



# Security analysis of image copy detection systems based on SIFT descriptors

Thanh-Toan Do

► **To cite this version:**

Thanh-Toan Do. Security analysis of image copy detection systems based on SIFT descriptors. Image Processing. Université Rennes 1, 2012. English. <tel-00766932>

**HAL Id: tel-00766932**

**<https://tel.archives-ouvertes.fr/tel-00766932>**

Submitted on 19 Dec 2012

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



**THÈSE / UNIVERSITÉ DE RENNES 1**  
*sous le sceau de l'Université Européenne de Bretagne*

pour le grade de  
**DOCTEUR DE L'UNIVERSITÉ DE RENNES 1**

*Mention : Informatique*

**École doctorale Matisse**

présentée par

**Thanh-Toan DO**

préparée à l'unité de recherche IRISA – UMR6074  
Institut de Recherche en Informatique et Systèmes  
Aléatoires

**Security analysis  
of image copy  
detection systems  
based on SIFT  
descriptors**

**Thèse soutenue à Rennes  
le 27 septembre 2012**

devant le jury composé de :

**David GROSS-AMBLARD**

Professeur à l'Université de Rennes 1 / *Président*

**Michel CRUCIANU**

Professeur au Conservatoire National  
des Arts et Métiers / *Rapporteur*

**Patrick BAS**

Chargé de recherche CNRS / *Rapporteur*

**Gwenaël DOËRR**

Chargé de recherche Technicolor R&D /  
*Examineur*

**Patrick GROS**

Directeur de recherche INRIA / *Directeur de thèse*

**Ewa KIJAK**

Maître de conférence à l'Université de Rennes 1 /  
*Co-directrice de thèse*



# Acknowledgements

First and foremost, I sincerely thank my advisors Ewa Kijak, Laurent Amsaleg and Teddy Furon for continuous support to my research and for patient correction of my work. I always found their office open when I had questions. They always allowed me to speak freely. I have learnt a lot from them during these past three years, not only knowledge but also professionalism in the work. I would not have been able to complete this thesis without their guidance. Besides, I would like to thank Patrick Gros who gave me the chance to pursue a PhD thesis in the TEXMEX team-project and who provided me a wonderful work environment.

I would like to thank David Gross-Amblard, Michel Crucianu, Patrick Bas, and Gwenaël Doërr for accepting to be members of my PhD jury and for their insightful comments on my thesis.

I would like to express my most sincere gratitude to all members of TEXMEX research team for a great working atmosphere with lots of support they have daily shared with me. I have learned a lot from their motivation, enthusiasm, and immense knowledge.

A special thank and much gratitude to my parents and my wife Thi-Nu-Phi Nguyen. They have always been there for me, encouraging me and helping me. I could not have accomplished this thesis without having them on my side.

Lastly, to all of those acknowledged here, friends, teachers and colleagues, thank you for being you and letting me be a part of your lives.



# Contents

Acknowledgements	1
Contents	2
1 Introduction	5
1.1 Content-Based Retrieval usage at the turn of the century . . . . .	5
1.2 Focus of the thesis . . . . .	6
1.3 Structure of the thesis . . . . .	8
2 Security threats of CBIRS	9
2.1 What is security? . . . . .	9
2.2 Security threats of CBIRS . . . . .	11
2.2.1 Trust model . . . . .	11
2.2.2 Goals of the pirate . . . . .	12
2.2.3 Measurements . . . . .	12
2.2.4 The Knowledge of the pirate . . . . .	13
2.2.4.1 Kerckhoffs' Principle . . . . .	13
2.2.4.2 The Kerckhoffs principle and CBIRS . . . . .	14
2.2.4.3 Access to Oracles . . . . .	15
2.3 Summary . . . . .	16
3 State of the art	17
3.1 Overview of CBIRS . . . . .	17
3.1.1 Feature extraction . . . . .	18
3.1.2 Indexing . . . . .	19
3.1.3 False positive removal . . . . .	20
3.2 Security of contents . . . . .	20
3.2.1 Robustness attacks . . . . .	20
3.2.2 Security attacks . . . . .	22
3.2.3 Competitive technologies . . . . .	22
3.2.3.1 Some related applications . . . . .	25
3.3 Summary . . . . .	26

