include a lot of outliers or when too few matches are detected.

Accordingly, we propose a novel approach to predict outliers and later specify the location of a sub-image in its whole scene even if most of feature matches are outliers or when just a few feature matches are detected. To achieve this, we analyze the spatial distribution of the feature matches between the sub- and whole scene images. In the optimal case, the feature matches build a dense region in the whole scene. However, in real examples, a set of false matches may appear and create other dense regions in the whole scene (see Figure 50(a)). Even when the matches form a dense region, they may include a set of false matches. This is, because of the replicated patterns or textures in the whole scene or sub-image. Hence, we study the correlation between matched features utilizing their location details and by applying the theorem of congruent triangles employing the spatial locations of the feature matches. Therefore, we call this approach COngruent TriAngles Approach (COTA) [16]. COTA employs the geometrical properties of the feature matches to filter them and compute the scale difference between them. This is done by building all potential triangles within feature matches in the sub-image and their

corresponding in the whole scene. After that, COTA verifies the congruent property of the corresponding triangles. Given are sample pair of non-collinear matches with locations P_i, P_j, P_k in image I and their corresponding $P_i^{'}, P_j^{'}, P_k^{'}$ in image $I^{'}$. COTA accepts them as a part of the correlating group if the edges joining these locations satisfy the following relations:

$$\begin{aligned}
&\left| \frac{P_iP_j}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i^{'}P_j^{'}}{max\left\{P_i^{'}P_j^{'}, P_i^{'}P_k^{'}, P_j^{'}P_k^{'}\right\}} \right| < edge_tolerance \\
&\left| \frac{P_iP_k}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i^{'}P_k^{'}}{max\left\{P_i^{'}P_j^{'}, P_i^{'}P_k^{'}, P_j^{'}P_k^{'}\right\}} \right| < edge_tolerance \\
&\left| \frac{P_jP_k}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i^{'}P_k^{'}}{max\left\{P_i^{'}P_j^{'}, P_i^{'}P_k^{'}, P_j^{'}P_k^{'}\right\}} \right| < edge_tolerance
\end{aligned} \tag{66}$$

To apply the idea of the COTA approach on the list of feature matches shown in Figure 47(a), we construct samples of triangles between feature matches. Based on Eq. 66, we detect that one of these feature matches is outlier (presented using red color in Figure 47(b)).

Theoretically, the differences in (66) should be zeros. However, since features are approximately localized, we permit a difference of between the edges of triangles smaller than a predefined threshold *edge_tolerance* between the corresponding edges. We computed *edge_tolerance* in terms of height and width of images as follows:

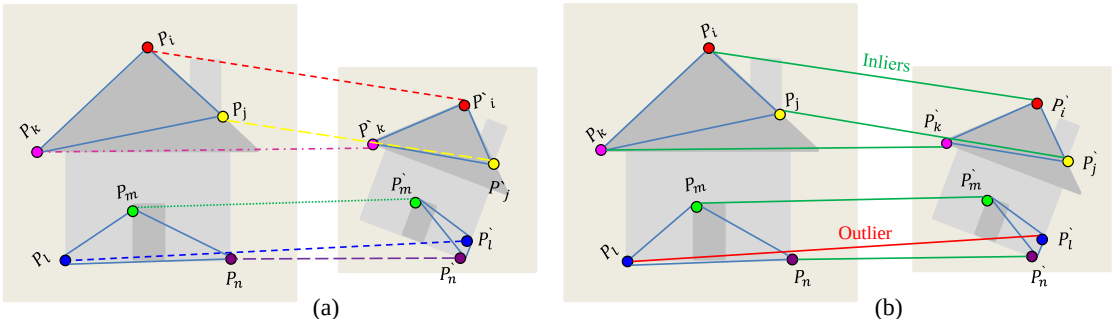


Figure 47: Triangle congruent approach (COTA). (a) presents the methods of triangle construction. (b) clarifies that COTA splits the feature matches into inliers (green lines) and outliers (red lines).

$$edge\_tolerance = MAX(\frac{Log(MAX(W,H))}{MAX(W,H)}, \frac{Log(MAX(W^{'},H^{'}))}{MAX(W^{'},H^{'})}) \tag{67}$$

To avoid the impact of scale and resolution change, we justify that no identification case occur between P_i, P_j, P_k of I or between $P_i^{'}$, $P_j^{'}$, $P_k^{'}$ of $I^{'}$. So that, all matches are accepted that their locations satisfy the following condition.

$$d\left(P_i, P_j\right) > Log(MIN(\frac{W}{2}, \frac{H}{2})) \vee d\left(P_i^{'}, P_j^{'}\right) > Log(MIN(\frac{W^{'}}{2}, \frac{H^{'}}{2})) \tag{68}$$

where H, W and $H^{'}$, $W^{'}$ are the height and width of the whole scene and sub-image, respectively. The steps of COTA are repeated iteratively for all possible combinations of matches. If a specific amount of feature matches (bigger than a predefined threshold *corr_thr*) fulfills relations (66) and (68), then the correlation between matches is confirmed. Otherwise, no correlation can be defined furthermore, the retrieved image is inferred as non-relevant.

7.3.1 Outlier Filtering

After determining the "relevant" images applying the hypothesis of COTA, we process the feature matches again to filter out the false matches i.e. outliers, and to determine the scale difference and location of the sub-image on the whole scene. To complete this step, we compute the scale difference between the sub-image and whole scene as follow:

$$avg_{scale} = \frac{1}{3}\left(\frac{P_iP_j}{P_i'P_j'} + \frac{P_iP_k}{P_j'P_k'} + \frac{P_jP_k}{P_i', P_k'}\right) \tag{69}$$

This computation is accomplished for all construct triangles in sub- and their correspondences in the whole scene. So that, for each pair of matched features a vector of edge ratios is formed. These ratios present scale differences between images. Given are N matched features, COTA constructs N vectors of average scale ratio $V(avg_{scale})$. Each vector contains $C(N,3) = \frac{N!}{3!(N-3)!}$ elements. To filter out the irregular values, the median and absolute deviation around the median (of the values in vectors) are computed as described in [106]. The median appropriates the vectors $V(avg_{scale})$ more than mean since the correct avg_{scale} has almost the highest frequency in $V(avg_{scale})$. The out filtering of irregular values is accomplished as follows: first, the values of $V(avg_{scale})$ are ranked incrementally. After that, the median M of this vector is determined. The median is subtracted of all $V(avg_{scale})$ elements to obtain a vector of Absolute Deviation Scales ($V(ADS) = |M - V(avg_{scale})|$). Let "$MADS$" be the median of $V(ADS)$. Finally, the accepted avg_{scale} values are selected employing the relation:

$$M - 3 \cdot MAD \leq avg_{scale} \leq M + 3 \cdot MAD \tag{70}$$

Where $MAD = b \cdot MADS$ and $b = 1.4826$. To avoid the case of a very small accepted range of values (when $MADS = 0$), a very small number $0 < \beta << 1$ is assigned to $MADS$ when its value is zero. All matched features that their vectors include many irrelevant values (greater than a specific threshold *reject_thr*) are marked as outliers. The remained matches form the correlating group. The relationships between matches in the filtered group are analyzed anew to determine the robust matches i.e. "inliers" that will be the best candidates to estimate the scale difference and location information of the sub-image in the whole scene.

7.3.2 Scale and Location Estimation by COTA

Based only on the set of inliers matches, we compute the exact scale difference $\tilde{S}$ between the sub-image and whole scene as the average of scale difference of the top three pairs of matches that are estimated as inliers using COTA hypotheses. After that, a pair of these top matches $Pi, P_i^{'}$ is pick up and the distance is computed between $P_i^{'}$ and the top-left $(TL^{'})$ and bottom-right $(BR^{'})$ corners of sub-image. The location of the sub-image in the whole scene is defined based on the computed top-left TL, bottom-right BR, width W_{sub} and height H_{sub} as presented in Figure 48 using the following formulas:

$$TL = Pi - TL^{'} \cdot \tilde{S} \tag{71}$$

$$BR = Pi + BR^{'} \cdot \tilde{S} \tag{72}$$

$$W_{sub} = W^{'} \cdot \tilde{S} \tag{73}$$

$$H_{sub} = H^{'} \cdot \tilde{S} \tag{74}$$

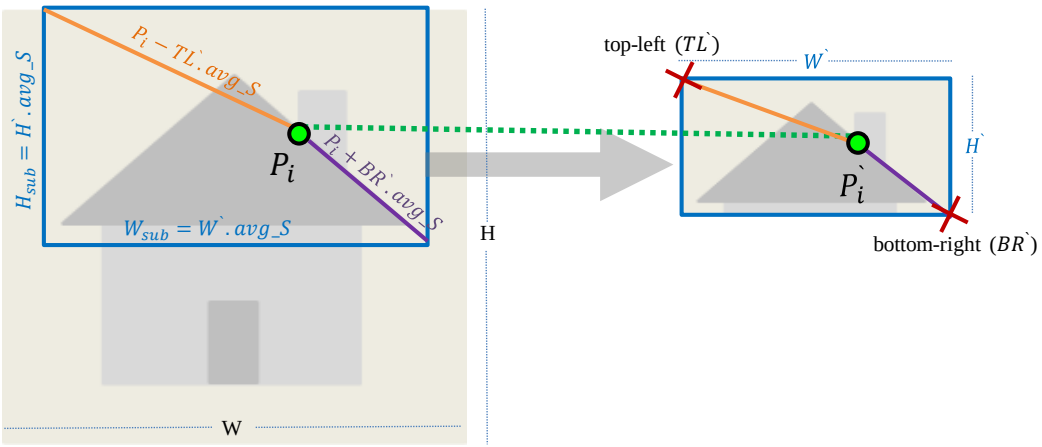


Figure 48: Localization of the sub-image in the whole scene employing COTA.

7.3.3 Evaluation

To evaluate the performance of COTA, we computed the mean recall MR and the mean average precision MAP. To calculate the localization error by COTA, we calculated the relative error of sub-image localization by dividing the offset errors by the length or width of the query image depending on the direction in which the offset occurs. In this way, we avoid the bias of the calculated error to the resolution of the query image. To present the error of localization in pixels, we estimated the absolute error as the sum of offsets in horizontal and vertical directions. After that, we computed the mean of the relative and absolute errors over all images in the employed dataset.

7.3.4 Dataset Description

To verify the robustness of COTA, we constructed our image dataset using panorama images downloaded from [2]. The details of this dataset are given in Section 4.5 and it contains $50,000$ sub-images. We utilized the original 250 panoramas as queries. We reduced the resolution of panorama images too to build four sets of query images, each containing 250 images. The built sets of query images are Q_Scale1, Q_Scale2, Q_Scale3 and Q_Scale4, and they have 60%, 30% 15% and 5% of the resolution of the original panoramas, respectively. Using these settings, we analyzed the effect of scale change on correlation detection and determined which scale is more appropriate to estimate the correlation and localize the sub-images.

7.3.5 Result

The features of both the sub-image dataset and panorama (query) images were extracted utilizing the SIFT algorithm. Since the values of scale and contrast parameters of the SIFT algorithm affect the retrieval process [14], we suggested giving them fix values during all of our experiments. The features were structured by creating the $kd-$tree. The nearest neighbor and Euclidean distance were applied to define the matched features. The resulted images were sorted based on their similarity to the query image. As each query image has 200 relevant images in the dataset, the top 200 ranked images are recommended to be the retrieved set. The Experiments discuss three scenarios: First, specify the effect of scale change of query images on the ranking of retrieved results. Second, estimate and exclude non-relevant but retrieved sub-images by COTA. Third and finally, determine the location of the sub-image in the whole scene and compute the localization error.

7.3.5.1 Down Scale Effect To explain the effect of scale change on the retrieved results, we scaled query images using various rates as explained in 7.3.4. Afterward, we evaluated the retrieved results as described in Subsection 7.3.3. Table 34 presents the results when query images are scaled using various percentages. The results show the best performance is obtained when query images were down-scaled to 30% of the size of the original image. Both mean recall and average precision decrease when query images have the original size or down-scaled to 60% of the original one. The reason is the high resolution, which produces more features and later increases the chance of generating false matches. Moreover, Table 34 reports that the variance of mean recall increases when the scale decrease to cover only 15% or 5% of the resolution of original queries.

7.3.5.2 Exclude non-Relevant Results To filter the results, we applied the relations (66) and (68) of Subsections 7.3 and 7.3.1 on the best 200 retrieved sub-images. Table 35 presents the average of rejected non-relevant and retrieved sub-images. It shows that COTA retrieves the best results for queries with 30%

Table 34: Comparison of retrieval performance by COTA when query images are scaled down using various scale levels. The average mean recall and the average variance of the recall are computed based on the top 200 retrieved images.

Scale down ratio	Mean recall%	Variance of recall%	Average precision%
Q_Scale1: 60%	67.77	1.88	53.36
Q_Scale2: 30%	**80.00**	**1.38**	**67.09**
Q_Scale3: 15%	73.70	2.03	61.86
Q_Scale4: 5%	70.94	2.17	60.43

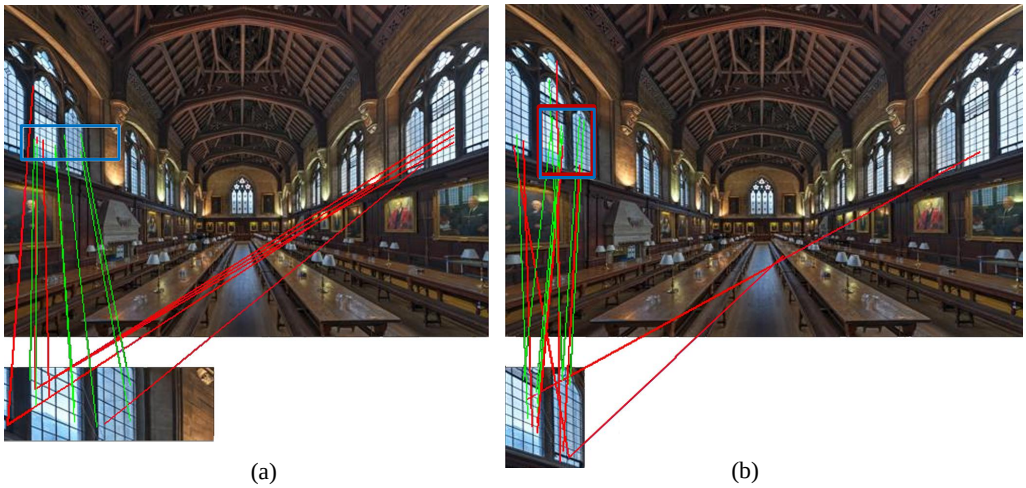
of the resolution of original queries (i.e. Q_Scale2). However, COTA detects all non-relevant results even when the query down-scaled to only 5% of the resolution of the original queries (i.e. Q_Scale4). Furthermore, Table 35 clarifies that COTA identifies a part of retrieved and relevant results as non-relevant. This occurs; since, in some cases, only a few matches are found, which most of them are false matches. COTA obtains the least amount of rejected relevant results concerning query images of set Q_Scale2 and then by Q_Scale1.

Table 35: The performance of COTA to detect non-relevant retrieved images (first row). Through this process, some of the relevant and retrieved results are detected as non-relevant (second row). For this experiment, we set correlating threshold $corr\_thr = 30\%$ of the size of average edge vector (see Subsections 7.3.1 and 7.3.2).

Scale down	Q_Scale1	Q_Scale2	Q_Scale3	Q_Scale4
Rejected non-relevant & retrieved	97.22	**99.96**	99.86	99.82
Rejected relevant & retrieved	7.20	**4.02**	15.16	21.85

(a) (b)

Figure 49: Examples where COTA estimates the correlation group of matches (green lines) and localize the sub-image in the whole scene successfully. Whereas, RANSAC fails to predict the relationship. (a) the detected features are 530 and 35 in the whole scene and sub-images respectively. The total number of matches is five and no outliers are detected by COTA. (b) only three matches are found but COTA estimates them all as inliers and therefore it localizes the sub-image correctly.



(a) (b)

Figure 50: (a) An example where our proposed method (COTA) determines correctly the correlating group of matched features and localized the sub-image in the whole scene (blue box) whereas, RANSAC fails to predict the relationship between these images. In this example, the extracted features in the whole scene and sub-images are 359 and 48 respectively. The total number of matches is 17. Additionally, more than half of matches are detected as outliers (i.e. more than 58% of matches are outliers). (b) An example shows that COTA and RANSAC (red box) models find successfully the sub-image and predict its location in the whole scene. In this example, the detected features are 359 and 70 in the whole scene and sub-images respectively. Out of 16 matches seven are identified as outliers (red lines) i.e. 41% of matches are outliers.

7.3.5.3 Localize Sub-images in Whole Scenes After estimating the correlating group of matches, locations of sub-images were identified as described in Subsection 7.3.2. For our experiment, we employed a rejection threshold $reject\_thr = 25\%$ to exclude irrelevant matches. We compared the performance of COTA with the performance of RANSAC to estimate the correlation between images and localize sub-images in whole scenes. The results describe that our method is more robust than RANSAC in assessing the correlation and location of sub-images in whole scenes. Table 36 presents that the performance of RANSAC decreases faster than the performance of COTA when the resolution of queries decreases like in Q_Scale3 and Q_Scale4. This is, due to the instability of RANSAC to the increment amount of outliers. Figure 49 shows an example where COTA assesses the correlation between the matches and predicts the location of the sub-image successfully even when just five matches are detected. However, these matches are not sufficient to estimate the correlation by RANSAC. Figure 50(a) shows an instance of outliers filtering by COTA. In this case, COTA predicts the correlation and location of a sub-image even when more than half of the matches are marked being outliers. When the amount of known outliers is lesser than the half of matches, RANSAC, as well as our method, estimates the correlation among the correlating group as described in Figure 50(b). Table 37 indicates that the least relative localization error is found by Q_Scale1 next by Q_Scale2. The least absolute error of localization is obtained by

Q_Scale4 and next by Q_Scale2. However, the localization shifting is trivial through all established scales. Consequently, COTA completes the localization successfully, even when query images are down-scaled. We did not calculate the localization errors detected by the RANSAC model because it estimates the homography change between two images based on the correlating group of matches. When the RANSAC model estimates the transformation correctly, we suggested that the retrieved image is localized unless the model predicts no correlation between images.

Table 36: The performance of the RANSAC model versus COTA given below. The results present the rate of exact localized sub-images applying our method and projected images using the RANSAC model.

Scale down ratio	Q_Scale1	Q_Scale2	Q_Scale3	Q_Scale4
RANSAC (%)	83.92	79.33	60.32	42.76
COTA (%)	**95.00**	**94.68**	87.00	73.40

Table 37: Relative and absolute localization errors by COTA when the location of a sub-image is determined in a whole scene.

Scale down ratio	Q_Scale1	Q_Scale2	Q_Scale3	Q_Scale4
Relative error (%)	$\mathbf{1.90 \cdot 10^{-3}}$	$3.09 \cdot 10^{-3}$	$1.05 \cdot 10^{-2}$	$7.15 \cdot 10^{-2}$
Absolute error (pixel)	2.53	2.48	3.12	**1.11**

7.3.6 Summary & Limitations

We introduced a method to improve sub-image retrieval and localization. We achieved this by identifying the correlating group of features between sub- and whole scene images. We computed the correlation by estimating the symmetry of all constructed triangles that are built based on the locations of matched features. Based on the correlating group, we decided whether a sub-image is a part of a given whole scene. After that, we used the correlating group to determine the location of the sub-image in the whole scene without any previous knowledge about the relationship between them. The results characterized that COTA is more robust than RANSAC in assessing the correlation and localize sub-images even when the matches contain many outliers or only a set of few correct matches (three matches) are found. Moreover, COTA supports users with reasonable explanation about the exact correlation (image/sub-image and scale difference) and the reason of reject or accept one of retrieved image that it detected as non-relevant or relevant respectively by COTA. Figure 51 presents an overview with examples of the details of COTA.

COTA has the limitation that it cannot identify the outliers when matches satisfy Equation (66) but they are false matches. Figure 52(a) shows a case where COTA

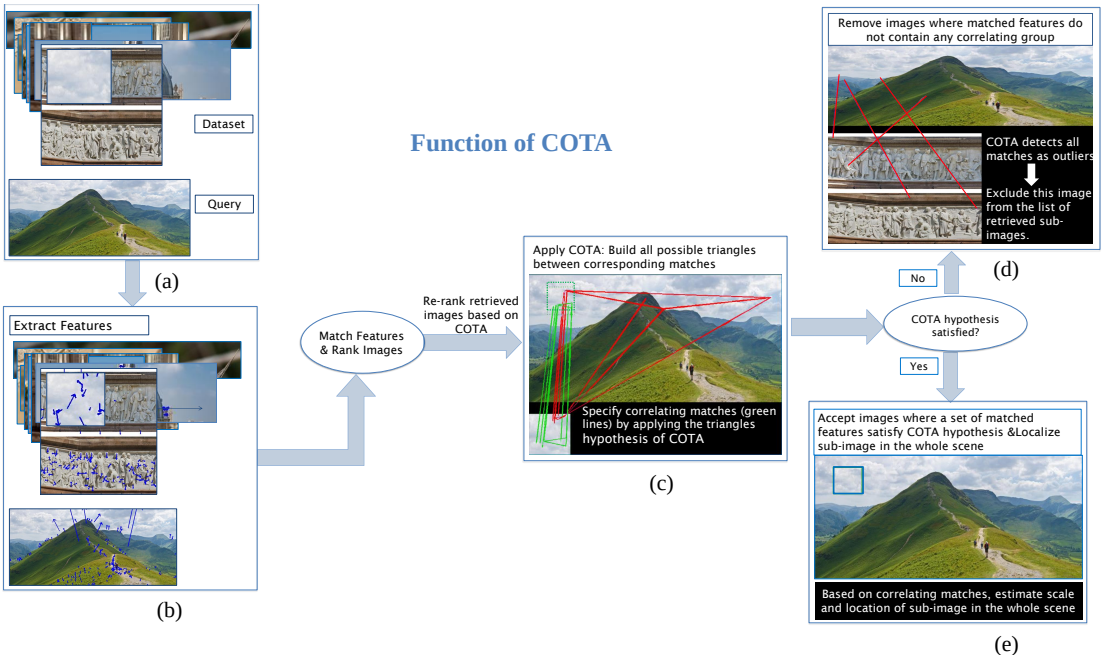


Figure 51: Clarification of COTA function. (a) presents samples of sub- and query datasets. (b) extracted SIFT features from both datasets. (c) green dash box presents the ground truth of sub-image location in the whole scene. Two pair of triangles are drawn, green pair for the correlated matches and red for the false matches. (d) the sub-image is retrieved as "relevant" but COTA detects no correlated matches therefore, this image is excluded of the retrieved results. (e) blue box presents the estimated location of sub-image by COTA.

fails to define all outliers since features P_s, P_t, P_r and P_s', P_t', P_r' fulfill the congruent properties in Equation (66). This explains that COTA cannot detect such cases of outliers even when the lines that they connect these matches are crossed in the $2D$ space. Moreover, COTA fails to determine the correct locations of sub-images in whole scenes when transformations include rotation or reflection. In addition, COTA is time-consuming compared to some of related methods (such as RANSAC and PROSAC) since it processes the matches twice to detect the outliers, define the scale difference and localize the sub-image in the whole scene.

To overcome these restrictions, we present two methods to improve of COTA. These are fourth point COTA (4COTA) and extended COTA (ECOTA).

7.4 Fourth Point COTA

To improve the confidence level of the COTA, we introduce the 4COTA approach [63]. The main contribution of 4COTA is to check the existence of a fourth pair of matches for each corresponding triangles. The first step of 4COTA is to check the condition in Equation (66). If this condition is verified, then 4COTA looks for a fourth pair of matches that are placed inside the corresponding triangles. In the case of fulfilling both conditions, we increase the score of all four matches. This process is iterated for

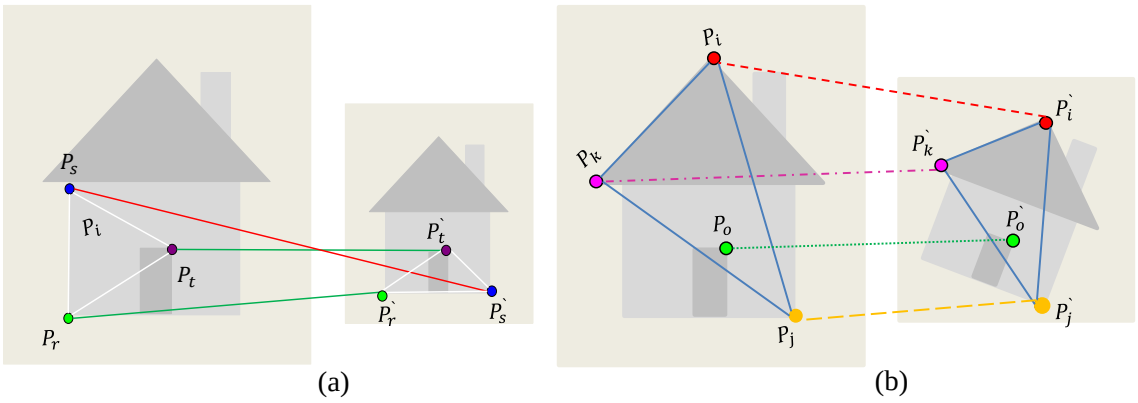


Figure 52: (a) the limitation of COTA. P_s, P_t, P_r and $P_s^{'}$, $P_t^{'}$, $P_r^{'}$ satisfy Equation (66) in spite of the fact that the pair $P_sP_s^{'}$ are outliers. (b) the main idea of 4COTA. For each constructed pair of triangles $P_iP_jP_k$ and $P_i^{'}P_j^{'}P_k^{'}$, a search process is accomplished to find a fourth point P_m inside $P_iP_jP_k$. When P_o is found, the spatial location of its corresponding $P_o^{'}$ is checked. If $P_o^{'}$ is located inside $P_i^{'}P_j^{'}P_k^{'}$ then all scores of all four matches is increased.

all possible triangles. The matches with the highest score are employed to estimate the correlation and to perform the localization. Figure 52(b) illustrates the basic idea of 4COTA practicing a simple example. However, 4COTA is slower than COTA, since it requires looking for a fourth match each time the condition in Equation (66) is confirmed. For a set of N matches the number of iteration is $(N-3)\cdot C(N,3)$ i.e. the increment number of matches affect negatively on the processing time. Moreover, 4COTA is not applicable when the number of matches is lesser than four.

7.5 Extended Congruent Triangles Approach (ECOTA)

As explained in the previous subsection COTA fails to detect the false matches when Equation (66) is satisfied. 4COTA tries to improve the performance by introducing the hypothesis of the fourth pair of matches. However, 4COTA fails to predict the affine transformation in the case of reflection. Moreover, both COTA and 4COTA are time-consuming comparing to some of the previous methods. To overcome these problems, we introduce Extended COTA (ECOTA) [12]. The hypothesis of ECOTA is to connect between the matches employing vectors instead of edges. Mathematically it is known, that vectors have three component i.e. length, orientation and gradient. The suggestion of ECOTA is to exploit the details of orientation and gradient in addition to the length that is utilized in COTA. We aim to exploit the orientation and gradient properties to detect more robust matches and to overcome the problem of COTA explained in Figure 52(a). In addition, the hypothesis of ECOTA determines the correlation in cases of orientation and reflection transformations. As shown in Figure 53(a), we construct vectors $\overrightarrow{P_iP_j}$, $\overrightarrow{P_iP_k}$, $\overrightarrow{P_jP_k}$ in I and $\overrightarrow{P_i^{'}P_k^{'}}$, $\overrightarrow{P_i^{'}P_j^{'}}$, $\overrightarrow{P_i^{'}P_k^{'}}$ in $I^{'}$. In the case of correct matches, the corresponding vectors should satisfy Equation (66) and have the same gradient and orientation. For the matches with locations

P_i, P_j, P_k in image I and their corresponding matches $P_i^{'}, P_j^{'}, P_k^{'}$ in image $I^{'}$, we compute the gradient as:

$$\begin{aligned} m_{ij} &= \frac{y_j - y_i}{x_j - x_i}, \quad m_{ij}^{'} = \frac{y_j^{'} - y_i^{'}}{x_j^{'} - x_i^{'}} \\ m_{ik} &= \frac{y_k - y_i}{x_k - x_i}, \quad m_{ik}^{'} = \frac{y_k^{'} - y_i^{'}}{x_k^{'} - x_i^{'}} \\ m_{jk} &= \frac{y_k - y_j}{x_k - x_j}, \quad m_{jk}^{'} = \frac{y_k^{'} - y_j^{'}}{x_k^{'} - x_j^{'}} \end{aligned} \tag{75}$$

After that, we applied $atan2$ function to compute the angles in range $(-\pi, \pi]$. The function $atan2$ returns an angle in the range $(-\pi, \pi]$ and is computed in terms of $arctan$ function, whose returns an angle in the range $(-\pi/2, \pi/2]$, as follows:

$$atan2(y,x) = \begin{cases} arctan(\frac{y}{x}) & \text{if } x > 0 \\ arctan(\frac{y}{x}) + \pi & \text{if } x < 0 \text{ and } y \geq 0 \\ arctan(\frac{y}{x}) - \pi & \text{if } x < 0 \text{ and } y < 0 \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0 \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0 \end{cases} \tag{76}$$

Hence we computed the angles as follows and as shown in Figure 53

$$\begin{aligned} \varphi_{ij} &= atan2(y_j - y_i, x_j - x_i), \quad \varphi_{ij}^{'} = atan2(y_j^{'} - y_i^{'}, x_j^{'} - x_i^{'}) \\ \varphi_{ik} &= atan2(y_k - y_i, x_k - x_i), \quad \varphi_{ik}^{'} = atan2(y_k^{'} - y_i^{'}, x_k^{'} - x_i^{'}) \\ \varphi_{jk} &= atan2(y_k - y_j, x_k - x_j), \quad \varphi_{jk}^{'} = atan2(y_k^{'} - y_j^{'}, x_k^{'} - x_j^{'}) \end{aligned} \tag{77}$$

In the ECOTA, these matches should satisfy the conditions of COTA (i.e. Equation (66)) and the following condition:

$$\left|\varphi_{ij} - \varphi_{ij}^{'}\right| \leq \vartheta \qquad \wedge \qquad \left|\varphi_{jk} - \varphi_{jk}^{'}\right| \leq \vartheta \qquad \wedge \qquad \left|\varphi_{ik} - \varphi_{ik}^{'}\right| \leq \vartheta \tag{78}$$

Where ϑ is a very small angle produced by inexact localization of features and we set it to be one tenth of degree in radian i.e.

$$\vartheta = \frac{1}{10} \times 1^o \times \frac{\pi}{180} \tag{79}$$

Figure 53(b) presents a case of producing a false match (i.e. the pair $\{P_u, P_u^{'}\}$ is outlier). We apply the hypotheses of ECOTA and compute the angles φ_{uv}, $\varphi_{uv}^{'}$ and φ_{uw}, $\varphi_{uw}^{'}$ (that are produced through the connection between $\{P_u, P_v\}$, $\{P_u^{'}, P_v^{'}\}$ and $\{P_u, P_w\}$, $\{P_u^{'}, P_w^{'}\}$ respectively). We find out that the angles do not satisfy

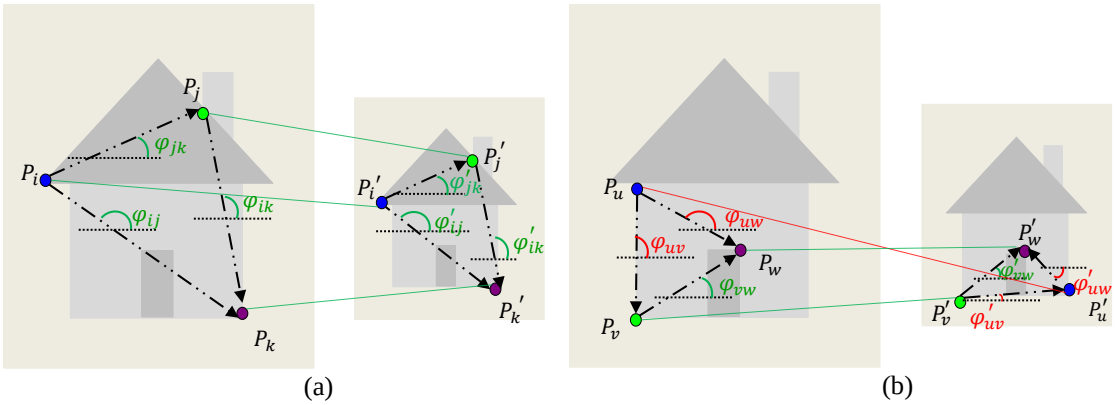


Figure 53: (a) core idea of ECOTA, $\overrightarrow{P_iP_k}$ vectors $\overrightarrow{P_iP_j}$, $\overrightarrow{P_iP_k}$, $\overrightarrow{P_jP_k}$ and their corresponding $\overrightarrow{P_i'P_j'}$, $\overrightarrow{P_i'P_k'}$, $\overrightarrow{P_j'P_k'}$ have identical orientations respectively. (b) direct outliers detection by applying ECOTA (vectors $\overrightarrow{P_uP_v}$, $\overrightarrow{P_uP_w}$ and their correspondence $\overrightarrow{P_u'P_v'}$, $\overrightarrow{P_u'P_w'}$ have different orientations).

Equation (77). More specifically, the angles φ_{uv}, $\varphi_{uv}^{'}$ and φ_{uw}, $\varphi_{uw}^{'}$ are not identical which indicates that $P_u, P_u^{'}$ are outliers. In such a case, the angles between the other two correct pairs of matches i.e. φ_{vw}, $\varphi_{vw}^{'}$ preserve their consistency. This hypothesis is very valuable since it determines directly whether the congruent property is satisfied. Moreover, it defines instantly which pairs of matches disrupt the congruent property. Therefore, by applying ECOTA no need to repeat the process of matches verification twice (like in COTA and 4COTA) to detect the outliers. We call the pair of matches which justify the hypothesis of ECOTA "inliers". The steps of ECOTA are summarized as [12]: We determine min_N in the experiment. The break of "For loop" is after having specific amount of matches in $V_{Inliers}$.

7.5.1 ECOTA and Rotation

To make the ECOTA usable in case of rotation, we extend our hypothesis of angles (Equation (78)), i.e. if the pair of matches satisfy Equation (66) but do not satisfy Equation (78) we check whether the angles verify the following condition:

$$\varphi_{ij} = \varphi_{ij}^{'} \pm \theta \quad \wedge \quad \varphi_{jk} = \varphi_{jk}^{'} \pm \theta \quad \wedge \quad \varphi_{ik} = \varphi_{ik}^{'} \pm \theta \tag{80}$$

where $\theta >> \vartheta$. If the statement in Equation (80) is confirmed, we estimate a difference θ in rotation between images.

7.5.2 ECOTA and Reflection

To detect the reflection transformation, we upgrade once again the statement of angles. For pair of matches that they verify the condition in Equation (66) but do not fulfill the conditions in Equations (78) or (80), we check the following conditions:

$$\varphi_{ij} + \varphi_{ij}^{'} = 0 \pm \vartheta \quad \wedge \quad \varphi_{jk} + \varphi_{jk}^{'} = 0 \pm \vartheta \quad \wedge \quad \varphi_{ik} + \varphi_{ik}^{'} = 0 \pm \vartheta \tag{81}$$

Algorithm 1 ECOTA

Sort the pair of matches based on the best score and store them in a vector V(Matches)
Retrieved $\wedge$ *Relevant* $= 0$
$V_{Inliers} = \{\}$, $V_{Outliers} = \{\}$, $V^{'}_{Inliers} = \{\}$, $V^{'}_{Outliers} = \{\}$
if $(N \geq min_N)$ **then**
  for $i = 1$, $j = i + 1$, $k = j + 1$ to N **do**
    if $(\overrightarrow{P_iP_j} \nparallel \overrightarrow{P_iP_k}) \wedge (\overrightarrow{P^{'}_iP^{'}_j} \nparallel \overrightarrow{P^{'}_iP^{'}_k})$ **then**
      Construct triangles P_i, P_j, P_k & $P^{'}_i, P^{'}_j, P^{'}_k$
      if (Equation (66)) **then**
        compute angles form Equation (77)
        if (Equations (78)) **then**
          $V_{Inliers} \leftarrow \{P_i, P_j, P_k\}, V^{'}_{Inliers} \leftarrow \{P_i, P_j, P_k\}$
        else if (Equations (78) only for one angle φ_{ij}, $\varphi^{'}_{ij}$) **then**
          $V_{Inliers} \leftarrow \{P_i, P_j\}$, $V^{'}_{Inliers} \leftarrow \{P_i, P_j\}$
          $V_{Outliers} \leftarrow \{P_k\}$, $V^{'}_{Outliers} \leftarrow \{P_k\}$
        else
          $V_{Outliers} \leftarrow \{P_i, P_j, P_k\}$, $V^{'}_{Outliers} \leftarrow \{P_i, P_j, P_k\}$
        end if
      end if
      if Size $(V_{Inliers}) \geq thr_{Inliers}$ **then**
        Break For Loop
      end if
    end if
  end for
end if
if $Size(V_{Inliers}) \neq 0$ **then**
  for $m = 1$ to N **do**
    if (P_m not in $V_{Inliers}$) **then**
      triangle $P_mP_jP_k$, $P^{'}_mP^{'}_jP^{'}_k$; $\{P_j, P_k\} \subset V_{Inliers}$ &$\{P^{'}_j, P^{'}_k\} \subset V^{'}_{Inliers}$
    end if
    if (Equations (66) and (78)) **then**
      $V_{Inliers} \leftarrow \{P_m\}$, $V^{'}_{Inliers} \leftarrow \{P^{'}_m\}$
    else
      $V_{Outliers} \leftarrow \{P_m\}$, $V^{'}_{Outliers} \leftarrow \{P^{'}_m\}$
    end if
  end for
end if
if $\frac{Size(V_{Inliers})}{Size(V(Matches))} \geq reject_thr$ **then**
  Retrieved $\wedge$ Relevant $= 1$
end if
if Retrieved $\wedge$ Relevant **then**
  Compute scale difference $\tilde{S}$ and location as given in Equations (71)
end if
return $V_{Inliers}, V_{Outliers}$, Retrieved $\wedge$ Relevant

$$\varphi_{ij} + \varphi_{ij}^{'} = \pi \pm \vartheta \quad \wedge \quad \varphi_{jk} + \varphi_{jk}^{'} = \pi \pm \vartheta \quad \wedge \quad \varphi_{ik} + \varphi_{ik}^{'} = \pi \pm \vartheta \quad (82)$$

If the matches confirm Equation (66) and Equation (82) or (84) then we deduce that their is reflection transformation between images.

7.5.3 Determine the Affine Transformation with ECOTA

As discussed previously, by employing the ECOTA, we decide whether the relationship between images implies an affine transformation such as rotation, reflection, shifting (when images are overlapped), up-scaling, down-scaling or a combination of various kinds of affine transformations. To determine the kind of affine transformation, after justifying the conditions of the ECOTA, we compare the distribution of the matches in both images. If the pair of matches cover a small area of one image but a large area of the other and they satisfy Equation (80) or (82) or (84), we conclude a relationship of zoom-in/out with rotation or reflecting respectively. But if the pair of matches cover about the same area of both images and Equation (77) is justified but the matches are localized in opposite locations with respect to centers of images, then we detail that images are overlapped (affine transformation is shifting). When no of the previous cases is satisfied then images are duplicated but one of them may differ in scale or scale and rotation.

7.5.4 Localization with ECOTA

Given are two images I and $I^{'}$, to localize I based on ECOTA in case of satisfying Equation (77), we compute the ratios in Equation (69) based on the correct matches. If a value of $s \neq 1$ is calculated then, there is scale change equal to s between the images. Based on Equations (80), (82), and (84), the rotation or reflection are computed. We apply the inverse transformation on I to localize it on $I^{'}$.

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix} + \begin{pmatrix} x_c \\ y_c \end{pmatrix} \quad (83)$$

where (x_c, y_c) is the center of image I and (x_r, y_r) the rotated result. From Equation (83) we find:

$$\begin{aligned} x_r &= (x - x_c)cos\theta - (y - y_c)sin\theta + x_c \\ y_r &= (x - x_c)sin\theta + (y - y_c)cos\theta + y_c \end{aligned} \quad (84)$$

When a reflection transformation is detected, the localization is accomplished by replacing each pixel (x, y) with $(W - x, y)$ or $(x, H - y)$ or $(W - x, H - y)$ depending on the kind of reflection (i.e. refleciton to X-axis, Y-axis or origin) where W and H are the width and height of image respectively.

7.6 Evaluation Setting

To evaluate the performance of ECOTA, COTA and 4COTA against RANSAC, PROSAC and LMEDS, we constructed different image benchmarks. In the following, we present the types of near-duplicate images and the benchmarks that we employed in our evaluation.

7.6.1 Near- & Partial-Duplicate Images

In this chapter, we restrict the definition of near-duplicate images as images that they present the same scene but differ in resolution, illumination, adding noise or blur, or one of the following affine transformation i.e. scale change, rotation, flipping or shifting (which cause overlapped images). In this thesis, near-duplicate images have one or a combination of these transformations. Figure 54 presents the cases that they are suggested as near-duplicates. So near-duplicate image can be the same scene but up or down scaled or a sub-image. In both cases near-duplicate image may contain another type of transformation such as rotation or flipping. We produced

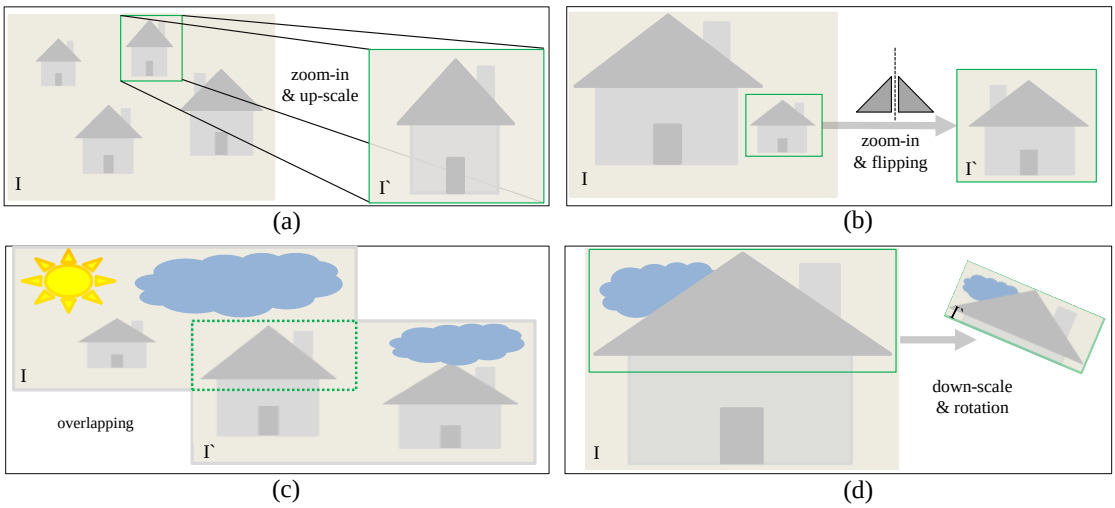


Figure 54: Samples of near-duplicate images that are discussed in this work. (a) sub-image of a panorama with scale change. (b) sub-image with reflection operation and scale change. (c) case of overlapping between two images. (d) sub-image, down-scale and rotation.

various kinds ND images as follows:

7.6.1.1 Same Scene Altering In this case, we produced image datasets that present same scenes as queries, but they differ in scale, orientation or flipping. To create the scaled dataset, we scaled images down and up to cover 15%, 30%, 50%, 100%, 200%, 300% and 500% of the resolution of original images. The goal of scale change is to determine the limits of our method in correlation detection, localization, and explanation. After that, we rotated the set of scaled images around their centers

using the angles 15^o, 30^o, 45^o and 90^o to build image datasets that contain scaled and rotated images. We employed the scaled images once again to produce a flipped dataset. We created it by flipping the scaled images using the horizontal or vertical axes or concerning the origin point. The goal of applying rotation and flipping in addition to scale change is to check whether our proposed method can detect and explain the correlation in such cases.