4	Overview of solutions	27
4.1	The experiment setup	27
4.1.1	Overview of our system and the evaluation criteria	27
4.1.2	Description	28
4.1.3	Indexing	32
4.1.4	The geometric verification	33
4.1.5	Dataset and queries	33
4.1.6	Security assumption	34
4.2	Three angles of attacks	34
4.2.1	Attacking the keypoint detection	34
4.2.2	Attacking the keypoint description	34
4.2.3	Attacking the false positive removal	35
4.3	Summary	35
5	Attacking keypoints	37
5.1	Reducing Scores by Keypoint Removal	37
5.1.1	Removal with Minimum Local Distortion (RMD)	38
5.1.2	Removal by smoothing	41
5.2	Forging New Keypoints	44
5.3	Properties of New Keypoints	46
5.4	Large Scale Experiments	48
5.4.1	Results	48
5.4.2	Analysis	50
5.4.3	Discussion	52
5.5	Summary	52
6	Attacking descriptors	55
6.1	SIFT: impact of keypoint orientation on keypoint descriptor	56
6.1.1	From orientation to descriptor	56
6.1.2	Changing Orientations Impacts Descriptors	56
6.2	SVM for Changing Orientations	57
6.3	Evaluation of the orientation attack	60
6.3.1	Ability to Change Orientations	61
6.3.2	Cropped Patches Center	63
6.3.3	Impact at Image-Level	65
6.3.4	Impact on Large-Scale Recognition	66
6.4	Combination of Attack	67
6.4.1	Global System	68
6.4.2	Experiments With Multiple Voting and Single Voting	71
6.4.3	Geometric Verification	71
6.5	Summary	72

7	Attacking geometry	75
7.1	PiP Visual Modifications	76
7.1.1	Producing PiP Visual Modifications	76
7.1.2	Recognizing Contents Despite PiP	76
7.1.2.1	Prior Knowledge for Separating Images	77
7.1.2.2	Identification Without Prior Knowledge	79
7.2	CBIRS-aware PiP Creation	80
7.2.1	Image Washing Before PiP	80
7.2.2	Creating the PiP Attack	81
7.2.2.1	Determining Candidate Visual Distractors	82
7.2.2.2	Inserting a Visual Distractor	83
7.2.2.3	Blurring Boundaries	86
7.2.2.4	Inserting Multiple Visual Distractor	87
7.3	Large-Scale Experiments	88
7.3.1	Experiment 1: Inserting One Distractor	88
7.3.1.1	Effectiveness of the Attacks	88
7.3.1.2	Details on the distractors	91
7.3.2	Experiment 2: Inserting Two Distractors	91
7.4	Summary	92
8	Conclusions and Perspectives	93
8.1	Conclusions	93
8.2	Perspectives	94
9	Résumé étendu	97
9.1	Description du problème	97
9.1.1	Objectif de la thèse	97
9.1.2	Structure du manuscrit	98
9.2	Chapitre 2 : menaces sur la sécurité des CBIRS	99
9.3	Chapitre 3 : état de l'art	100
9.3.1	Vue générale des CBIRS	100
9.3.2	Sécurité du contenu	101
9.4	Chapitre 4 : contexte expérimental et survol des contributions	103
9.4.1	Critère d'évaluation	103
9.4.2	Description de l'image, indexation et faux positif	103
9.4.3	Dataset et les requêtes	104
9.4.4	Trois manières d'attaquer les images et de créer des quasi-copies	104
9.5	Chapitre 5 : attaque des points d'intérêts	104
9.5.1	Réduction du scores par suppression des points d'intérêts	105
9.5.2	Création de nouveaux points d'intérêts	106



9.5.3	Évaluation des attaques . . . . .	106
9.6	Chapitre 6 : attaque du descripteur . . . . .	107
9.6.1	Impact du changement de l'orientation sur les descripteurs	107
9.6.2	Utilisation d'un SVM pour modifier les orientations . . . .	107
9.6.3	Évaluation . . . . .	109
9.7	Chapitre 7 : prendre en compte la cohérence géométrique . . . . .	109
9.7.1	Lavage d'image . . . . .	110
9.7.2	Création de l'attaque PiP . . . . .	110
9.7.3	Frontières floues . . . . .	111
9.7.4	Évaluation de la méthode . . . . .	111
9.8	Chapitre 8: Conclusions et perspectives . . . . .	112
9.8.1	Conclusions . . . . .	112
9.8.2	Perspectives . . . . .	112
	Bibliographie	123
	List of figures	125
	List of tables	125
	Contents	