7.6.1.2 Sub-Image Altering We cropped sub-images of various locations of original image datasets. The cropped images cover areas between 4% and 15% of the original images. We scaled, rotated, and flipped these sub-images using the same setting as in previous Paragraph 7.6.1.1 to build near-duplicate sub-image datasets.

7.6.1.3 Overlapped Images We created the overlapped image dataset to justify the capability of our algorithm in correlation detection and explanation in the case of overlapping. To build this dataset, we cropped sub-images of each original image that overlapped with about 30% of their areas. The produced overlapped images were scaled-up/down and (or) rotated as described in Paragraph 7.6.1.1. Consequently, the overlapped dataset contains overlapped images with various kinds of affine transformations. For each scene, one of the overlapped images is utilized as a query.

7.6.1.4 Adding Blur, Noise or Illumination Change To evaluate the performance of our model to various kinds of image deformation we selected a subset of constructed images using the setting in Paragraphs 7.6.1.1 and 7.6.1.2 and we applied Blur, noise, illumination change, or rotation on them.

7.6.2 Datasets

To compare the performance of our approach ECOTA with the previous once in solving the task of correlation prediction and localization, we constructed five image datasets: scaled panoramas (PANO), scaled Oxford buildings (OXB), aerial scene (Aerial), paintings (PAIN), and mixed affine transformation (ATRANS) datasets. There is a substantial variance between these datasets in scene representation. Therefore, we evaluate whether our proposed algorithm is robust to various types and structures of images. In the following, we explain the details of these datasets.

7.6.2.1 Scaled Panoramas (PANO) Typical samples of this dataset [16] are presented in Figure 55(a). It contains sub-images of 200 panoramas [16]. They are images of landscapes or sights. The details of the PANO dataset are presented in Section 4.5 and it includes $20,000$ sub-images. Figure 55(a) presents a sample of this dataset.

7.6.2.2 Scaled Oxford Buildings (OXB) The Oxford building dataset includes 55 different sights and objects of Oxford. The resolution of images in this dataset is 1024 in width or height [143]. We randomly picked a subset of 500 images of this dataset and apply the same setting as in PANO to build our scaled Oxford buildings (OXB). This dataset includes $50,000$ and is presented in Section 4.4. Figure 55(b) shows a sample of this dataset.

Figure 55: Sample of image datasets. (a) PANO dataset, (b) OXB dataset, (c) Aerial dataset and (d) PAIN dataset

7.6.2.3 Aerial Scene (Aerial) The aerial dataset is a subset of a published aerial benchmark called AID [185]. Aerial dataset contains $20,000$ sub-images. The details of this dataset are described in Section 4.6. A sample of them is shown in Figure 55(c). The structure of aerial images differs from panoramas, Oxford buildings, and paintings since they are taken from far distances and cover big scenes. Therefore, we discuss correlation detection in this case separately.

7.6.2.4 Paintings (PAIN) This dataset is a small subset of the free accessible dataset "Your Paintings" [1]. The details of this dataset are presented in Section 4.7. A sample of this dataset is presented in Figure 55(d). This dataset has been used in [51] to discuss the problem of object retrieval in paintings. In our research, we employed it to check whether our proposed method can estimate the correlation in the case of paintings. PAIN dataset contains $36,000$ sub-paintings.

7.6.2.5 Mixed Affine Transformation (ATRANS) We constructed a mixed dataset using the panorama images described in Section 4.5 [12]. This dataset includes five types of transformations i.e. the same image, cropping sub-images, shifting (i.e. overlapped image with the query, where overlapping area cover about 30% of the area of both images), flipping, and rotation of sub-images. The details of the ATRANS dataset are given in Section 4.5 and it includes $60,000$ images. The aim of building these datasets is to check whether the ECOTA realizes the difference between these five cases, i.e. not only detects the correlation but also illustrates the kind of correlation.

7.6.3 Evaluation Measures

We used Visual Studio and the OpenCV library to perform our experiments. To evaluate the performance of the ECOTA approach and compare it with RANSAC, PROSAC, LMEDS, COTA and 4COTA. We extracted three types of keypoint features for all image datasets, i.e. SIFT, SURF and BRISK keypoints. For SIFT and SURF keypoints, we applied the $kd-$tree with the nearest neighboring approach in the matching stage [116]. Since BRISK build a binary descriptor, we used Hamming distance to compute the distance between keypoints [105]. After that, we employed RANSAC, PROSAC, LMEDS, COTA, 4COTA and ECOTA to retrieve ND images for the benchmark collections. Finally, we evaluated the performance of the various methods by computing the amount of relevant and retrieved images that are successfully retrieved (i.e. the mean recall MR). Moreover, we computed the localization error as the relative offset in the horizontal and vertical direction. When ECOTA localizes the image I' on the image I, it computes the offset error as:

$$relative\ offset = \frac{1}{2}(\frac{\Delta x}{W} + \frac{\Delta y}{H}) \tag{85}$$

Table 38: The required time by RANSAC, PROSAC, LMEDS, COTA, 4COTA and ECOTA to estimate the correlation between images. The average time over all images of PANO dataset is presented in millisecond (ms)

Time complexity employing various keypoints			
Method	SIFT	SURF	BRISK
RANSAC	1.58 ms	1.59 ms	2.63 ms
PROSAC	0.72 ms	**0.52** ms	**0.62** ms
LMEDS	9.61ms	7.94 ms	8.64 ms
COTA	4.21 ms	1.86 ms	2.06 ms
4COTA	8.37 ms	4.28 ms	5.67 ms
ECOTA	**0.61** ms	0.66 ms	0.69 ms

where W and H are the width and height of I, Δx, Δy are the offset in horizontal and vertical directions respectively. In this way of localization error computation, ECOTA avoids the bias to image sizes.

7.7 Results & Decision

In our experiment, before applying ECOTA, we ranked the matches based on their similarity. To avoid the effect of rescaling of images, ECOTA accepts only the pair of matches $(P_i, P_i^{'})$, $d(P_j, P_j^{'})$ that the distance between their locations satisfy Equation (68). The least number of required pair of matches to apply ECOTA is $min_N = 3$. To compute the initial set of inliers, ECOTA stops the iteration operation (i.e. "For loop" in Algorithm 1) when $N > 10 \wedge Length(V_{Inliers}) \geq 5$ otherwise, when $Length(V_{Inliers}) \geq 2$. Employing these initial settings ECOTA filters the outliers and estimates the affine transformations [12].

7.7.1 Comparison of Time Complexity

We compared the required time to accomplish the step of correlation prediction by RANSAC, PROSAC, LMEDS, COTA, 4COTA and ECOTA. For the RANSAC, PROSAC and LMEDS, we considered the phase of fitting a model to matched features. For the COTA, 4COTA and ECOTA, we computed the required time to estimate the correlation between the pair of matches and to determine the transformation between images. The comparison was completed using the PANO dataset. Table 38 shows that the required time is independent of the type of extracted keypoints i.e. the performance is equivalent for each of SIFT, SURF and BRISK keypoints. Also, it presents that ECOTA needs the least time to compute the correlation and determine the affine transformation when SIFT keypoints are extracted whereas, PROSAC is faster than ECOTA when SURF or BRISK keypoints are used. However,

Table 39: The mean recall MR of RANSAC, PROSAC, LMEDS, COTA, 4COTA & ECOTA on PANO dataset

Scale	Method	RANSAC	PROSAC	LMEDS	COTA	4COTA	ECOTA
100%	SIFT	83.74	83.71	81.91	99.20	99.54	**99.92**
	SURF	96.75	95.52	96.67	97.68	95.56	**98.20**
	BRISK	85.37	81.65	85.60	90.28	90.30	**93.16**
30%	SIFT	78.86	65.98	76.14	95.33	96.09	**97.10**
	SURF	81.30	72.28	82.46	83.65	**87.84**	87.02
	BRISK	69.75	59.30	67.44	69.41	**76.19**	75.58
200%	SIFT	84.55	83.53	80.96	99.50	99.54	**99.96**
	SURF	81.30	96.83	97.38	98.02	96.31	**98.38**
	BRISK	95.94	90.18	91.72	95.07	92.97	**96.51**

Table 40: Localization error of RANSAC, PROSAC, LMEDS, COTA, 4COTA & ECOTA utilizing PANO dataset

Scale	Method	RANSAC	PROSAC	LMEDS	COTA	4COTA	ECOTA
100%	SIFT	0.0016	0.0016	0.0016	0.0013	0.0016	**0.0013**
	SURF	0.0024	0.0020	0.0020	0.0018	0.0026	0.0018
	BRISK	0.0028	0.0029	0.0028	0.0026	0.0030	0.0025
30%	SIFT	0.0033	0.0036	0.0031	0.0025	0.0026	**0.0024**
	SURF	0.0040	0.0046	0.0038	0.0037	0.0037	0.0035
	BRISK	0.0049	0.0057	0.0049	0.0048	0.0044	0.0045
200%	SIFT	0.0016	0.0016	0.0016	0.0016	0.0016	**0.0013**
	SURF	0.0020	0.0019	0.0019	0.0018	0.0025	0.0018
	BRISK	0.0027	0.0025	0.0024	0.0023	0.0029	0.0021

the performance of PROSAC in correlation prediction is lower than ECOTA since PROSAC employs only a set of top-ranked matches (based on similarity) to fit them in a model and this is once again a non-deterministic method since there is no confirmation that the top ranked matches are all correct matches.

7.7.2 Result on PANO Dataset

To compare the performance of RANSAC, PROSAC, LMEDS, COTA, 4COTA and ECOTA employing the PANO dataset, we computed the mean recall MR on a set of top retrieved results, which its size 1.5% more than the relevant images. After

Table 41: Results of RANSAC, PROSAC, LMEDS, COTA, 4COTA & ECOTA on the OXB dataset using the mean recall MR.

Scale	RANSAC	PROSAC	LMEDS	COTA	4COTA	ECOTA
100%	88.62	87.04	85.25	**99.57**	99.55	99.52
50%	85.37	83.32	81.31	99.20	98.81	**99.39**
30%	83.74	77.85	80.36	96.40	97.52	**98.45**

Table 42: The mean recall MR of RANSAC, PROSAC, LMEDS, COTA, 4COTA & ECOTA on the Aerial dataset

Scale	RANSAC	PROSAC	LMEDS	COTA	4COTA	ECOTA
100%	94.31	92.41	91.03	**99.86**	99.29	99.31
50%	80.00	75.00	80.00	99.08	98.13	**99.16**
30%	71.42	47.62	76.19	94.47	90.12	**94.82**

that, we estimated the correlation based on all six introduced approaches. Table 39 explains that all methods (i.e. RANSAC, PROSAC, LMEDS, COTA, 4COTA and ECOTA) perform well when the scale factor 100% is used. The performance of RANSAC, PROSAC and LMEDS decrease very dramatically when sub-images are re-scaled using the factor of 30%. This is due to the effect of the decreasing number of matches when the sub-images are down-scaled. The results show that the ECOTA outperforms the other methods on all scales. Table 40 presents the average relative offset, which is computed as given in Equation (85). The maximum localization error is defined by image down-scaling with a factor of 30% since the performance of all methods decreases in this case (as shown in Table 39). The localization error of 0.0013 is defined by ECOTA, which means that the average shifting is lesser than three pixels in both horizontal and vertical directions. This shifting error is too small compared to the dimensions of images in the PANO dataset.

7.7.3 Result on OXB, Aerial & PAIN Datasets

Since the performance of the SIFT algorithm exceeds the SURF and BRISK algorithms, we present the results for OXB, Aerial & PAIN datasets by applying only the SIFT algorithm. Tables 41, 42 and 43 present the mean recall MR only for scale factors 100%, 50% and 30% since the performance of all methods are equivalent when images are up-scaled using the factors 200% and 300%. Tables 41 and 43 explain that the performance of all methods decrease by down-scaling of images. Table 42 shows that when Aerial dataset is employed, the performance of PROSAC decreases to be about 48% when images down-scaled with the ratio 30%. This means that the candidate of matches to fit them in a model contains a lot of false matches. The comparison of Tables 39, 41, 42, and 43 details that ECOTA is more robust than

Table 43: The mean recall MR of RANSAC, PROSAC, LMEDS, COTA, 4COTA & ECOTA on the PAIN dataset

Scale	RANSAC	PROSAC	LMEDS	COTA	4COTA	ECOTA
100%	86.99	83.97	83.85	98.92	96.84	**99.51**
50%	83.74	84.12	84.91	98.52	96.86	**99.28**
30%	78.60	73.49	74.86	91.96	91.75	**93.88**

the other methods to dataset change i.e. to various types of images. The maximum decreasing of the performance of ECOTA by down-scaling of images is lesser than 5% for various datasets. However, when RANSAC, PROSAC or LMEDS are employed, the decreasing of performance exceeds 12%. This comparison describes that ECOTA outperforms the most of previous works, COTA and 4COTA too.

7.7.4 Classification Result

We compare the performance of the RANSAC and ECOTA to classify images of the ATRANS dataset. ATRANS contains five categories of images, for each query image, we estimated the corresponding class based on the RANSAC and ECOTA methods. Table 44 presents the ratio of correct classified images. It describes that ECOTA outperforms the RANSAC in this task. By analyzing the result, we found out that the RANSAC fails in most cases of reflection estimation. We did not compare with the COTA and 4COTA approaches since, they cannot detect the reflection.

7.7.5 Robustness Against Image Altering

To estimate the robustness of ECOTA to different kinds of image Altering (noise, blur, illumination change and rotaion), we applied the following altering separately on the panorama image dataset (PANO). These are: Gaussian noise with $\sigma^2 = 0.15$ (more details in [13]), blur with a factor $\sigma = 3$, illumination increasing with value $Il^+ = 70$ and decreasing with value $Il^- = 50$ and rotation using the angles 15^o, 30^o, 45^o and 90^o as described in [13]. After that, we checked the performance (the mean recall MR) of RANSAC, COTA, 4COTA and ECOTA in correlation detection using SIFT keypoints. Table 45 present that ECOTA outperforms RANSAC in solving the task of correlation detection in case of image altering. When blur or noise are applied to images, the performance of the RANSAC decrease very strongly since the number of matches decreases or the amount of false matches increase.

7.7.6 Localization & Outlier filtering

ECOTA is robust even when the false matches are 50% or more of the total matches. To clarify this, we present two examples in Figure 56 where more than 40% of matched features are outliers. Figure 56(a) shows that ECOTA filter successfully all

Table 44: Classification results of RANSAC & ECOTA on ATRANS dataset employing the mean recall MR.

Method	Scale 30%	Scale 100%	Scale 200%
RANSAC	54.14	65.81	67.44
ECOTA	**90.45**	**94.70**	**93.02**

Table 45: Comparison results in case of image Altering using the mean recall MR.

Altering	RANSAC	COTA	4COTA	ECOTA
Blur	43.09	95.53	**98.51**	97.75
Il^+	91.87	98.90	98.56	**99.50**
Il^-	83.74	93.84	94.01	**95.54**
Noise	71.54	94.79	94.96	**96.88**
Rotation	97.14	99.36	99.59	**99.95**

false matches as outliers. In this example, the total matches are 25. Only 23 of them satisfy the condition in Equation (68). By further processing using the hypothesis of ECOTA, 10 of them are defined as correct matches. Figure 56(b) presents an example where 30 matches are detected. ECOTA identifies 16 of them (i.e. 53%) as outliers. The rest matches are employed by ECOTA to estimate the affine transformation and predict the spatial location. In both examples in Figure 56, only ECOTA could predict the correct correlation between images. Figure 57 presents various cases where all methods (Figure 57(a)), two methods (Figure 57(b)) or only ECOTA (Figure 57(c)) estimated the correct affine transformation. Figure 57(d) presents a case where all methods failed in detecting the correlation between images. ECOTA detects a wrong location in this case since the same pattern is iterated in the top part of the whole scene. In addition, no features are detected in the corresponding patch to the sub-image. In this case, we repeated the experiment by modifying the setting of feature extraction to obtain more features in the whole scene. The result in Figure 58 explains that ECOTA detected the correct location. The matched features, in this case, are 25 one of them is excluded by applying Equation 68. As shown in Figure 58(a), from the rest 24 matches, ECOTA defined 12 false as outliers (i.e. half of the total matches). In this example RANSAC, PROSAC and LMEDS failed to identify the correlation since they cannot predict the correct correlation utilizing such amount of false matches is the same as correct matches.

7.8 Summary

In this chapter, we presented approaches to improve correlation detection between matched features and compute the transformation between near-duplicate images.

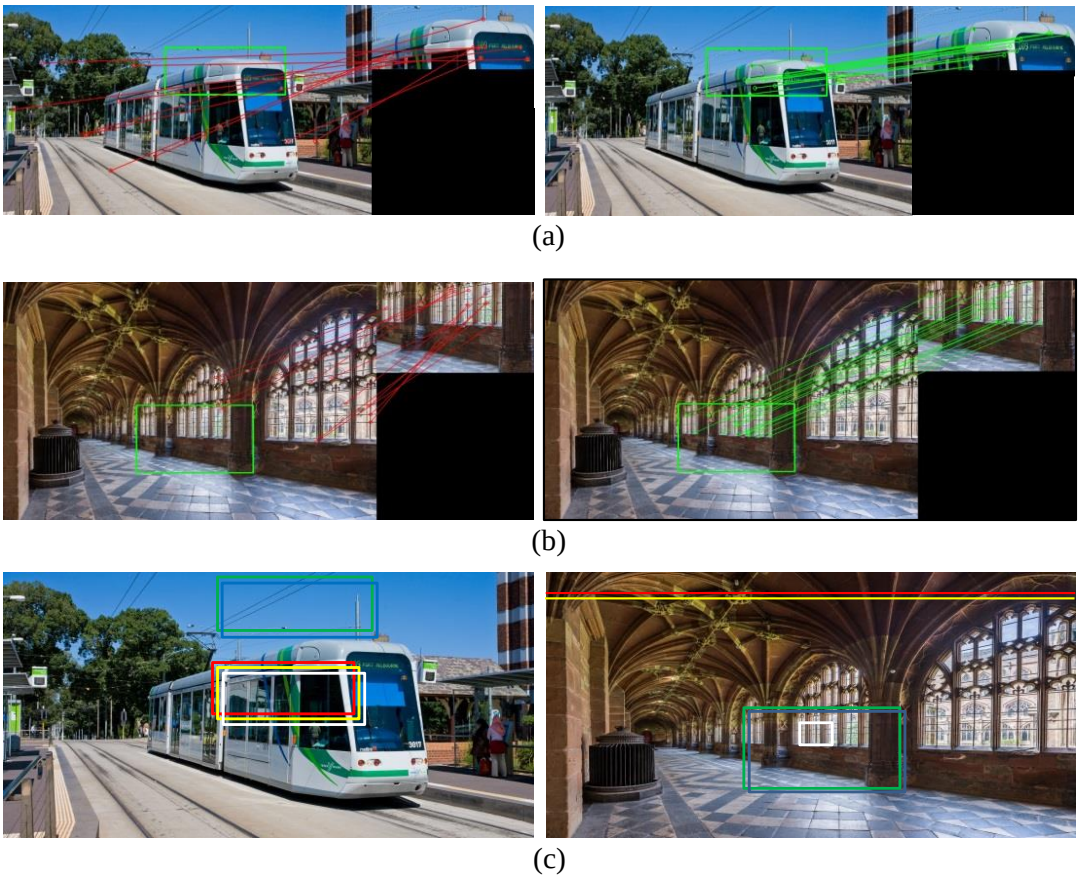


Figure 56: Filter the matches into inliers & outliers using ECOTA. Green box present the location by ECOTA which is the same as the ground truth. There are 23pair of matches. ECOTA detects (a) 10 of them as outliers (red lines) and (b) 13 of as inliers (green lines).

The introduced approaches address the presented problems in **RQ.3**. The results show that the ECOTA, COTA and 4COTA outperform the RANSAC, PROSAC and LMEDS. In addition, ECOTA reduces the processing time and has the proficiency to detect also reflection transformations. Moreover, ECOTA detects the correct transformation when half of the matches are outliers or when very few matches (e.g. only three) are found, which are impossible in a lot of state of the art approaches. In addition, ECOTA produces the smallest localization error compared to the other state of the art methods.

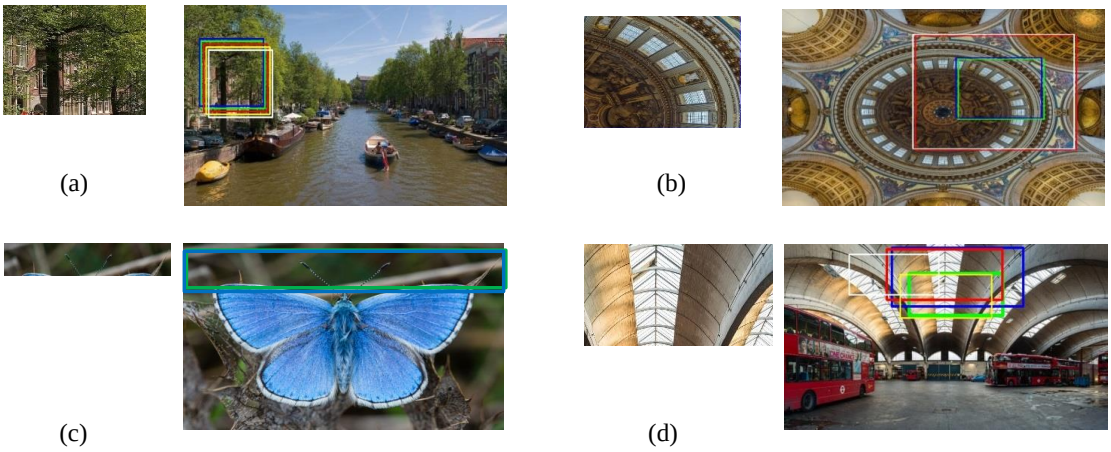


Figure 57: Localization of sub-images in whole scene using RANSAC (red), PROSAC (yellow), LMEDS (white) & ECOTA (blue). The ground-truth is the Green box. (a) Localization by all methods correct (b) Only by ECOTA and RANSAC correct. (c) All methods fail.

Figure 58: Filter the outliers using ECOTA. ECOTA detects 12 outliers (red lines) as well as 12 inliers (green lines) of total 24 pair of matches. ECOTA identifies the correct transformation.

8 Task based Evaluation: Limits and Potential of ECOTA

In contrast to the approaches in Chapters 5 and 7 which focus on detecting ND-images in the default settings as defined in Section 1.3 without any altering in the content of images, we focus in this chapter on altered ND-settings i.e. images that have been derived from others by copying, moving and modifying of an object. Such cases occur sometimes in media to show a larger crowd than it actually was. Figure 59 presents samples where portions of crowds have been duplicated to appear larger [18], [145]. The detection of duplicate objects, i.e. copied, modified and moved objects in the

(a) (b)

Figure 59: Samples of copy-moved images. (a) presents a photo published in Le Maghreb, a Tunisian newspaper, on January 2012 [18]. (b) shows a photo published by the national news agency Bernana, Malaysia [145]. Both photos were digitally altered by duplicating multiple portions of crowds to show them larger.

same image, is challenging since we need to define the identical objects or regions in an image. Moreover, we require to identify whether the copied object or area has been altered before duplicating it. For this, we applied the ideas of our approaches presented in Chapters 5 and 7 and propose the ECOTA-duplicate approach to detect copy, moved and (may be) modified objects in the same image, i.e. the approach allows to search for ND-images for a given query image and provides in addition the copied parts within the image. In the following sections, we present the details of our approach.

8.1 Duplicate Objects Detection

A way to modify images is to select (employing specific tools) and copy an object of an image and then set it in another location of the same image. The object could be re-sized, rotated, or flipped before fitting it in a new location.

The detection of images being modified by duplicating objects or areas is challenging since the original and modified images have the same content. The modified

image differs from the original one with only one object or area. The available tools nowadays to alter images allow powerful altering of images that are difficult to detect by the human visual system. Consequently, the requirement to verify the authenticity of an image becomes more important.

In this chapter, to detect the copied, modified and moved objects, we applied firstly the SIFT algorithm to extract the features from images. However, especially when objects are scaled-down or when backgrounds are very complex, no features have been found in the duplicate objects. The reason is by applying the SIFT algorithm with its default settings, we get a huge amount of features in the background and in many cases no features in the duplicate object. Hence, no matching between objects in the original and modified images will be detected. To overcome this problem, we employed our idea presented in Section 5.2 to prune the list of extracted features and at the same time preserve the most invariant and robust features, that are included in the copied, moved objects. This has been done by changing the values of σ and contrast parameters of SIFT keypoints as described in Section 5.2.

To detect the similarity and difference between the original image and the modified image which includes a duplicate object, we introduce a method ECOTA-duplicate, which relies on ECOTA (presented in Section 7.5) to identify the duplicate object in an image. Based on the properties of ECOTA, ECOTA-duplicate determines whether the duplicate object includes any type of transformations such as scale-up/down or rotation. In the following subsection we details ECOTA-duplicate.

8.1.1 ECOTA-Duplicate

ECOTA-Duplicate includes two main steps. The first is splitting the feature matches into two lists. The second is to apply ECOTA on each individual list. In the following, we explain these steps.

8.1.1.1 Split Feature Matches After extracting SIFT features from both images, we applied k-d tree algorithm to determine the mapped features. However, unlike the employed method in Subsection 7.6.3, we do not apply the best neighboring algorithm to select the best feature matches instead, we considered all multiple matches i.e. each feature of one image may match to many features in the second image. We considered the multiple matches to detect the duplicate object in the modified image. In this case, we cannot apply ECOTA on the produced feature matches since ECOTA constructs the triangles between feature matches in both images and in case of multiple feature matches the features of the duplicate object will be counted as outliers. To overcome this problem, we split the feature matches between two images into two lists. The first list contains feature matches of the original object with the background. The second includes the duplicate ones. The details of the algorithm are presented in Appendix D. Figure 60 presents some cases we discussed in our splitting strategy 2. Given are two images I and $I^{'}$, where $I^{'}$ is the same as I, but it has only one duplicate (i.e. copy-moved and modified) object.

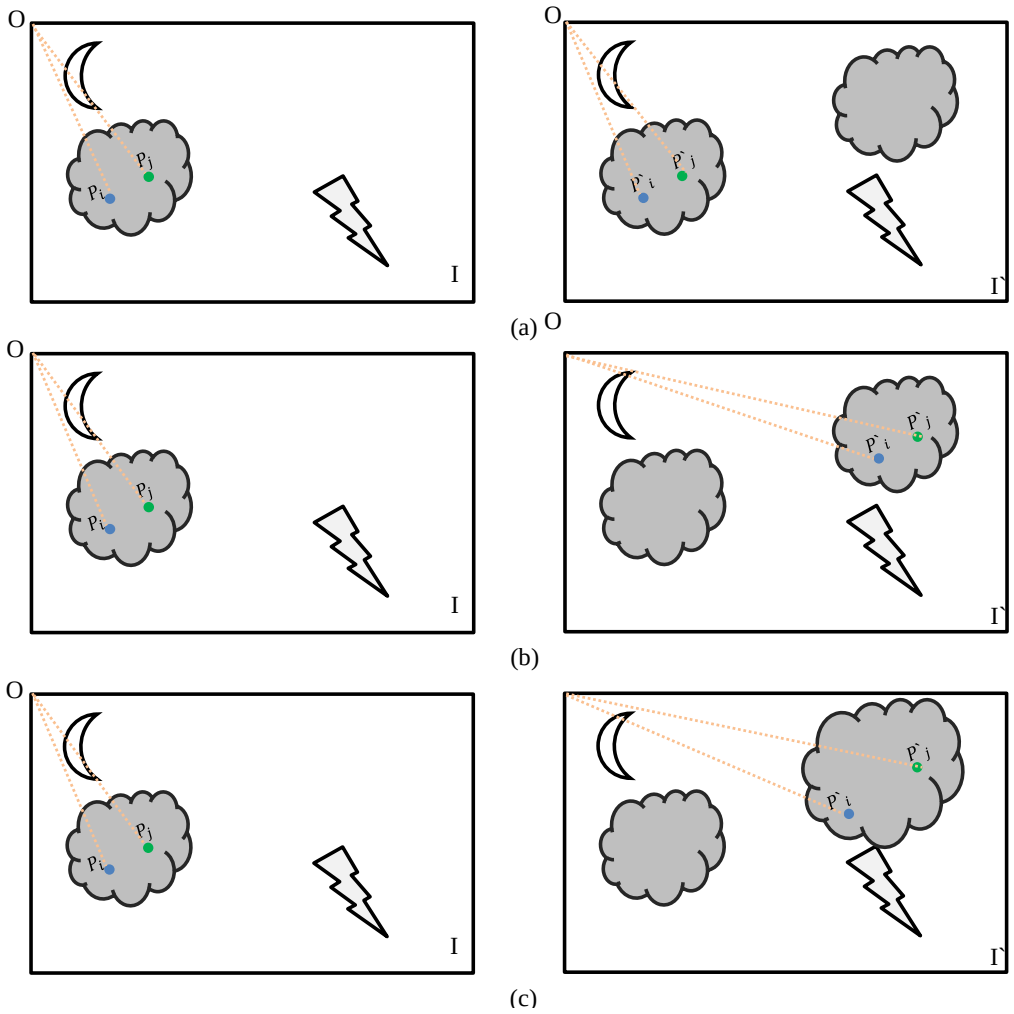


Figure 60: Samples of feature matches locations between two images I and $I^{'}$. $I^{'}$ contains a duplicate object. (a) presents the case where the pairs $(P_i, P_i^{'})$ and $(P_j, P_j^{'})$ belong to the original object in both images. (b) and (c) display possible cases where P_i, P_j and $P_i^{'}$, $P_j^{'}$ belong to the original and duplicate objects respectively.

Let N be the number of detected feature matches between I and $I^{'}$, $LP_{original}$ and $LP^{'}_{duplicate}$ are the lists of matches location in I and $I^{'}$, respectively. We split N into two lists $M_{original}$ the list of feature matches including the original object and the background of I and $I^{'}$ and $M_{duplicate}$ the list of feature matches between the original object in I and the duplicate object in $I^{'}$. Given are pairs $(P_i, P_i^{'})$ and $(P_j, P_j^{'})$ of features matches between images I and $I^{'}$, we require to investigate whether these pairs are inliers and in this case we should determine whether $P_i^{'}$ and $P_j^{'}$ belong to the original or duplicate object in image $I^{'}$. For this, we check the distances between (P_i, P_j) and $(P_i^{'}, P_j^{'})$, if they are identical (i.e. $\mid \vec{P_iP_j} \mid = \mid \vec{P_i^{'}P_i^{'}} \mid$), we compute

the distances between the origin and each of P_i, P_j, $P_i^{'}$ and $P_j^{'}$ if $\mid \overrightarrow{OP_i} \mid = \mid \overrightarrow{OP_i^{'}} \mid$ and $\mid \overrightarrow{OP_j} \mid = \mid \overrightarrow{OP_j^{'}} \mid$ then the pairs $(P_i, P_i^{'})$ and $(P_j, P_j^{'})$ belong to the original object or background in both images (as presented in Figure 60(a)), therefore, we append them to $M_{original}$. Otherwise i.e. if $\mid \overrightarrow{OP_i} \mid \neq \mid \overrightarrow{OP_i^{'}} \mid$ and $\mid \overrightarrow{OP_j} \mid \neq \mid \overrightarrow{OP_j^{'}} \mid$, then ($P_i^{'}$ and $P_j^{'}$) belong to the duplicate object (as shown in Figure 60(b)) hence, we append$(P_i, P_i^{'})$ and $(P_j, P_j^{'})$ to $M_{duplicate}$. Supposing that $\mid \overrightarrow{P_iP_j} \mid < \mid \overrightarrow{P_i^{'}P_i^{'}} \mid$ or $\mid \overrightarrow{P_iP_j} \mid > \mid \overrightarrow{P_i^{'}P_i^{'}} \mid$ and ($\mid \overrightarrow{OP_i} \mid < \mid \overrightarrow{OP_i^{'}} \mid$ and $\mid \overrightarrow{OP_j} \mid < \mid \overrightarrow{OP_j^{'}} \mid$) or ($\mid \overrightarrow{OP_i} \mid > \mid \overrightarrow{OP_i^{'}} \mid$ and $\mid \overrightarrow{OP_j} \mid > \mid \overrightarrow{OP_j^{'}} \mid$) (as presented in Figure 60(c)) then either the duplicate object is scaled-down or up, hence in this case we append$(P_i, P_i^{'})$ and $(P_j, P_j^{'})$ to $M_{duplicate}$. After completing the splitting of all feature matches following the details of Algorithm 2, we employ ECOTA on each of $M_{original}$ and $M_{duplicate}$ to filter out the outliers and to determine the type of applied transformation on the duplicate object.

8.1.1.2 ECOTA Application Feature matches of each of $M_{original}$ and $M_{duplicate}$ lists satisfy the requirements of ECOTA, hence we can apply it (as described in Section 7.5) on each list separately. We call this the application of ECOTA on both $M_{original}$ and $M_{duplicate}$ lists as *ECOTA-duplicate*. However, ECOTA computes the thresholds of edge and angles tolerance based on the size of images (i.e. thresholds are static in each case) as presented in Eq. 67) and Eq. 79), respectively. These thresholds are not helpful in case of duplicate object detection, since both images have the same resolution only the duplicate object may be up / down-scaled, rotated or flipped. To utilize the benefits of thresholds of ECOTA, we compute dynamic thresholds instead of static thresholds. Hence, each time we construct a triangle, we define the tolerance of edge and angle based on the values of its own edges and angles. To compute the tolerance of edges dynamically, for each constructed triangle in image I and its correspondence in image $I^{'}$, we compute the maximum edge as:

$$max\_edge = MAX(|\overrightarrow{P_iP_j}|, |\overrightarrow{P_iP_k}|, |\overrightarrow{P_jP_k}|) \tag{86}$$

and

$$max\_edge^{'} = MAX(|\overrightarrow{P_i^{'}P_j^{'}}|, |\overrightarrow{P_i^{'}P_k^{'}}|, |\overrightarrow{P_j^{'}P_k^{'}}|) \tag{87}$$

based on Eq. 86 and Eq. 87, we compute the *edge_angle_tolerance* as:

$$edge\_angle\_tolerance = MAX(\frac{log(max\_edge)}{max\_edge}, \frac{log(max\_edge^{'})}{max\_edge^{'}}) \tag{88}$$

The condition in Eq. 66 will be:

$$\begin{aligned}
&\left| \frac{P_iP_j}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i'P_j'}{max\left\{P_i'P_j', P_i'P_k', P_j'P_k'\right\}} \right| < edge_angle_tolerance \\
&\left| \frac{P_iP_k}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i'P_k'}{max\left\{P_i'P_j', P_i'P_k', P_j'P_k'\right\}} \right| < edge_angle_tolerance \\
&\left| \frac{P_jP_k}{max\left\{P_iP_j, P_iP_k, P_jP_k\right\}} - \frac{P_i'P_k'}{max\left\{P_i'P_j', P_i'P_k', P_j'P_k'\right\}} \right| < edge_angle_tolerance
\end{aligned} \tag{89}$$

Hence, we compute the *edge_angle_tolerance* dynamically and its value depends on the constructed triangles (i.e. *edge_angle_tolerance*) will be small for a triangle with at least one long edge than other triangles with relative shorter edges). The *edge_angle_tolerance* depends on the length of edges and its value is small for long edges therefore, using it as a tolerance value of angles too is very suitable.

To make the threshold of angles dynamic too, we use the *edge_angle_tolerance* in Eq. 78 to get the following condition:

$$\begin{aligned}
\left|\varphi_{ij} - \varphi_{ij}^{'}\right| &\leq edge_angle_tolerance \\
\left|\varphi_{jk} - \varphi_{jk}^{'}\right| &\leq edge_angle_tolerance \\
\left|\varphi_{ik} - \varphi_{ik}^{'}\right| &\leq edge_angle_tolerance
\end{aligned} \tag{90}$$

So, the definition of *edge_angle_tolerance* improves the detection of duplicate objects in cases o scale change, rotation or flipping.

However, the usage of *edge_angle_tolerance* causes sometimes passing some outliers in the inlier list. This happens when the length difference is too big between the minimum and maximum edges of a constructed triangle. Figure 61 shows an example where the all three edges satisfy the condition in Eq. 88 but at least the pairs $A, A^{'}$ and $B, B^{'}$ are false matches. In this example $max\_edge(ABC) = 35$, $max\_edge(A^{'}B^{'}C^{'}) = 49$ and $log(35)/35 = 0.044$, $log(49)/49 = 0.035$. Using Eq. 88 $edge\_angle\_tolerance = log(35)/35 = 0.044$. The differences between the corresponding normalized edges are:

$$\left|\frac{\overline{AB}}{max\_edge(ABC)} - \frac{\overline{A'B'}}{max\_edge(A'B'C')}\right| = \left|\frac{6}{35} - \frac{7}{49}\right| = 0.029$$

$$\left|\frac{\overline{BC}}{max\_edge(ABC)} - \frac{\overline{B'C'}}{max\_edge(A'B'C')}\right| = \left|\frac{34}{35} - \frac{49}{49}\right| = 0.029$$

$$\left|\frac{\overline{AC}}{max\_edge(ABC)} - \frac{\overline{A'C'}}{max\_edge(A'B'C')}\right| = \left|\frac{35}{35} - \frac{47}{49}\right| = 0.040$$

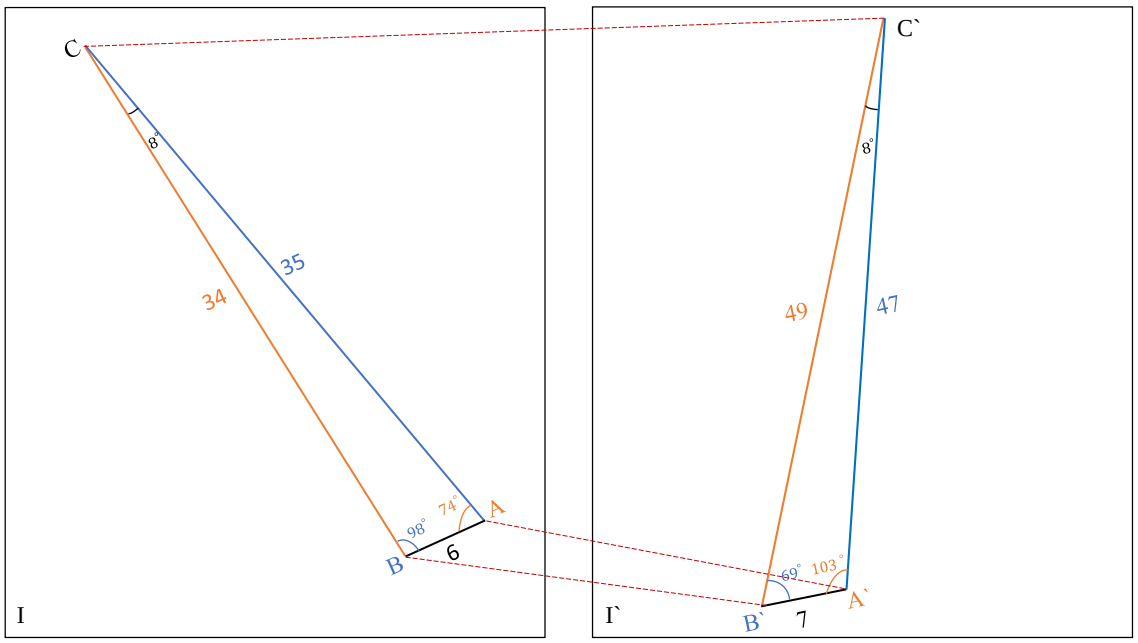


Figure 61: Samples of feature matches between two images I and $I^{'}$. The pairs $A, A^{'}$ and $B, B^{'}$ satisfy the condition in Eq. 88 even they are outliers.

Hence all edges satisfy the *edge_angle_tolerance* Eq. 89 and pass as inliers but the pairs $A, A^{'}$ and $B, B^{'}$ are false matches. To solve this problem i.e. to minimize the detection of outliers as inliers, we impose additional constraint to check the relation between the length of sides of the constructed triangle. This is:

$$MIN(|\overrightarrow{P_iP_j}|, |\overrightarrow{P_iP_k}|, |\overrightarrow{P_jP_k}|) \times 5 \geq MAX(|\overrightarrow{P_iP_j}|, |\overrightarrow{P_iP_k}|, |\overrightarrow{P_jP_k}|)$$

The constraint in Eq. 8.1.1.2 avoids constructing triangles where the difference between the shortest and longest edges is too big (i.e. when one angle is big and another one lesser than 10^o).

8.2 Experiment Settings

To evaluate the performance of ECOTA-duplicate, we used the duplicate Objects dataset detailed in Section 4.8. As evaluation measures, we employed the mean recall and the variance of the recall described in Section 2.5. To compare our results with the Block-Point approach presented in [21], we compute the precision and recall on the pixel level of mask images to determine the detected area of an object. This is done as follows [21]:

$$Precision - Pixel - Level = \frac{n(Area_D \cap Area_R)}{n(Area_D)} \tag{91}$$

$$Recall - Pixel - Level = \frac{n(Area_D \cap Area_R)}{n(Area_R)} \tag{92}$$

where $n(Area_D)$ is the number of pixels in the retrieved area and $n(Area_R)$ the number of pixels in the duplicate area.

We present our results for Precision-Pixel-Level and Recall-Pixel-Level as average over all the 20 images (i.e. MP-Pixel and MR-Pixel) of duplicate Objects dataset for each transformation individually. Additionally, we analyzed the performance of our algorithm on the basis of how accurately it could detect the modification of the duplicate object. The parameters we checked for this are both the degree of rotation and the scaling factor of the duplicate object. To evaluate the performance of ECOTA-duplicate in the individual cases, we calculated the variance of average Recall-Pixel-Level.

8.3 Results and Discussion

To present the results, we grouped the modified images into four groups i.e. rotation in range $[-25^o, 25^o]$ with step 5^o, rotation in range $[-5^o, 5^o]$ with step 1^o, rotation in range $[0^o, 330^o]$ with step 30^o, scale change in range [0.25, 2] with step 0.25 and scale change in range [0.8, 1.2] with step 5. In the tables of results, we denoted the rotation and the scale changes as rx and sx, respectively, whereas x is a positive or negative number. We compared the result of ECOTA-duplicate with the results of the Block-Point approach [21] when SIFT keypoints have been used. By extraction of SIFT keypoints we controlled automatically the values of σ and contrast thresholds (as described in Section 5.2.1) to keep the number of extracted features lesser than 800 per image without using any weights.