# Chapter 1

## Introduction

### 1.1 Content-Based Retrieval usage at the turn of the century

Content-Based Retrieval (CBR) is the generic term describing the problem of searching for digital material in large multimedia databases. CBR systems are of great diversity: they deal with a plurality of media (text, still images, music, videos) and offer a wide range of querying modes, from simple query-by-example schemes to more complex search processes involving user feedback aiming at best bridging the semantic gap. In a way, CBR systems promote the cultural and historical value of multimedia contents. They make the multimedia databases very useful, their contents reusable, spurring the enrichment of the artistic and cultural patrimony. CBR has proved to be a marvelous technology, recognizing content even when severely distorted.

Here are two recent applications of such content recognition that reached the public and are becoming quite popular: anyone can download a small application for Apple's iPhone that does song recognition. It uses low level audio signatures and content-based similarity searches. Also, anyone can now create superb image panoramas with fully automatic geometrical and color corrections. It uses local image descriptor type of content-based techniques. Overall, CBR systems have so far been used in very cooperative and "friendly" settings where it benefits content providers business, while increasing users digital experience enjoyment.

#### **From Friendly to Hostile Environments**

However, we recently witness another use of this technology. CBR is used to filter multimedia contents in order to protect the creation of the few from the piracy of the many [46, 49, 73]. CBR techniques are used to "clean the Internet", stopping the upload of material violating copyright law on User Generated Con-

tents (UGC) sharing platforms such as YouTube, or forbidding downloads from P2P networks. During the Online Content for Creativity conference, organized by the European Commission Directorate-General for Information Society and Media, panelists were asked “Is filtering DRM 2.0?”. But tables turn: MultiMedia Intelligence, a market research firm, published a study in January 2008 predicting that these content identification technologies are “poised for dramatic growth as they could fulfil the potential of digital rights technologies for monetizing digital content”. Instead of forbidding the upload of illegal copyrighted materials, UGC sites now imagine ways to monetize that contents as it is indirectly a form of advertisement for the copyright holders.

Overall, filtering is an application of CBR techniques that is quite different from its primary goal: the environment is now hostile in the sense that filtering restricts users freedom, controlling and/or forbidding distribution of content. Filtering typically requires to extract low-level signatures from multimedia contents and to query a database containing the material to be protected. Alarms suggesting copyright infringements are raised when matches are found.

The traditional approach for copyright protection is digital watermarking. This approach refers to the process of embedding information to multimedia content which may be used to verify its authenticity or the identity of its owners. However, the Achilles’ heel of digital watermarking is its robustness against geometrical attacks (cropping, stretching, change of aspect ratio, etc) and the lack of diversity in the secret keys. Because of the disadvantages of digital watermarking, various Content-Based Retrieval Systems [46, 49, 73, 69, 53, 78, 50, 10, 4, 35, 32, 64, 26, 66, 89, 85, 54] enforcing copyright protection for images and videos have been recently proposed. Evaluating their efficiency is so crucial in the real-world that a specific track (Content-Based Copy Detection track) runs since the 2008 TRECVID challenge [84].

In all new applications of CBR, systems no longer magnify cultural richness, but protect the commercial value of contents. Because there are valuable goods to protect, there may have serious hackers willing to circumvent the system. Therefore, it is legitimate to carefully investigate the security side of content-based retrieval system – this is the goal of this thesis.

## 1.2 Focus of the thesis

The work presented in this thesis focuses on Content-Based Image Retrieval Systems, CBIRS. There are some actors playing a role in the system and they can be trusted or not. However, in this thesis, we work under assumption that querying users are dishonest; they are pirates. Under this assumption, CBIRS’ use in a hostile context raises the problem of their security: is it possible for a dishonest



Figure 1.1: Right: Original Lena. Left: visual rendering after successive application of the proposed modifications. Detail of this figure is shown in section 6.4.1.

querying user (pirate) to mislead such a system so that it doesn't recognize a carefully distorted quasi-copy of a protected image? **The goal of this thesis is to examine this new problem of characterizing the security of existing content-based copy detection systems.**

The CBIRS considered in the thesis is made of several components. The early component connected to the query image. The descriptors from this image are extracted and used for retrieval via a high-dimensional search strategy. Then results of retrieval is enhanced via false positive removal, resulting in a list of ordered images. The system decides whether query is or is not quasi-copy of the protected image by considering top images in the result list. If the protected image appears at the top of the result list, it means that the query is a quasi-copy.