Tables 46, 47 and 48 present the results when copied objects were rotated using the ranges $[-25^o, 25^o]$ with step 5^o, $[-5^o, 5^o]$ with step 1^o, respectively. They show that the Block-Point approach [21] retrieve better MP-Pixel than ECOTA-duplicate. The reason is the Block-Point approach [21] uses the benefits of keypoints and dominant color to segment the input image and then to determine the similar region. However, the MR-Pixel of ECOTA-duplicate exceeds the one of the Block-Point approach [21] in all cases of rotation and scale changes. The best MR of ECOTA-duplicate is found when rotation of -4^o, -3^o or 3^o, 4^o has been applied followed by rotation of -1^o and 1^o. Figure 62 presents the results of ECOTA-duplicate where the duplicate object has been rotated by 4^o (first column) and -20^o (second column). The first row of results shows that ECOTA-duplicate detects the similarity between the original and modified images. The second presents the detection and localization of the duplicate object and the third displays the detected outliers by ECOTA-duplicate. The fourth row details the output of ECOTA-duplicate. The output clarifies that ECOTA-duplicate excludes the outliers even when they are more than 80% of the total feature matches in the list $M_{duplicate}$. In addition the output displays that ECOTA-duplicate computes the angle in the range 0^o, 360^o], therefore

Table 46: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of rotation in range $[-25^o, 25^o]$ with step of five degrees. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	r-25	r-20	r-15	r-10	r10	r15	r20	r25
Block-Point [21]								
MR-Pixel	9.00	10.00	15.00	12.00	15.00	10.00	11.00	9.00
MP-Pixel	95.00	96.00	97.00	98.00	98.00	98.00	95.00	93.00
ECOTA-duplicate								
MR-Pixel	43.07	43.07	44.33	48.13	41.80	41.67	42.73	42.93
MP-Pixel	48.13	51.67	40.53	45.60	43.07	45.60	49.40	44.33
VR-Pixel	12.11	12.07	28.21	15.10	12.09	9.17	10.85	9.43
MR	67.13	67.13	67.13	59.53	59.53	67.13	67.13	67.13
VR	18.25	18.25	18.25	19.11	19.11	18.25	18.25	18.25
Maximum Outliers % of ECOTA-duplicate	74.30	77.30	82.14	78.61	74.54	76.80	73.81	77.78

the angle -20^o is computed as 340^o by ECOTA-duplicate. The black rectangles in the first and second rows present the detected similarity between images and they are drawn based on the minimum and maximum of the keypoint coordination. They do not cover whole images in the first row since no keypoints are detected in the patches of images where the intensity does not change.

In case of rotation in range [0^o, 330^o] with step 30^o, Tables 49 and 50 show that the best MR values are obtained by 0^o and 180^o. The MR, VR and VR-Pixel values of the Block-Point approach [21] have not been given, therefore we could not compare with it. The VR and VR-Pixel values of the ECOTA-duplicate are small in about all cases. Figure 63 presents an example where ECOTA-duplicate detects successfully the similarity between the original and modified images and the difference between of them, i.e. detects the duplicate object in case of rotation by 60^o and 330^o. In addition, it presents that ECOTA-duplicate filter out all outliers (third row). Moreover, it determines the number of inliers and outliers in the list of the duplicate object, whether the duplicate object has been rotated or scaled, the rotation angle and the scale ratio. The results show clearly that ECOTA-duplicate detects the exact rotation angle but sometimes with very small error (lesser than 1^o).

In case of duplicate objects with scale changes, Tables 51 and 52 presents that ECOTA-duplicate obtains better MP-Pixel and MR-Pixel values than the Block-Point approach [21] when the duplicate object has been scaled-down with values

Table 47: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of rotation in range $[-5^o, -1^o]$ with step of one degree. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	r-5	r-4	r-3	r-2	r-1
Block-Point [21]					
MR-Pixel	11.00	11.00	9.00	8.00	9.00
MP-Pixel	97.00	99.00	97.00	95.00	98.00
ECOTA-duplicate					
MR-Pixel	51.93	46.61	59.53	48.13	64.60
MP-Pixel	50.67	45.60	41.80	44.33	44.33
VR-Pixel	18.15	16.58	17.73	17.73	11.21
MR	65.87	79.80	79.80	59.53	73.47
VR	19.74	18.94	19.74	18.94	17.76
Maximum Outliers % of ECOTA-duplicate	81.4	80.43	74.13	80.95	80.00

Table 48: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of rotation in range $[1^o, 5^o]$ with step of one degree. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	r1	r2	r3	r4	r5
Block-Point [21]					
MR-Pixel	12.00	13.00	11.00	10.00	10.00
MP-Pixel	97.00	94.00	96.00	98.00	98.00
ECOTA-duplicate					
MR-Pixel	46.61	69.67	49.40	43.07	43.07
MP-Pixel	51.93	54.47	60.80	53.20	39.27
VR-Pixel	16.05	8.68	9.47	18.15	17.33
MR	73.47	59.53	79.80	79.80	65.87
VR	18.08	19.26	18.12	18.12	17.04
Maximum Outliers % of ECOTA-duplicate	76.08	75.56	77.78	75.56	77.14

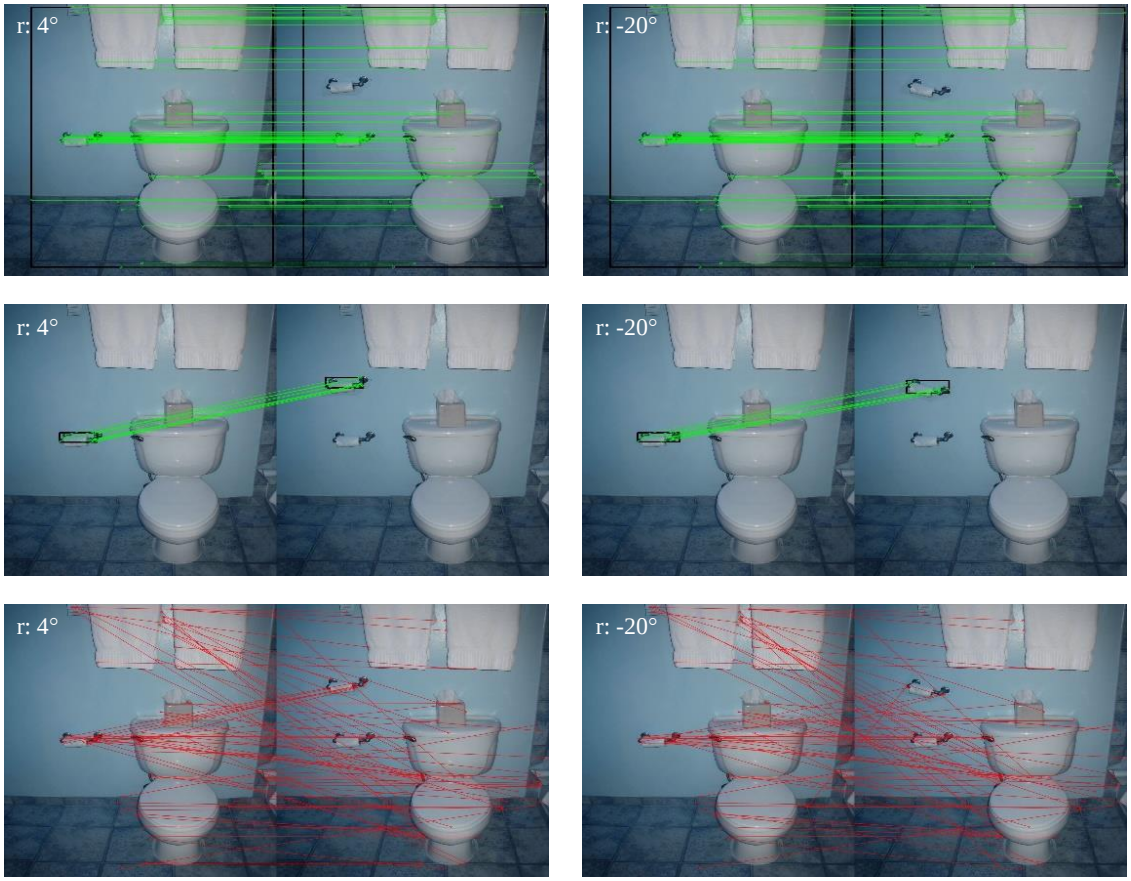


Figure 62: Sample results of ECOTA-duplicate when the duplicate object is rotated by 4^o (first column) and by -20^o (second column). The first row presents the similarity between the original and modified images employing SIFT features. The second row shows the difference between them i.e. the duplicate object and its location. The third row displays the detected outliers by ECOTA-duplicate. The fourth row presents the output of ECOTA-duplicate. The angle -20^o is detected by ECOTA-duplicate as 340^o i.e. ECOTA-duplicate compute the angle in the range $[0^o, 360^o]$.

0.25 and 0.5. In the other cases of scaling-down / up, the Block-Point approach [21] presents the best MP-Pixel values and ECOTA-duplicate finds the best MR-Pixel values. The best MR of ECOTA-duplicate are found by scale ratios 95%,105%, 90% followed by 100%, 110% and 200% and the worst performance of ECOTA-duplicate is found by scaling the objects down with ration 25%. Figure 64 presents an example where ECOTA-duplicate detects the similarity and difference between the original

Table 49: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of rotation in range $[0^o, 150^o]$ with step of 30^o. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	r0	r30	r60	r90	r120	r150
Block-Point [21]						
MR-Pixel	22.00	12.00	8.00	11.00	5.00	7.00
MP-Pixel	99.00	99.00	97.00	99.00	99.00	90.00
ECOTA-duplicate						
MR-Pixel	58.06	39.27	28.67	31.67	38.00	35.47
MP-Pixel	44.33	48.13	40.53	51.93	53.20	54.47
VR-Pixel	12.02	12.22	9.00	8.13	10.00	9.22
MR	73.46	59.53	45.60	59.53	67.13	67.13
VR	18.94	19.74	18.94	18.15	19.74	19.74
Maximum Outliers % of ECOTA-duplicate	74.00	76.21	74.13	73.08	76.09	78.00

Table 50: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of rotation in range $[180^o, 330^o]$ with step of 30^o. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	r180	r210	r240	r270	r300	330
Block-Point [21]						
MR-Pixel	9.00	7.00	11.00	10.00	10.00	8.00
MP-Pixel	99.00	93.00	98.00	90.00	90.00	98.00
ECOTA-duplicate						
MR-Pixel	43.07	43.07	39.00	43.07	39.00	43.20
MP-Pixel	57.00	39.27	38.00	55.73	49.40	39.00
VR-Pixel	10.81	20.07	12.08	12.17	11.01	11.00
MR	73.47	53.20	53.20	65.87	59.53	53.20
VR	19.03	18.00	18.94	18.41	19.55	18.13
Maximum Outliers % of ECOTA-duplicate	77.72	73.18	74.20	74.00	74.21	76.52

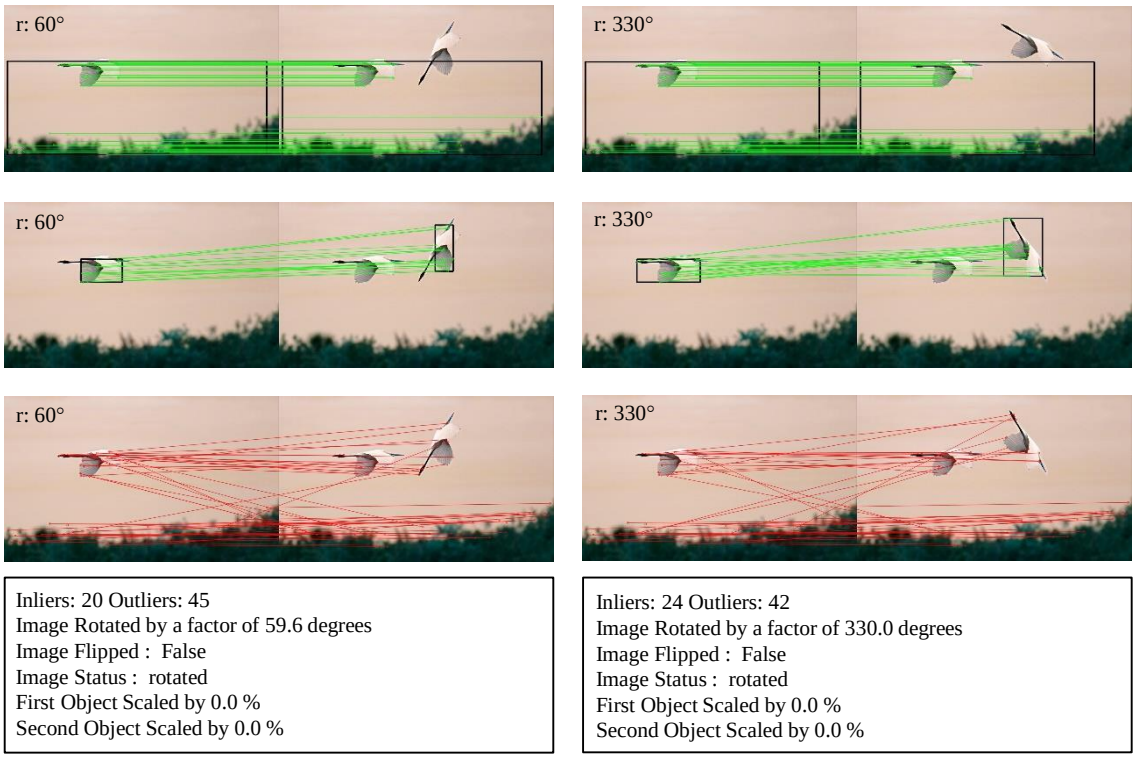


Figure 63: Sample results of ECOTA-duplicate when the duplicate object is rotated by 60^o (first column) and by 330^o (second column). The first row presents the similarity between the original and modified images employing SIFT features. The second row shows the difference between them i.e. the duplicate object and its location. The third row displays the detected outliers by ECOTA-duplicate. The fourth row presents the output of ECOTA-duplicate.

image and modified image when the duplicate object is scaled down by ratios 25% (first column) and 0.50% (second column). However, when the duplicate object down-scaled by 25%, ECOTA-duplicate detects some outliers as inliers, as shown in the second row. The reason is the big difference in size between the original and duplicate object. Hence, ECOTA-duplicate detects the wrong scaling ratio (in the fourth row the detected ratio is 39% instead of 25%). When the object is down-scaled by the ratio 50%, ECOTA-duplicate detects the scaled and duplicate object correctly and computes the correct scale ratio (i.e. in the fourth row the detected ratio is 50%). The output of ECOTA-duplicate in the fourth row presents that ECOTA-duplicate detects the correct status of the duplicate object, i.e. not rotated, not flipped but only scaled. Figure 65 presents two cases where ECOTA-duplicate fails to detect the duplicate object. The reason is the duplicate object is too small therefore, lesser than three feature matches have been detected between the original and duplicate objects. This number of feature matches is not enough to apply ECOTA-duplicate. The result in all tables and figures presents that the ECOTA-duplicate retrieve more areas of copied objects than the Block-Point approach [21]. To justify the robustness

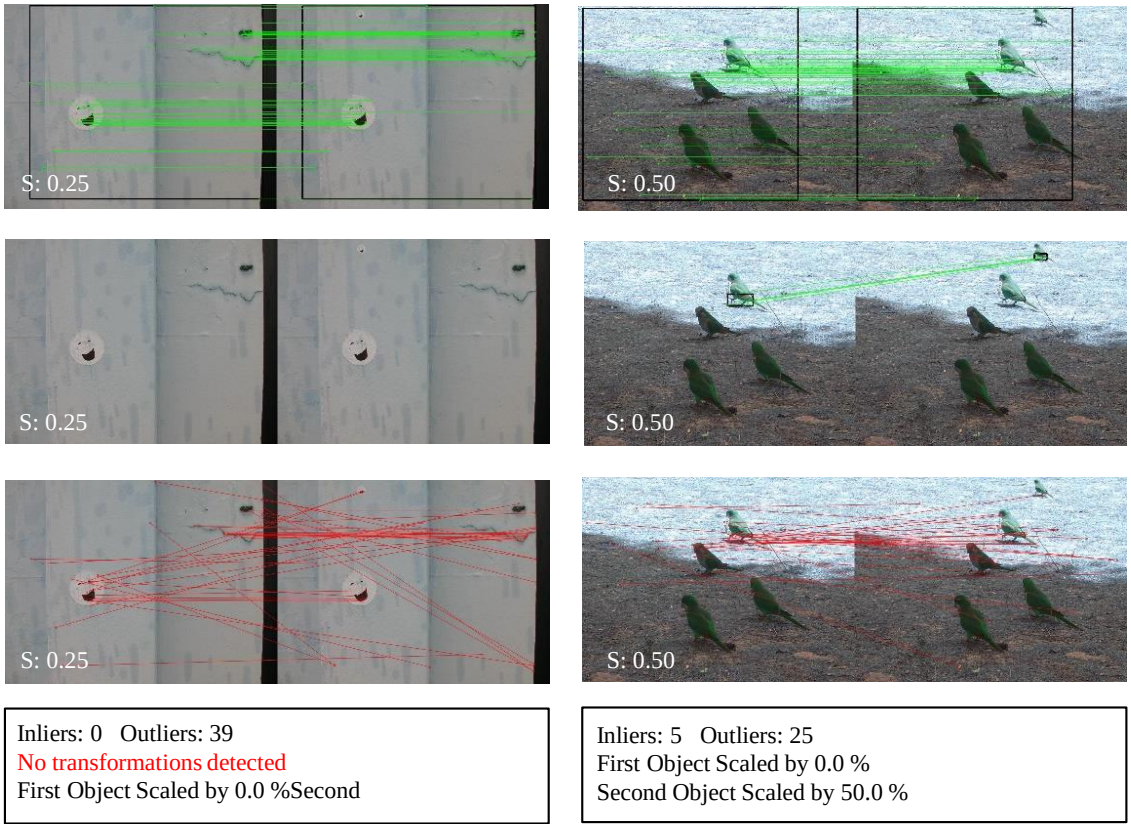


Figure 64: Sample results of ECOTA-duplicate when the duplicate object is scaled-down by 25% (first column) and by 50% (second column). The second row shows the difference between them, i.e. the duplicate object and its location. However, since the duplicate object is too small in case of down-scaling by 25%, some outliers are detected as inliers by ECOTA-duplicate. Hence, the output of ECOTA-duplicate presents wrong estimated scale value (in the first column fourth row).

of the ECOTA-duplicate approach, we present in all tables and in all Figures the maximum number of outliers, where ECOTA-duplicate is still able to detect the duplicate objects. The results show that ECOTA-duplicate is robust even when more the 80% of feature matches are outliers.

8.4 Summary

In this chapter, we presented the application of our approaches presented in Chapters 5 and 7 to detect copied-modified-moved objects in the same image. The task here is to detect the similarity and the difference between the original and modified images. For this, we presented the ECOTA-duplicate approach, which detects the similarity between both images. In addition, it detects the duplicate object and determines whether this object has been rotated or scaled. The ECOTA-duplicate approach is robust against the amount of the outliers since it detects and localizes the duplicate object even when the outliers are more than 80% of the total feature

Table 51: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of scaling change in range $[0.25, 2]$ with step of 0.25. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	s0.25	s0.5	s0.75	s1	s1.25	s1.5	s1.75	s2
Block-Point [21]								
MR-Pixel	0.00	4.00	8.00	23.00	10.00	8.00	9.00	3.00
MP-Pixel	0.00	42.00	80.00	99.00	98.00	95.00	91.00	75.00
ECOTA-duplicate								
MR-Pixel	31.67	91.03	33.96	72.20	64.60	76.00	70.93	62.07
MP-Pixel	51.93	43.32	51.93	39.75	40.21	42.10	44.00	44.53
VR-Pixel	8.00	2.63	6.11	23.52	19.14	25.43	19.57	16.33
MR	49.27	59.53	59.53	67.13	65.87	65.87	59.98	67.13
VR	19.34	18.31	18.76	17.10	19.38	19.38	18.14	17.16
Maximum Outliers % of ECOTA-duplicate	84.67	81.25	76.67	74.41	80.41	76.10	75.23	76.60

matches. We compared the ECOTA-duplicate approach with the Block-Point approach [21] and we found that the Block-Point approach detects better MP-Pixel values that the ECOTA-duplicate approach since the Block-Point approach used the keypoints and dominant color features but we used in our approach only keypoint features. However, the ECOTA-duplicate approach detects better MR-Pixel than the Block-Point approach, which means that the ECOTA-duplicate approach retrieves more areas of duplicate objects than the Block-Point approach.

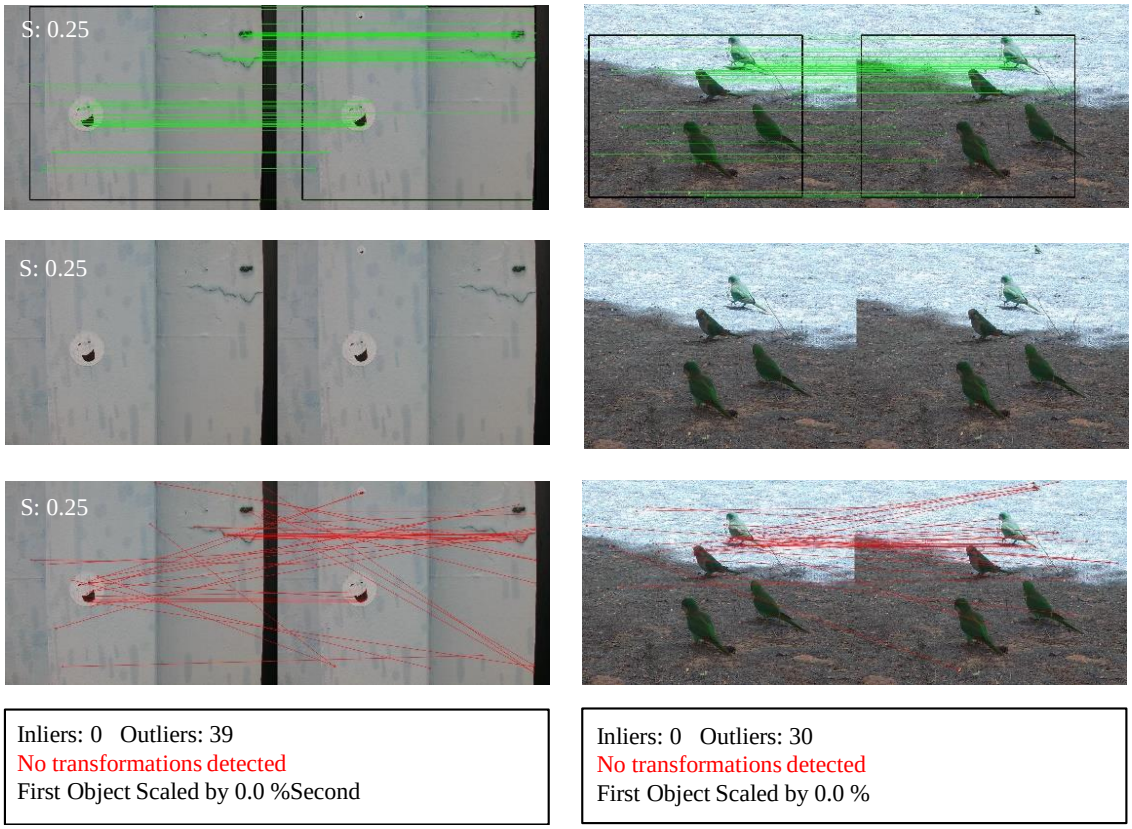


Figure 65: Sample results of ECOTA-duplicate when the duplicate object is scaled-down by 0.25%. The second row presents that ECOTA-duplicate fails to detect the duplicate object. The output (fourth row) shows the results of ECOTA-duplicate in form of no duplicate object has been detected.

Table 52: Comparison of our method ECOTA-duplicate and the Block-Point approach [21] in case of scaling change in range $[0.8, 1.2]$ with step of 0.05. The comparison is done in terms of, MR, Precision-Pixel-Level, Recall-Pixel-Level, the variance of MR and variance of Recall-Pixel-Level. We approximated the values of the Block-Point approach as they are shown only in graphs.

Measure	s0.80	s0.85	s0.90	s0.95	s1.05	s1.10	s1.15	s1.20
Block-Point [21]								
MR-Pixel	4.00	3.00	2.00	5.00	7.00	11.00	3.00	2.00
MP-Pixel	81.00	93.00	96.00	99.00	99.00	99.00	98.00	98.00
ECOTA-duplicate								
MR-Pixel	55.73	43.07	53.20	60.80	45.60	64.60	68.40	59.53
MP-Pixel	53.47	44.33	46.87	37.47	43.07	42.87	44.33	38.23
VR-Pixel	16.13	13.00	18.26	20.52	12.18	23.60	20.19	22.64
MR	53.20	65.87	72.20	73.47	67.13	67.13	73.47	59.53
VR	18.94	19.07	18.94	18.22	19.31	19.07	18.74	19.71
Maximum Outliers % of ECOTA-duplicate	68.30	77.01	74.41	77.78	75.00	77.78	78.30	75.38

9 Conclusion

The goal of this thesis was to improve and accelerate the detection and retrieval of near-duplicate images and to determine the spatial correlation between the ND-images without any prior knowledge about their content. Near-duplicate images are images that present the same scene but differ (slightly) with scale, illumination, perspective, resolution, rotation or level of noise. As shown in Figure 1, image-based content retrieval systems start, in general, by representing images with one or more kinds of their features. Next, these features are structured to make them accessible for further process and simplify their matching. The matching process is accomplished by computing the similarity between the query and dataset images. After that, the results are ranked based on their decreasing similarity with the query. The set of top N images are retrieved (where N is defined based on the task). Finally, the performance of the system is evaluated by computing the number of relevant images that are successfully retrieved. In case of dealing with near-duplicate images, an additional step is required to identify the spatial correlation between queries and their retrieved images.

The main idea of our work is to get better understanding of the content of images and their feature extraction methods to argue why two images are similar (or not similar) and to improve feature detection and matching techniques. In addition, the most approaches that we improved in this work do not need any training stage and process images without any prior knowledge about their content. Hence we do not need big image datasets for training (contrary to deep learning techniques). For these reasons, we do not use any of deep learning techniques in our work.

In the scope of improvement of near-duplicate image retrieval system, we presented in Chapter 1, Section 1.2 our research questions of this thesis. These questions inspired of the open challenges clarified in Chapter 3, Section 3.6. **RQ.1** presented the possible improvement in the feature extraction step to accelerate the matching process and preserve the quality of the extracted features. The idea of **RQ.2** is to accelerate the matching process of the ND-images and enhance the ranking of the relevant in the list of retrieved images. Whereas, the focus of **RQ.3** is about identifying the spatial correlation between the ND-images with an approach that indicates the non-relevant images. This approach should describe the correlation between the ND-images without any prior knowledge of their content. In the following sections, we conclude the methodology and results of each of **RQ.1**, **RQ.2** and **RQ.3**, respectively.

9.1 RQ.1: Improve Feature Extraction

Our method to improve the feature extraction step was reported in Chapter 5. The SIFT algorithm is the preferable keypoint detector and descriptor in solving ND-retrieval tasks since SIFT keypoints are invariant to affine transformation and robust to scale change, adding noise or blur to images. The SIFT algorithm builds a

descriptor of 128 elements for each extracted keypoint. The indexing and matching of these descriptors employing a big image dataset is time- and memory-consuming. Therefore, we proposed our own method to compress the regions around SIFT keypoints and build 64 dimensional descriptors. We called our method region compressed SIFT (RC-SIFT). We constructed the row- and column-RC-SIFT and justified that both of them perform, as well as the original SIFT algorithm. We proved that RC-SIFT outperforms the state-of-the-art methods such as SIFT$-64D$ proposed in [102] and SURF$-64D$ introduced in [27] in solving the ND-retrieval task. Furthermore, we showed that the RC-SIFT requires lesser computation costs in indexing and matching steps than the original SIFT. We evaluated the invariant and stability of the RC-SIFT against various kinds of image deformations. We found that RC-SIFT is invariant to rotation, illumination and scale change and robust to adding noise and perspective change.

We evaluated the robustness of our RC-SIFT$-64D$ and the original SIFT$-128D$, SIFT$-64D$ and SURF$-64D$ to combinations of image transformations and deformations. We found that the performance of the RC-SIFT$-64D$ is still similar to the original SIFT$-128D$. Moreover, we deduced that the increased amount of noise in the combinations decreases the robustness of all methods. In addition, we found out that the combination with blur destroys their robustness too.

The amount of the extracted SIFT descriptors is not constant for all images. It depends on the content, intensity change and resolution of images. To control the amount of the extracted SIFT keypoints in the ND-retrieval field, we introduced a method in Chapter 5, Section 5.2.1 to prune the list of the extracted SIFT and RC-SIFT keypoints based on their scale or contrast properties. Next, to involve weights computed based on scale, contrast and orientation differences in the matching stage. We obtained the best performance by truncating keypoints based on their scale property and at the same time employing weights based on orientation and contrast properties.

9.2 RQ.2: Accelerate Image Matching & Improve Ranking

To accelerate image matching and retrieval steps, we introduced our hybrid approaches in Chapter 6. The idea of these approaches is to match the near-duplicate images based on their HSV color feature. After that, to re-rank only a subset of the top retrieved images based on their SIFT keypoints. In this way, we reduced the cost of SIFT keypoints matching between a query and all image dataset. For our approaches F-HS-SIFT and FP-HS-SIFT, we built fuzzy color histograms instead of crisp ones. In addition, we found that skipping the third channel of the HSV color space obtains more robust color histograms to small lighting changes. We justified that the segmentation of images into blocks follows, by calculating the color histograms for the whole image plus individual blocks improves the performance of the hybrid approaches F-HS-SIFT and FP-HS-SIFT. Moreover, we found that

re-ranking the retrieved images, which form only 5% of the total dataset size, based on their SIFT keypoints improve the performance comparing to extracting only the SIFT algorithm or the HSV color histograms. Hence, the hybrid approaches F-HS-SIFT and FP-HS-SIFT required lower costs to complete ND-image retrieval task than applying of only keypoint detectors and descriptors (i.e. SIFT keypoints).

9.3 RQ.3: Estimate the Spatial Correlation

We presented our methods COTA and its extension ECOTA to improve the correlation detection between the near-duplicate images in Chapter 7. ECOTA utilizes the spatial locations of feature matches between two images. It computes the congruency of the constructed triangles between the positions of feature matches. Based on the congruency of these triangles the decision is made whether two images are near-duplicate or not. Moreover, the type of correlation between the ND-images is identified. The advantages of ECOTA are that it is proficient in distinguishing the outliers of feature matches while estimating the kinds of spatial transformations. ECOTA filters out the non-relevant elements of the list of retrieved images and arguments the rejection of these images. COTA and ECOTA outperform the state-of-the-art methods since they employ indicators to estimate the set of features that are useful in the transformation prediction step.

While RANSAC, PROSAC and LMEDS approaches fail to estimate the correlation when more than 40% of feature matches are outliers, the performance of ECOTA is robust. Moreover, the number of outliers does not affect its performance. Furthermore, it filters out the outliers and computes the spatial correlation between images even when 70% of the matches are outliers. Based on the correct feature matches, ECOTA computes and describes the type of correlation between images. Consequently, it classifies the retrieved images based on the detected correlation (i.e. same but at different scale or rotation, sub-, reflected or overlapped image).

We verified the performance of ECOTA using various structures of images. It shows robust and consistent overall datasets utilizing multiple settings. We proved that ECOTA works well under significant differences between images in scale and rotation or scale and reflection. It can estimate the correlation between images even if only a few feature matches are obtained that are not sufficient for the state-of-the-art methods as discussed above.

We evaluated our approaches in Chapter 5 (Section 5.2.1) and 7 in solving interactive near-duplicate retrieval problems through the task of copied duplicated and moved object detection. For this, we presented our approach (ECOTA-duplicate) in Chapter 8. We justified that, ECOTA-duplicate detects the similarity between the original and modified image, moreover, it determines the difference between of them and localize the duplicated object in the modified image. However, we found out that ECOTA-duplicate is almost not able to detect the duplicated object or detect it wrong when the scaling ratio is lesser than 0.25%. But ECOTA-duplicate

is robust and still able to exclude all outliers even when more than 80% of feature matches are outliers.

9.4 Future Work

We presented in Chapter 7 our method to filter out outliers of feature matches and estimate the spatial correlation between the near- or partial-duplicate images. We justified the performance of ECOTA in general for benchmarks of various structures. We applied it to identify the overlapping between images. For this specific case, we defined particular criteria in Subsection 7.7.4 to make ECOTA proficient in identifying the correlation between the overlapped images. This property of ECOTA could be applied to complete the registration of image (i.e. to build panoramas).

To improve the detection of Copied-modified-moved object, the ECOTA-duplicate approach (presented in Chapter 8) can be improved by using an additional type of feature such as engaging the color histogram of the candidate object or areas, as discussed in Subsection 6.3. In this case, the building of the color histogram is accomplished after defining the boundary of the candidate infringed areas utilizing the hypothesis of the ECOTA. Based on this, ECOTA-duplicate can be improved to detect multiple copies of an object or area in an image.

To improve the detection of ND-images by applying neural network, we can apply ECOTA in pre-processing step to guide a neural network to predict the transformation between the ND-images with a minimum error. Supplying only the inliers set of feature matches reduces the costs of training stage since only the outliers are excluded before starting the training stage. Similar idea has been applied in [35] by applying the RANSAC model to guide a neural network in estimating the transformation between the ND-images.

Appendices

A Affine Transformation Matrix

Figure 66 presents an example of all possible affine transformations of an image that contains the letter F

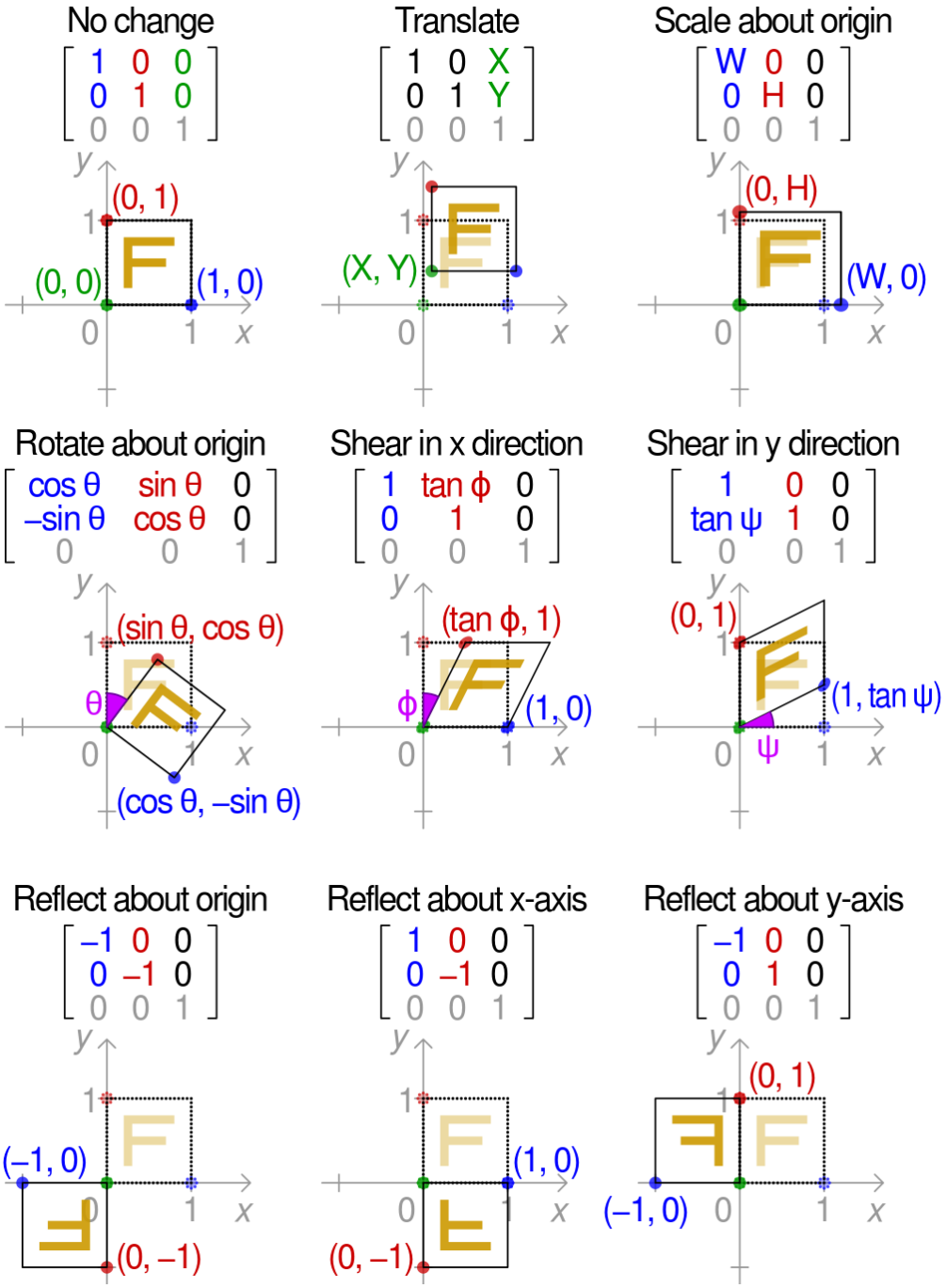


Figure 66: Various forms of image affine transformations [48].

B Comparison of Keypoint Detectors and Descriptors

To clarify the reason of selecting the SIFT keypoint detector and descriptor, we present in this appendix a comparison between some kinds of gradient and binary keypoint detectors and descriptors. We select the SURF of the gradient keypoints and BRIEF, ORB and BRISK of the binary ones. To apply the comparison, we employ the benchmark presented in [5], which contains five types of changes: i.e. viewpoint change, blurring, rotation, illumination change, and JPEG compression (the details of the Homography benchmark dataset are presented in Section 4.1). The keypoints of all of these images are extracted employing all of the selected keypoints detectors and descriptors. We analyze the performance of the SIFT, SURF, BRIEF, ORB, and BRISK employing various threshold. The keypoints

Table 53: The retrieval performance of the ***SIFT*** algorithm employing various values for its parameters and the zoomed and rotated boat images. The results present the comparison of the original image (img1) with the transformed images (img2, img3, img4, img5 and img6). The O,C,E, & G refer to the number of octaves, contrast threshold, edge threshold and the Gaussian filter respectively.

O	C	E	G	sim1,2	sim1,3	sim1,4	sim1,5	sim1,6	Average
7	0.04	10	1.6	57.28	56.86	56.36	56.49	55.74	56.54
5	0.04	10	1.6	56.83	57.58	57.29	57.43	56.26	57.08
3	0.04	10	1.6	58.98	58.38	58.25	58.77	56.62	58.20
3	0.08	10	1.6	48.63	53.52	49.12	51.91	47.28	50.09
3	0.12	10	1.6	27.90	34.62	43.66	49.64	35.03	38.17
3	0.04	6	1.6	59.42	57.83	58.53	58.98	57.18	**58.38**
3	0.04	16	1.6	59.21	58.38	58.35	57.89	56.51	58.06
3	0.04	10	1.3	57.63	57.01	56.27	55.82	55.26	56.41
3	0.04	10	2.4	61.02	58.01	57.74	57.46	56.11	58.07
3	0.04	10	2.4	62.18	57.20	57.07	56.96	56.00	57.88

B.1 Performance of the SIFT Algorithm

We evaluated the performance of the SIFT algorithm employing various values for its octave, contrast threshold, edge threshold and Gaussian filter parameters. We repeated the experiments using 7, 5 and 3 octaves. More experiments were done employing the contrast thresholds: 0.12, 0.08 and 0.04 and edge thresholds: 16, 10 and Gaussian filter of sizes 2.4, 1.6 and 1.3. The similarity between img1 and img2,...,

img6 is computed by dividing the keypoints matches by the number of keypoints in the query (i.e. img1). The results of comparing the similarity between img1 and the transformed images of the *zoom and rotation bark image* (the seventh row in Figure 27) are presented in Table 53. We denote these similarities as *sim1,2, sim1,3, sim1,4, sim1,5 and sim1,6*. The results in the average column show that the best performance of the SIFT algorithm is obtained employing three octaves, value of 0.04 for the contrast threshold, value 6 for edge threshold and Gaussian filter of size 1.6.

B.2 Performance of the SURF Algorithm

To evaluate the performance of the SURF algorithm in detecting the similarity between the transformed images of [5], we use Hessian threshold 10, 100, 200, 700, and 1000 for edge detection. We set the number of octaves to 1, 4 and 10. Table 54 presents that the best performance is obtained for the Hessian threshold 1000 utilizing four octaves. In the original implementation of the SURF algorithm the recommended value of the Hessian threshold is lesser or equal 500 but this threshold produces huge amount of keypoints, therefore we employed bigger values.

Table 54: The retrieval performance of the ***SURF*** algorithm employing various values for its parameters and the zoomed and rotated boat images. The results present the comparison of the original image (img1) with the transformed images (img2, img3, img4, img5 and img6).

Octaves	Hessian threshold	sim1,2	sim1,3	sim1,4	sim1,5	sim1,6	Average
1	100	55.10	52.30	51.65	52.88	52.18	52.82
4	100	55.14	53.23	51.13	51.94	51.79	52.65
10	100	55.15	53.29	51.12	51.96	51.83	52.67
4	10	54.92	52.14	51.59	51.50	51.28	52.29
4	200	55.04	54.06	51.64	52.72	51.95	53.17
4	700	55.09	54.51	52.16	53.04	53.95	53.75
4	1000	57.11	53.45	53.36	54.83	53.81	**54.51**

B.3 Performance of the ORB Algorithm

To evaluate the performance of the ORB algorithm, we employ scale thresholds : 1.2, 1.5,2.0 and the FAST thresholds 15, 20, 250 for corners detection. We set the number of octaves to 1, 4 and 10. As shown in Table 55, the best result is found for the scale threshold 2.0 and FAST threshold 20.

Table 55: The retrieval performance of the ***ORB*** algorithm employing various values for its parameters and the zoomed and rotated boat images.

Scale	FAST threshold	sim1,2	sim1,3	sim1,4	sim1,5	sim1,6	Average
1.2	20	51.60	51.40	54.60	55.20	53.40	53.24
1.5	20	53.59	54.74	53.04	55.87	55.24	54.50
2.0	20	53.78	54.75	57.33	54.46	56.95	**55.46**
1.2	15	51.47	51.32	53.98	54.70	53.15	52.92
1.2	25	51.60	51.40	54.60	55.20	53.40	53.24

B.4 Performance of the BRIEF Algorithm

The BRIEF algorithm builds only descriptors therefore, we construct the BRIEF descriptors employing SIFT features. We evaluate the performance of the BRIEF algorithm using descriptors of various lengths: i.e. 16, 32, and 64. Table 56 presents that the descriptors of length 16 obtains the best performance.

Table 56: The retrieval performance of the ***BRIEF*** algorithm employing descriptors of length 16, 32, and 64 and the zoomed and rotated boat images.

Descriptor length	sim1,2	sim1,3	sim1,4	sim1,5	sim1,6	Average
16	53.10	51.93	53.04	54.43	54.48	**53.44**
32	53.25	53.01	52.79	53.25	54.04	53.27
64	54.77	51.96	52.52	52.85	52.25	52.93

B.5 Performance of the BRISK Algorithm

We evaluate the performance of the BRISK algorithm employing 3, 4, and 6 octaves and using the corner thresholds: 5, 10, 20, and 30. Table 57 shows that the best performance is detected employing three octaves and corner threshold of value 5.

B.6 Comparison of SIFT, SURF, ORB, BRIEF & BRISK Algorithms

We compare the performance of the SIFT, SURF, ORB, BRIEF, and BRISK algorithms for all transformations types shown in Figure 27. The comparison is done employing the parameters that obtain the best performance for each algorithm[1]. Table 58 presents that the SIFT and next ORB algorithms outperform the other algorithms in cases of *adding blur*. Whereas, the SIFT and next BRISK algorithms

[1]These result are found as part of project and are not published in any conference or journal.

Table 57: The retrieval performance of the ***BRISK*** algorithm employing various values for its parameters and the zoomed and rotated boat images.

Octaves	Corner threshold	sim1,2	sim1,3	sim1,4	sim1,5	sim1,6	Average
3	30	50.17	55.16	52.22	52.73	48.61	51.78
4	30	51.19	55.53	52.57	52.54	49.10	52.19
6	30	51.19	55.51	52.57	52.54	49.14	52.19
3	5	54.98	54.80	54.10	53.70	53.84	54.28
3	10	54.78	54.98	54.48	54.10	53.48	54.36
3	20	57.83	56.03	56.03	54.84	54.52	**55.85**

obtain the best performance in case of *zoom and rotation* and *viewpoint change* transformations. When the *JPEG compression* is employed. When the *lighting* of images decrease, the best results are found by the BRIEF followed by the SIFT algorithms. Form these results we find out that the SIFT algorithm obtains at most one of the best results i.e. the SIFT keypoints are more robust to various types of image transformation.

Table 58: Comparison results of the SIFT, SURF, ORB, BRIEF & BRISK algorithms employing set of image of various types of transformations [5].

Transformation type	SIFT	SURF	ORB	BRIEF	BRISK
Blur Bikes	**64.69**	56.87	**62.30**	45.26	55.42
Blur Trees	**63.56**	57.21	**63.23**	53.79	60.47
JPEG compression ubc	71.90	**74.18**	**78.77**	64.93	69.87
Light leuven	**64.87**	66.00	64.11	**67.05**	60.89
Viewpoint graf	**64.81**	61.29	62.97	60.44	**64.58**
Viewpoint wall	**67.72**	63.35	63.47	62.58	**65.71**
Zoom and rotation bark	**58.38**	54.51	55.46	53.44	**55.85**
Zoom and rotation boat	**61.00**	58.11	57.94	56.14	**59.08**

C Comparison of SIFT and RC-SIFT Involving various Weights

In Section 5.2.6, we displayed the results of employing the scale and contrast properties to truncate the list of extracted keypoints and involving weights that their values depend on the contrast, scale, and orientation properties. In this appendix, we present the results involving various values of the weight W. Table 59 presents that the performance of both SIFT and RC-SIFT decreases when the value of W decreases. The best results are achieved when $W = 0.9$. Table 60 displays that the best mean average precision and the least variance of recall are obtained when $W = 0.9$. The Caltech-Buildings benchmark is employed to present the results.

Table 59: The retrieval performance of SIFT$-128D$ and RC-SIFT$-64D$ employing *various initial weight values*. The lists of features are ranked and truncated based on their ***contrast property***. The results are presented using the *Caltech-Buildings* benchmark.

Weight	Descriptors properties		SIFT$-128D$		RC-SIFT-64	
	Contrast	Orientation	MR4	MR10	MR4	MR10
0.9	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**44.21**	**57.22**	**44.85**	**58.00**
0.7	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	33.00	41.88	34.67	43.33
0.5	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	28.00	39.30	28.66	40.00
0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	24.21	35.58	25.84	36.00

Table 60: The mean average precision and variance of recall of theSIFT$-128D$ and RC-SIFT$-64D$ when *various weight values* are employed. The lists of features are ranked and truncated based on their ***contrast property***. The results are presented using the *Caltech-Buildings* benchmark.

Weight	Descriptors properties		SIFT$-128D$		RC-SIFT-64	
	Contrast	Orientation	MAP	VR	MAP	VR
0.9	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	**41.02**	8.08	**41.78**	8.06
0.7	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	29.84	9.14	30.42	8.88
0.5	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	23.20	10.02	24.68	9.41
0.1	$\Delta Cont$<0.1	$\Delta Ori < \frac{\pi}{10}$	19.32	10.33	20.08	9.69

D ECOTA-Duplicate: Split Feature Matches into two Lists

Let N be the number of detected feature matches between I and $I^{'}$, $LP_{original}$ and $LP^{'}_{duplicate}$ are the lists of matches location in I and $I^{'}$, respectively. We aim to split the list of N feature matches into two lists $M_{original}$, the list of feature matches including the original object and the background of I and $I^{'}$, and $M_{duplicate}$, the list of feature matches between the original object in I and the duplicate object in $I^{'}$. Given are pairs $(P_i, P^{'}_i)$ and $(P_j, P^{'}_j)$ of features matches between images I and $I^{'}$, we determine whether $P^{'}_i$ and $P^{'}_j$ belongs to the original or duplicate object in image $I^{'}$ by applying the splitting algorithm 2. The details of the splitting algorithm are detailed as follows:
In all of our experiments we set the parameter ϵ, given in algorithm 2, to $\epsilon = 1$ i.e., we allow localization error of one pixel in all directions (since the localization is done based on the SIFT).

Algorithm 2 Convert feature matches into two lists

Require: in N, $LP_{original}$, $LP'_{duplicate}$
Ensure: out $M_{original}$, $M_{duplicate}$
  for $i = 1$ to $N - 1$ **do**
    for $j = i + 1$ to N **do**
      pick two pair matches (P_i, P_j), (P'_i, P'_j) *from* $LP_{original}$, $LP'_{duplicate}$
      if $dis(P_i, P_j) \mp \epsilon < dis(P'_i, P'_j)$ **then**
        if $\mid \vec{OP_i} \mid = \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \neq \mid \vec{OP'_j} \mid$ **then**
          $M_{original} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon \neq \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon = \mid \vec{OP'_j} \mid$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{original} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon \neq \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon \neq \mid \vec{OP'_j} \mid$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid = \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid = \mid \vec{OP'_j} \mid$ **then**
          $M_{original} \leftarrow (P_i, P'_i)$ *AND* $M_{original} \leftarrow (P_j, P'_j)$
        end if
      else if $dis(P_i, P_j) = dis(P'_i, P'_j)$ **then**
        if $\mid \vec{OP_i} \mid = \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid = \mid \vec{OP'_j} \mid$ **then**
          $M_{original} \leftarrow (P_i, P'_i)$ *AND* $M_{original} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon \neq \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon \neq \mid \vec{OP'_j} \mid$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        end if
      else if $dis(P_i, P_j) \mp \epsilon \geq dis(P'_i, P'_j)$ **then**
        if $\mid \vec{OP_i} \mid \mp \epsilon = \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon \neq \mid \vec{OP'_j} \mid$ **then**
          $M_{original} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon \neq \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon = \mid \vec{OP'_j} \mid$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{original} \leftarrow (P_j, P'_j)$
        else if $(\mid \vec{OP_i} \mid \mp \epsilon < \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon < \mid \vec{OP'_j} \mid)$
          OR $(\mid \vec{OP_i} \mid \mp \epsilon > \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon > \mid \vec{OP'_j} \mid)$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon \neq \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon \neq \mid \vec{OP'_j} \mid$ **then**
          $M_{duplicate} \leftarrow (P_i, P'_i)$ *AND* $M_{duplicate} \leftarrow (P_j, P'_j)$
        else if $\mid \vec{OP_i} \mid \mp \epsilon = \mid \vec{OP'_i} \mid$ AND $\mid \vec{OP_j} \mid \mp \epsilon = \mid \vec{OP'_j} \mid$ **then**
          $M_{original} \leftarrow (P_i, P'_i)$ *AND* $M_{original} \leftarrow (P_j, P'_j)$
        end if
      end if
      $i \leftarrow i + 1$
    end for
  end for
  return $M_{original}$, $M_{duplicate}$

References

[1] Art uk is a cultural education charity. bbc – your paintings. http://www.bbc.co.uk/arts/yourpaintings/. [Online; accessed July-2020]. (cited on pages 52, 58, and 132)

[2] Panorama images. http://en.wikipedia.org/wiki/User:Diliff#/media. [Online; accessed August-2018]. (cited on pages 52, 55, and 119)

[3] *The Challenges of clustering high dimensional data*. New Vistas in Statistical Physics-Applications in Econophysics, Bioinformatics, and Pattern Recognition. Wille LT, editor, 2004. (cited on pages 63 and 84)

[4] Color spaces. https://commons.wikimedia.org/wiki/User:Datumizer, October 2020. [Online; accessed October-2020]. (cited on pages x and 12)

[5] The visual geometry group. affine covariant features. https://www.robots.ox.ac.uk/ṽgg/research/affine/, January 2021. [Online; accessed January-2021]. (cited on pages xi, xii, xxiii, 25, 26, 28, 51, 53, 61, 162, 163, and 165)

[6] The visual geometry group. oxford buildings benchmark. https://www.robots.ox.ac.uk/ṽgg/data/oxbuildings/, January 2022. [Online; accessed Febraury-2022]. (cited on page 54)

[7] S. Agarwal, A. Verma, and P. Singh. Content based image retrieval using discrete wavelet transform and edge histogram descriptor. *Proceedings of the 2013 International Conference on Information Systems and Computer Networks, ISCON 2013*, pages 19--23, 2013. (cited on pages 39 and 40)

[8] S. Agarwal, A. K. Verma, and N. Dixit. Content based image retrieval using color edge detection and discrete wavelet transform. *2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, pages 368--372, 2014. (cited on pages 39 and 40)

[9] A. Akansu, R. Haddad, and H. Caglar. Perfect reconstruction binomial qmf-wavelet transform. *Proceeding of SPIE Visual Communications and Image Processing*, 1360:609–618, 1990. (cited on page 39)

[10] M. Aly, P. Welinder, M. Munich, and P. Perona. Towards automated large scale discovery of image families. *Computer Vision and Pattern Recognition Second IEEE Workshop (CVPR)*, page 9–16, 2009. (cited on pages 51, 54, and 68)

[11] M. Aly, P. Welinder, M. Munich, and P. Perona. Caltech-building benchmark. http://www.vision.caltech.edu/malaa/datasets/caltech-buildings/, September 2015. [Online; accessed September-2015]. (cited on pages 53, 68, 90, and 91)

[12] A. A. Alyosef, C. Elias, and A. Nürnberger. Localization and transformation reconstruction of image regions: An extended congruent triangles approach.

2020 25th International Conference on Pattern Recognition (ICPR), pages 241--248, 2021. (cited on pages 52, 57, 58, 112, 113, 124, 126, 132, and 133)

[13] A. A. Alyosef and A. Nürnberger. Adapted SIFT descriptor for improved near duplicate retrieval. *In proceedings of the 5th International Conference on Pattern Recognition Applications and Methods ICPRAM*, pages 55--64, 2016. (cited on pages 51, 53, 62, 69, 78, 80, 81, 82, 87, 95, 97, 102, 105, and 136)

[14] A. A. Alyosef and A. Nürnberger. The effect of SIFT features properties in descriptors matching for near-duplicate retrieval tasks. *In proceedings of the 6th International Conference on Pattern Recognition Applications and Methods ICPRAM*, pages 703--710, 2017. (cited on pages 51, 53, 54, 62, 87, 94, 95, 102, and 119)

[15] A. A. Alyosef and A. Nürnberger. Near-duplicate retrieval: A benchmark study of modified SIFT descriptors. *Springer*, pages 121--138, 2017. (cited on pages 51, 53, 54, 62, 78, 80, 81, 95, 97, and 102)

[16] A. A. Alyosef and A. Nürnberger. Detecting sub-image replicas: Retrieval and localization of zoomed-in images. *Computer Analysis of Images and Patterns: International Conference on Computer Analysis of Images and Patterns (CAIP), Springer*, pages 257--268, 2019. (cited on pages 52, 56, 102, 105, 112, 113, 115, and 130)

[17] A. A. Alyosef and A. Nürnberger. Hybrid fuzzy binning for near-duplicate image retrieval: Combining fuzzy histograms and SIFT keypoints. *In proceedings of the 8th International Conference on Pattern Recognition Applications and Methods ICPRAM*, 241-248:241--248, 2020. (cited on pages 52, 53, 96, 98, 102, 103, 105, and 110)

[18] I. Amerini, L. Ballan, R. Caldelli, A. Bimbo, L. D. Tongo, and G. Serra. Copy-move forgery detection and localization by means of robust clustering with j-linkage. *Signal Processing: Image Communication Journal*, 28:659--669, 2013. (cited on pages xv and 140)

[19] K. Arai and A. Barakbah. Hierarchical k-means: An algorithm for centroids initialization for k-means. *Reports of the Faculty of Science and Engineering*, 36:25--31, 2007. (cited on page 29)

[20] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. (cited on pages 52 and 54)

[21] E. Ardizzone, A. Bruno, and G. Mazzola. Copy-move forgery detection by matching triangles of keypoints. *IEEE Transactions on Information Forensics and Security*, 10, Issue 10:2084--2094, 2015. (cited on pages xxii, 44, 59, 145, 146, 147, 148, 149, 150, 151, 153, and 155)

[22] K. Arthi and J. Vijayaraghavan. Content based image retrieval algorithm using color models. *International Journal of Advanced Research in Computer and Communication Engineering*, 2, Issue 3:35--42, 2013. (cited on page 36)

[23] A. Auclair, N. Vincent, and L. Cohen. Hash functions for near duplicate image retrieval. *Applications of Computer Vision (WACV)*, pages 1--6, 2009. (cited on pages 29 and 65)

[24] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. *European Conference on Computer Vision (ECCV 2014). Lecture Notes in Computer Science Springer*, 8689:584--599, 2014. (cited on pages 52 and 54)

[25] J. Baber, M. Bakhtyar, W. Noor, A. Basit, and I. Ullah. Performance enhancement of patch-based descriptors for image copy detection. *International Journal of Advanced Computer Science and Applications*, 7(3):449–456, 2016. (cited on pages 3 and 4)

[26] H. Bay, A. Ess, T. Tuytelaars, and L. Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110 n.3:346--359, 2008. (cited on pages 10, 24, 25, and 30)

[27] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. *Proceeding of European Conference on Computer Vision*, 110:407--417, 2006. (cited on pages 21, 23, 24, 25, 30, 36, 62, 69, 84, 85, 86, and 157)

[28] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18 (9):509–517, 1975. (cited on page 29)

[29] T. Berk, A. Kaufman, and L. Brownston. A human factors study of color notation systems for computer graphics. *Communications of the ACM*, 25(8):547–550, 1982. (cited on page 97)

[30] R. Biruté and F. Wolfgang. Ransac for outlier detection. *Geodesy and Cartography*, 31:83--87, 2005. (cited on page 112)

[31] J. M. Bland and D. G. Altman. *Statistics notes: measurement error*. BMJ, 1996. (cited on page 33)

[32] D. Bora, A. Gupta, and F. Khan. Performance of l*a*b* and hsv color spaces with respect to color image segmentation. *International Journal of Emerging Technology and Advanced Engineering*, 5, Issue 2:192--203, 2015. (cited on pages 36, 97, and 98)

[33] A. Bosch, A. Zisserman, and X. M. oz. Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2008. (cited on pages 38, 39, and 96)

[34] H. Bouma, A. Vilanova, J. Bescòs, B. Romeny, and F. Gerritsen. Fast and accurate gaussian derivatives based on b-splines. *Proceedings of the 1st International Conference on Scale Space and Variational Methods in Computer*

Vision. Springer-Verlag, Berlin, Heidelberg, 29:406–417, 2007. (cited on page 12)

[35] E. Brachmann and C. Rother. Neural-guided ransac: Learning where to sample model hypothesis. *In Proceeding of the IEEE International Conference on Computer Vision (ICCV)*, 2019. (cited on page 159)

[36] P. S. Bradley and U. M. Fayyad. Refining initial points for k-means clustering. *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 91--99, 1998. (cited on page 29)

[37] G. Bradski and A. Kaehler. Learning opencv: Computer vi-sion with the opencv library. *O'Reilly Media, Inc*, page 201–206, 2008. (cited on page 32)

[38] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993. (cited on page 45)

[39] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. *In European Conference on Computer Vision*, 2010. (cited on page 10)

[40] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Macbine Intelligence*, PAMI-8, No.6, 1986. (cited on page 39)

[41] A. Chadha, S. Mallik, and R. Johar. Comparative study and optimization of feature-extraction techniques for content based image. *International Journal of Computer Applications*, 52, Issue 20:35--42, 2012. (cited on page 36)

[42] Y. K. Chan and C. Y. Chen. Image retrieval system based on color-complexity and color-spatial features. *Assam University Journal of science & Technology*, 71:65--70, 2004. (cited on page 11)

[43] L. Chu, S. Jiang, S. Wang, Y. Zhang, and Q. Huang. Robust spatial consistency graph model for partial duplicate image retrieval. *IEEE Transactions on Multimedia*, 15:1982--1996, 2013. (cited on pages 41 and 43)

[44] O. Chum and J. Matas. Matching with prosac - progressive sample consensus. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, page 220–226, 2005. (cited on pages 41 and 42)

[45] O. Chum and J. Matas. Randomized ransac with sequential probability ratio test. *10th IEEE International Conference on Computer Vision ICCV'05*, 1:1727–1732, 2005. (cited on pages 41 and 42)

[46] O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. *In Proc. CIVR*, 2007. (cited on pages 3, 29, 36, and 65)

[47] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. *British Machine Vision Conference*, 2008. (cited on pages 29 and 65)

[48] Cmglee. Color spaces. https://commons.wikimedia.org/wiki/User:Cmglee, January 2021. [Online; accessed January-2021]. (cited on pages xvi, 10, and 161)

[49] D. Cohen. *Precalculus: A Problems-Oriented Approach*, volume 69 (4). (6th ed.), Cengage Learning, 2004. (cited on page 30)

[50] F. Crow. Summed-area tables for texture mapping. *Proceedings of SIGGRAPH*, 18(3):207–212, 1984. (cited on page 22)

[51] E. J. Crowley and A. Zisserman. The state of the art: Object retrieval in paintings using discriminative regions. *British Machine Vision Conference*, 2014. (cited on pages 52, 57, 58, and 132)

[52] Ş. Işik and K. Özkan. A comparative evaluation of well-known feature detectors and descriptors. *International Journal of Applied Mathematics, Electronics and Computers*, 3:1–6, 2015. (cited on pages 51 and 52)

[53] E. Cuevas and M. Díaz. A method for estimating view transformations from image correspondences based on the harmony search algorithm. *Computational Intelligence and Neuroscience*, 2015. (cited on pages 41 and 112)

[54] S. Damelin and W. Miller. The mathematics of signal processing. *Cambridge University Press*, 2011. (cited on page 13)

[55] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hash-ing scheme based on p-stable distributions. *Proceedings of the Sympo-sium on Computational Geometry*, page 253–262, 2004. (cited on page 29)

[56] D. Defays. An efficient algorithm for a complete-link method. *The Computer Journal. British Computer Society*, 20 (4):364–366, 1973. (cited on page 29)

[57] P. J. Diggle. A kernel method for smoothing point process data. *Journal of the Royal Statistical Society, Series C*, 34 (2):138–147, 1985. (cited on page 13)

[58] M. Douze, H. Jègou, H. Singh, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. *Proceeding of Image and Video Retrieval Conference*, 2009. (cited on page 36)

[59] X. Duanmu. Image retrieval using color moment invariant. *Seventh International Conference on Information Technology*, pages 200--203, 2010. (cited on page 36)

[60] M. P. Dubuisson and A. K. Jain. Fusing color and edge information for object matching. *IEEE Computer Society proceedings of 1st International Conference on Image Processing*, pages 982--986, 1994. (cited on pages 30 and 31)

[61] R. Dyer, H. Zhang, and T. Möller. A survey of delaunay structures for surface representation. 2009. (cited on page 44)

[62] P. Ee and P. Report. Histogram-based color image retrieval. *Image Rochester NY*, page 1–21, 2008. (cited on page 40)

[63] C. Elias. Detecting image replicas: Retrieval and localization of sub-images, intersect images and affine transformed images. Master's thesis, Otto-von-Guericke-Universität Magdeburg, 2020. (cited on page 123)

[64] H. Fan, L. Zheng, C. Yan, and Y. Yang. Unsupervised person re-identification: Clustering and fine-tuning. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 14 Issue 4:1551--6857, 2018. (cited on page 45)

[65] W. Feller. *An introduction to probability theory and its applications*. Wiley. Retrieved 10 August 2012, 1971. (cited on page 37)

[66] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24-Issue 6:381--395, 1981. (cited on page 41)

[67] D. Fleck and Z. Duric. Affine invariant-based classification of inliers and outliers for image matching. *Proceedings of the 6th International conference on Image Analysis and Recognition ICIAR*, pages 268--277, 2009. (cited on page 40)

[68] D. Fleck and Z. Duric. Using local affine invariants to improve image matching. *20th International Conference on Pattern Recognition, Istanbul*, pages 1844--1847, 2010. (cited on page 40)

[69] L. Floriani and E. Puppo. A survey of constrained delaunay triangulation algorithms for surface representation. *Pieroni, G.G. (eds) Issues on Machine Vision. International Center for Mechanical Sciences, Springer*, 307, 1989. (cited on page 44)

[70] J. J. Foo and R. Sinha. Pruning SIFT for scalable near-duplicate image matching. *In Proc. ADC*, pages 63--71, 2007. (cited on pages 37 and 38)

[71] S. Foolad and A. Maleki. A bottom-up visual attention model based on background color feature. *Third Basic and Clinical Neuroscience Congress (BCNC 2014)*, 2014. (cited on page 40)

[72] R. Frigyes. Untersuchungen über systeme integrierbarer funktionen. *Mathematische Annalen*, 69 (4):449–497, 1910. (cited on page 30)

[73] L. E. Garner. *An Outline of Projective Geometry*. North Holland, 1981. (cited on page 9)

[74] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. *Proc. VLDB Int. Conf. on Very Large Data Bases*, pages 518--529, 1999. (cited on page 29)

[75] K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. *Proceeding of the International Conference on Computer Vision (ICCV)*, 2005. (cited on page 29)

[76] K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. *The Journal of Machine Learning Research*, 8:725--760, 2007. (cited on page 29)

[77] I. Guyon, M. Nikravesh, S. Gunn, and L. A. Zadeh. *Feature Extraction Foundations and Applications*. Springer-Verlag Berlin Heidelberg, 2006. (cited on page 7)

[78] A. Haar. Zur theorie der orthogonalen funktionensysteme (erste mitteilung). *Springer*, 1360:331–371, 1910. (cited on page 39)

[79] G. A. Hambunan, C. G. A. Fernandez, and E. P. Mendoza. Comparison on various image deformations based on match ratings using ORB, BRIEF, SURF and SIFT. *Proceedings of International Conference on Technological Challenges for Better World 2018 Performance*, 2018. (cited on page 37)

[80] J. Han and K. K. Ma. Fuzzy color histogram and its use in color image retrieval. *IEEE Transactions on Image Processing*, 2002. (cited on pages 36 and 98)

[81] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society, Series C.*, 28 (1):100–108, 1979. (cited on page 29)

[82] A. Hast and A. Marchetti. Putative match analysis: A repeatable alternative to RANSAC for matching of aerial images. *In Proceedings of the International Conference on Computer Vision Theory and Applications*, 2:341–344, 2012. (cited on pages 41, 43, and 112)

[83] A. Hast, J. Nysjö, and A. Marchetti. Optimal ransac-towards a repeatable algorithm for finding the optimal set. *Journal of WSCG*, 21:21--30, 2013. (cited on page 42)

[84] R. S. Hunter. Accuracy, precision, and stability of new photo-electric color-difference meter. *Proceedings of the Thirty-Third Annual Meeting of the Optical Society of America*, 38 (12): 1094, 1948. (cited on page 11)

[85] R. S. Hunter. Photoelectric color-difference meter. *Proceedings of the Winter Meeting of the Optical Society of America*, 38 (7): 661, 1948. (cited on page 11)

[86] C. Iakovidou, N. Anagnostopoulos, A. Kapoutsis, Y. Boutalis, M. Lux, and S. Chatzichristofis. Localizing global descriptors for content based image retrieval. *EURASIP Journal on Advances in Signal Processing*, 2015. (cited on page 52)

[87] T. Iijima. asic theory on normalization of pattern (in case of typical one-dimensional pattern). *Bulletin of the Electrotechnical Laboratory*, 26:368--388, 1962. (cited on page 8)

[88] R. J. Deng, W. Dong, K. L. L. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. in computer vision and pattern recognition.

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), page 248–255, 2009. (cited on pages 51 and 54)

[89] A. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, 1986. (cited on page 9)

[90] M. Jiang, S. Zhang, H. Li, and D. N. Metaxas. Computer-aided diagnosis of mammographic masses using scalable image retrieval. *Biomedical Engineering, IEEE Transactions on*, pages 783--792, 2015. (cited on pages 29, 65, and 87)

[91] P. Jin, G. Xia, F. Hu, Q. Lu, and L. Zhang. Aid++: An updated version of aid on scene classification. *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 4721--4724, 2018. (cited on pages 52 and 57)

[92] L. Juan and O. Gwun. A comparison of SIFT, PCA-SIFT and SURF. *Int. J. Image Process.*, pages 143--152, 2009. (cited on page 37)

[93] L. Kabbai, M. Abdellaoui, and . A. Douik. Image classification by combining local and global features. *The Visual Computer, Springer*, 35:679–693, 2019. (cited on page 7)

[94] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. *In Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, pages 345--457, 2004. (cited on page 15)

[95] T. Kailath. The divergence and bhattacharyya distance measures in signal selection. *IEEE Transactions on Communication Technology*, 15:52--60, 1967. (cited on pages 30 and 31)

[96] E. Karami, S. Prasad, and M. Shehata. Image matching using SIFT, SURF, BRIEF and ORB: Performance comparison for distorted images. *Newfoundland Electrical and Computer Engineering Conference*, 2015. (cited on page 37)

[97] S. Karl. *Precalculus: A Functional Approach to Graphing and Problem Solving*, volume 69 (4). Jones & Bartlett Publishers, 2013. (cited on page 30)

[98] S. Kaur and V. K. Banga. Content based image retrieval: Survey and comparison between rgb and hsv model. *International Journal of Engineering Trends and Technology (IJETT)*, 4, Issue 4:192--203, 2013. (cited on pages 36 and 97)

[99] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 506--513, 2004. (cited on pages 37, 61, and 97)

[100] Y. Ke, R. Sukthankar, and L. Huston. Efficient near-duplicate detection and sub-image retrieval. *4th Proceedings of the 12th annual ACM international conference on Multimedia*, page 869–87, 2004. (cited on page 29)

[101] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29:1153--1160, 1981. (cited on page 55)

[102] N. Khan, B. McCane, and G. Wyvill. SIFT and SURF performance evaluation against various image deformations on benchmark dataset. *Proceeding of Conference on Digital Image Computing Techniques and Applications*, pages 501--506, 2011. (cited on pages xi, xvii, 37, 38, 52, 61, 62, 64, 66, 67, 69, 74, 80, 81, 84, 85, 86, 97, 102, and 157)

[103] P. Kun. *Fundamentals of Probability and Stochastic Processes with Applications to Communications*. Springer, 2018. (cited on page 37)

[104] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference*, 2:2169--2178, 2006. (cited on page 29)

[105] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. *International Conference on Computer Vision (ICCV)*, pages 2548--2555, 2011. (cited on pages xi, 26, 28, 30, 36, and 132)

[106] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata. Detecting outliers: Do not use standard deviation around the mean use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49:764--766, 2013. (cited on page 117)

[107] J. Li, X. Qian, Q. Li, Y. Zhao, L. Wang, and Y. Tang. Mining near duplicate image groups. *Multimed Tools Appl*, 74 (2):655 --669, 2014. (cited on page 39)

[108] J. Li, X. Qian, Q. Li, Y. Zhao, L. Wang, and Y. Y. Tang. Mining near duplicate image groups. *Springer Science and Business Media New York*, 2014. (cited on page 29)

[109] T. Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21(2):224--270, 1994. (cited on pages 7 and 18)

[110] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, 1994. (cited on page 8)

[111] T. Lindeberg. Scale-space. *Wiley Encyclopedia of Computer Science and Engineering*, pages 2495--2504, 2008. (cited on page 7)

[112] R. Lior and O. Maimon. Clustering methods. *Data mining and knowledge discovery handbook. Springer US*, pages 321--352, 2005. (cited on page 29)

[113] S. Liu, M. Sun, L. Feng, Y. Liu, and J. Wu. Three tiers neighborhood graph and multi-graph fusion ranking for multi-feature image retrieval: A manifold aspect. *Computing Research Repository (CoRR)*, 2016. (cited on pages 51 and 52)

[114] Y. Long, G. Xia, S. Li, W. Yang, M. Y. Yang, X. X. Zhu, L. Zhang, and L. Deren. On creating benchmark dataset for aerial image interpretation: Reviews, guidances and million-aid. *IEEE Journal of Selected Topics in*

Applied Earth Observations and Remote Sensing, 14:4205--4230, 2021. (cited on pages 52 and 57)

[115] K. Loquin and O. Strauss. Fuzzy histograms and density estimation. *n: Lawry J. et al. (eds) Soft Methods for Integrated Uncertainty Modelling. Advances in Soft Computing. Springer*, 37:45--52, 2006. (cited on page 98)

[116] D. Lowe. Distinctive image features from scale-invariant keypoints. *Journal of Computer Vision*, pages 91--110, 2004. (cited on pages x, 8, 17, 18, 19, 20, 30, 36, 62, 63, 64, 69, 84, 85, 86, 102, and 132)

[117] D. MacKay. Information theory, inference and learning algorithms. chapter 20. an example inference task: Clustering. *Cambridge University Press*, page 284–292, 2003. (cited on page 29)

[118] S. Manimala and K. Hemachandran. Performance analysis of color spaces in image retrieval. *The Journal of Systems and Software*, 7 Number II:94--104, 2011. (cited on page 11)

[119] B. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Inc., 2002. (cited on page 39)

[120] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. (cited on page 32)

[121] P. Marin-Reyes, J. Lorenzo-Navarro, and M. C. Santana. Comparative study of histogram distance measures for re-identification. *The Computing Research Repository (CoRR)*, 2016. (cited on pages 31, 100, and 101)

[122] D. Marr. *Vision*. Freeman, 1982. (cited on page 9)

[123] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *British Machine Vision Conference, Cardiff, Wales*, pages 384--393, 2002. (cited on pages 15, 16, 51, and 52)

[124] Z. Mehmood, F. Abbas, T. Mahmood, and et al. Content-based image retrieval based on visual words fusion versus features fusion of local and global features. *Computer Engineering and Computer Science, Arab J Sci*, page 7265–7284, 2018. (cited on pages 39 and 40)

[125] T. Mehta and C. K. Bhensdadia. Near duplicate image retrieval using multilevel local and global convolutional neural network features. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 11(4), 2020. (cited on page 46)

[126] E. Michaelsen, W. von Hansen, M. Kirchhof, J. Meidow, and U. Stilla. Estimating the essential matrix: GOODSAC versus RANSAC. *In Proceedings of the Photogrammetric Computer Vision PCV*, 2:1--6, 2006. (cited on pages 41, 43, and 112)

[127] K. Mikolajczyk and C. Schmid. Anaffine invariant interest point detector. *In European Conference on Computer Vision (ECCV)*, pages 128--142, 2002. (cited on page 17)

[128] K. Mikolajczyk and C. Schmid. Harris-affine & hessian affine: Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63--86, 2004. (cited on pages 7, 15, 28, 51, 52, and 97)

[129] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1615--1630, 2005. (cited on pages 28 and 97)

[130] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision (IJCV)*, 65:43--72, 2005. (cited on pages 15, 28, 37, and 97)

[131] R. Miranda. *Algebraic Curves and Riemann Surfaces*. AMS Bookstore, 1995. (cited on page 9)

[132] D. Myatt, P. Torr, N. Slawomir, J. Bishop, and R. Craddock. Napsac: High noise, high dimensional robust estimation - it's in the bag. *In British Machine Vision Conference (BMVC)*, 2002. (cited on page 42)

[133] A. Nazir, R. Ashraf, T. Hamdani, and N. Ali. Content based image retrieval system by using hsv color histogram, discrete wavelet transform and edge histogram descriptor. *International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1--6, 2018. (cited on pages 39 and 40)

[134] F. Nielsen. Introduction to hpc with mpi for data science. chapter 8: Hierarchical clustering. *Springer*, 2016. (cited on page 29)

[135] D. Nistèr and H. Stewènius. Ukbench benchmark. http://www.vis.uky.edu/štewe/ukbench/. [Online; accessed September-2015]. (cited on pages 11, 52, 68, 90, and 92)

[136] D. Nistèr and H. Stewènius. Scalable recognition with a vocabulary tree. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2161--2168, 2006. (cited on pages 29, 51, 52, 65, 66, 68, 78, 80, 87, 90, and 103)

[137] M. S. Nixon and A. S. Aguado. *Feature Extraction & Image Processing for Computer Vision(Third Edition)*. Elsevier Ltd, 2012. (cited on page 7)

[138] A. Oliva. Gist of the scene. *In Neurobiology of Attention, L. Itti, G. Rees and J. K. Tsotsos (Eds.)*, pages 251--256, 2005. (cited on page 36)

[139] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal in Computer Vision*, 42:145--175, 2001. (cited on page 36)

[140] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Progress in Brain Research: Visual perception*, 155:23--36, 2006. (cited on page 36)

[141] S. Pattanaik and D. Bhalke. Efficient content based image retrieval system using mpeg-7 features. *nternational Journal of Computer Applications*, I. 53:19--24, 2012. (cited on pages 39 and 40)

[142] M. Pavan and M. Pelillo. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:167–172, 2007. (cited on page 44)

[143] J. Philbin, O.Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. (cited on pages 52, 54, 104, and 131)

[144] A. Pourreza and K. Kiani. A partial-duplicate image retrieval method using color-based sift. *24th Iranian Conference on Electrical Engineering (ICEE)*, pages 1410--1415, 2016. (cited on page 40)

[145] C. Prakash and S. Maheshkar. Copy-move forgery detection using dywt. *International Journal of Multimedia Data Engineering and Management*, 8:1--9, 2017. (cited on pages xv and 140)

[146] W. K. Pratt. *Digital image processing (4th ed.)*. John Wiley & Sons, Inc, 2007. (cited on page 12)

[147] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C: the art of scientific computing (2nd edition)*. New York, NY, USA: Cambridge University Press, 1992. (cited on page 17)

[148] J. M. S. Prewitt. Object enhancement and extraction. *Bernice Sacks Lipkin und Azriel Rosenfeld (Hrsg.): Picture Processing and Psychopictorics. Academic Press, New York*, page 75–149, 1970. (cited on page 12)

[149] F. Radenović, G. Tolias, and O. Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *European Conference on Computer Vision ECCV 2016. Springer International Publishing*, pages 3--20, 2016. (cited on page 45)

[150] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrievalwith no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41 Issue 7:1655--1668, 2019. (cited on page 45)

[151] G. Rafael and R. Woods. *Digital image processing (3th ed.)*. Upper Saddle River, New Jersey: Pearson Education, 2008. (cited on pages 12 and 13)

[152] Y. Ren. Indexing and searching for similarities of images with structural descriptors via graph-cuttings methods. *Computer science. University de Bordeaux*, page 201–206, 2014. (cited on page 30)

[153] E. Rosten and T. Drummond. Machine learning for highspeed corner detection. *In European Conference on Computer Vision*, 1, 2006. (cited on page 10)

[154] E. Rostenand and T. Drummond. Machine learning for high-speed corner detection. *Proceedings of the European Conference on Computer Vision (ECCV)*, 3951, 2006. (cited on page 27)

[155] P. J. Rousseeuw and A. M. Leroy. Robust regression and outlier detection. *New York Wiley*, 1987. (cited on page 41)

[156] C. Royer. *Simultane Optimierung von Produktionsstandorten, Produktionsmengen und Distributionsgebieten*. utzverlag, Wirtschafts- und Sozialwissenschaften, 2001. (cited on page 30)

[157] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. *In Proceeding of the IEEE International Conference on Computer Vision (ICCV)*, 13, 2011. (cited on page 10)

[158] M. W. Schwarz, W. B. Cowan, and J. C. Beatty. An experimental comparison of rgb, yiq, lab, hsv, and opponent color models. *ACM Transaction on Graphics*, page 123–158, 1987. (cited on page 97)

[159] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal. British Computer Society*, 16 (1):30–34, 1973. (cited on page 29)

[160] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, pages 21--30, 2014. (cited on page 45)

[161] M. Singha and K. Hemachandran. Content based image retrieval using color and texture. *Signal & Image Processing: An International Journal (SIPIJ)*, 3:39--57, 2012. (cited on page 11)

[162] A. R. Smith. Color gamut transform pairs. *SIGGRAPH Computer Graphics Association for Computing Machinery New York, NY, USA*, 37:12–19, 1978. (cited on page 11)

[163] I. Sobel. An isotropic 3x3 image gradient operator. *Presentation at Stanford A.I. Project 1968*, 02 2014. (cited on page 12)

[164] J. Sporring, M. Nielsen, L. Florack, and P. Johansen. *Gaussian Scale-Space Theory*. Kluwer Academic Publishers, 1997. (cited on page 8)

[165] M. Stricker and M. Orengo. Similarity of color images. *In SPIE Conference on Storage and Retrieval for Image and Video Databases III*, 2420:381--392, 1995. (cited on page 36)

[166] G. Tang, Z. Liu, and J. Xiong. Distinctive image features from illumination and scale invariant keypoints. *Multimedia Tools and Applications, Springer*, 78(01), 2019. (cited on page 52)

[167] S. A. K. Tareen and Z. Saleem. A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. *International Conference on Computing, Mathematics and Engineering Technologies – iCoMET*, 2018. (cited on page 37)

[168] B. Tordoff and D. Murray. Guided sampling and consensus for motion estimation. *Conference: Proceedings of the 7th European Conference on Computer ECCV Vision-Part I*, page 82–96, 2002. (cited on page 42)

[169] P. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, page 138–156, 2000. (cited on pages 41 and 42)

[170] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors - survey. *Foundations and Trends in Computer Graphics and Vision*, 3(1):1--110, 2008. (cited on pages 51 and 52)

[171] K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32, Issue 9:1582--1596, 2010. (cited on pages 38, 39, and 96)

[172] J. van de Weijer, T. Gevers, and A. Bagdanov. Boosting color saliency in image feature detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):150–156, 2006. (cited on page 38)

[173] P. A. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1:511--518, 2001. (cited on pages 23 and 24)

[174] B. Waggener. *Pulse Code Modulation Techniques*. Springer, ISBN 9780442014360. Retrieved 13 June 2020, 1995. (cited on page 30)

[175] I. Wald and V. Havran. On building fast kd-trees for ray tracing, and on doing that in o(n log n). *In: Proceedings of the 2006 IEEE Symposium on Interactive Ray Tracing*, page 61–69, 2006. (cited on page 29)

[176] H. Walker. *Studies in the History of the Statistical Method*. Baltimore, MD: Williams & Wilkins, 1931. (cited on page 33)

[177] X. Wang, M. Yang, T. Cour, S. Zhu, K. K. Yu, and T. Han. Contextual weighting for vocabulary tree based image retrieval. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 209--216, 2011. (cited on pages 51 and 52)

[178] Y. Wang, Z. Hou, K. Leman, T. N. Pham, T. Chua, and R. Chang. Combination of local and global features for near-duplicate detection. *Advances in Multimedia Modeling. Springer Berlin Heidelberg*, pages 328--338, 2011. (cited on pages xi and 40)

[179] J. Weickert, S. Ishikawa, and A. Imiya. Scale-space has been discovered in japan. *Journal of Mathematical Imaging and Vision*, 10(3):237--252, 1997. (cited on page 8)

[180] J. N. Wilson and G. X. Ritter. *Handbook of Computer Vision Algorithms in Image Algebra*. CRS Press LLC, 2001. (cited on page 7)

[181] A. P. Witkin. Scale-space filtering. *Proc. 8th Int. Joint Conf. Art. Intell., Karlsruhe, Germany*, pages 1019--1022, 1983. (cited on pages x, 8, and 9)

[182] F. S. Woods. *Higher Geometry*. Ginn and Co., 1922. (cited on page 9)

[183] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 25--32, 2009. (cited on page 42)

[184] Z. Wu, Q. Xu, S. Jiang, Q. Huang, P. Cui, and L. Li. Adding affine invariant geometric constraint for partial-duplicate image retrieval. *20th International Conference on Pattern Recognition ICPR.2010*, pages 842--845, 2010. (cited on pages 3 and 4)

[185] G. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, X. Lu, and L. Zhang. Aid: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55:3965 -- 3981, 2017. (cited on pages 52, 57, and 132)

[186] L. Xiangru and H. Zhanyi. Rejecting mismatches by correspondence function. *International Journal of Computer Vision*, 89:1--17, 2010. (cited on page 41)

[187] P. Xiao, N. C., B. Tang, S. Weng, and H. Wang. Efficient SIFT descriptor via color quantization. *2014 IEEE International Conference on Consumer Electronics*, pages 1--3, 2014. (cited on page 38)

[188] D. Xu, T. Cham, S. Yan, L. Duan, and S. Chang. Near duplicate identification with spatially aligned pyramid matching. *IEEE Trans. Circuits and Systems for Video Technology*, 20:1068--1079, 2010. (cited on pages 3, 4, and 29)

[189] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. (cited on page 29)

[190] W. Yang, L. Xu, X. Chen, F. Zheng, and Y. Liu. Chi-squared distance metric learning for histogram data. *Mathematical Problems in Engineering*, 2015. (cited on pages 30 and 31)

[191] Y. Yang and S. Newsam. Comparing SIFT descriptors and gabor texture features for classification of remote sensed imagery. *In Proceedings of the 15th IEEE on Image Processing, San Diego, USA*, pages 1852--1855, 2008. (cited on page 29)

[192] H. Yu, M. Li, H. J. Zhang, and J. Feng. Color texture moments for content-based image retrieval. *Proceedings of IEEE International Conference on Image Processing*, pages 929--932, 2002. (cited on page 36)

[193] C. Zhang, S. Wang, Q. Huang, J. Liu, C. Liang, and Q. Tian. Image classification using spatial pyramid robust sparse coding. *Pattern Recognition Letters*, pages 1046--1052, 2013. (cited on pages 29 and 65)

[194] D. Zhang and S. Chang. Detecting image near duplicate by stochastic attribute relational graph matching with learning. *in Proc. ACM Multimedia Conf.*, pages 877--884, 2004. (cited on page 4)

[195] S. Zhang, M. Yang, T. Cour, K. Yu, and D. Metaxas. Query specific rank fusion for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37:660--673, 2012. (cited on pages 51 and 52)

[196] Y. Zhang, H. Wu, and L. Cheng. Some new deformation formulas about variance and covariance. *Proceedings of 4th International Conference on Modelling, Identification and Control(ICMIC2012)*, page 987–992, 2012. (cited on page 33)

[197] Y. Zhang, Y. Zhang, J. Sun, H. Li, and Y. Zhu. Learning near duplicate image pairs using convolutional neural networks. *International Journal of Performability Engineering*, 14:168--177, 2018. (cited on pages 45, 46, 52, and 85)

[198] B. Zhou and L. Zhang. Scene gist: A holistic generative model of natural image. *Proceedings of the 9th Asian conference on Computer Vision. Springer LNCS*, pages 395 --404, 2010. (cited on page 36)

[199] A. Zisserman. *Notes on Geometric and Invariance in Vision.* British Machine Vision Association and Society for Pattern Recognition, Chap. 2, 1992. (cited on page 9)

[200] M. Zuliani. https://www.cs.tau.ac.il/t̃urkel/imagepapers/RANSAC4Dummies.pdf. RANSAC for Dummies. [Online; accessed February-2021]. (cited on page 42)