The thesis focuses on the visual modifications of the protected image to disturb its descriptors such that the system fails to recognize the modified image (quasi-copy). In particular, the thesis will define modifications and check their impact on the descriptors. These modifications must be sufficiently important to deteriorate the image description without too severe visual artefacts. Figure 1.1 shows the original Lena image (left) and its modified image (right) after applying the proposed modifications. The PSNR between original Lena and modified Lena images is 31.30 dB.

The description studied in the thesis is "Scale Invariant Feature Transform (SIFT)", which is state of the art among the local description techniques. Based on a deep understanding about SIFT, we propose some techniques to delude

SIFT-based CBIRS. The modifications can be performed at the detection feature step or at the descriptor computation step, or both. We also propose a technique to attack the geometric verification step used as a post-filtering for removing false positives among the results returned by the retrieval.

### 1.3 Structure of the thesis

This thesis targets the security problems of CBIRS. For that purpose, in chapter 2, we clarify the difference between two concepts: robustness and security. Then, a security threats analysis of CBIRS is proposed. In detail, we present a trust model defining who can be pirate in CBIRS, the goals of the pirate in a particular application, the level of knowledge the pirate can have on a system. With respect to this latter point, we built some considerations about the Kerckhoffs' principle and the way it applies for CBIRS. Depending on the knowledge of the pirate, we list some kinds of attacks.

In chapter 3, we describe a complete CBIRS. Especially, we focus on the main components of the system such as feature extraction, indexing and false positive removal. Because this thesis focuses on the visual modifications of images to disturb their descriptors, the state of the art about attacking content methods and detecting attacked contents methods is also mentioned in chapter 3. Chapter 4 presents an overview of the three angles of attacks, including attacking the keypoint detection, attacking the keypoint description and attacking the false positive removal, against a CBIRS based on SIFT and the experiment setup.

**Main Contributions** The main contributions of thesis are in chapters 5, 6 and 7. In chapter 5, some techniques to attack SIFT detection step are proposed. The attack can be keypoint removal or keypoint creation. Chapter 6 proposes a method to attack descriptor computation. This is done by changing the orientation of the SIFT keypoints. A learning approach using SVM is proposed to solve this problem. The attack against a complete CBIRS including a geometric verification as post-filtering is proposed in chapter 7. The proposed attack is a kind of Picture in Picture visual modification seen from a security perspective. It includes some complex steps as the selection of candidate images, the creation of visual patches, the insertion of patches, and the blur of the boundaries. The experimental results show that our approach can delude the recognition of a complete CBIRS.

Chapter 8 ends the thesis by stating the lessons drawn from this study and presenting some perspectives of future research directions.

# Chapter 2

## Security threats of CBIRS

The previous chapter explains that CBIRS is used for filtering multimedia content and their effectiveness has been evaluated by the research community. However, these evaluations only consider the ability of CBIRS to recognize content modified by basic modifications, which do not require any knowledge about the system. In other words, they evaluate the robustness of CBIRS. Intuitively, a pirate can delude a system by exploiting his knowledge about the technologies used in the system. From this inspiration, in this thesis, we consider CBIRS under a new perspective: security.

First, it is necessary to clarify upfront the concept of security since robustness is a very similar notion masking what is at stake with security. After showing the difference between these two concepts, we provide a more solid plot about the security of CBIRS.

This chapter is structured as follows. Section 2.1 defines what is security for CBIRS and how it differs from robustness. Section 2.2 presents the security threats of CBIRS. Section 2.3 concludes the chapter and introduces the work of the next chapter.

### 2.1 What is security?

Nowadays, CBIRS typically use advanced indexing techniques and powerful low-level visual descriptors, making them both efficient and effective at returning the images from a database that are similar to query images. An abundant literature describes such systems and evaluates their ability to match images despite severe distortions [46, 21, 84].

CBIRS is deemed robust if the system succeeds in recognizing contents despite modifications. Robustness assessment is what has been done for years, benchmarking CBIRS against some general modifications such as geometric transformations, cropping, stretching, color adjustment, lossy compression, ... These

modifications are the basis of standard image processing softwares such as Adobe Photoshop, Microsoft Paint, . . . and they do not focus on any specific techniques related to security. However, the security of CBIRS is different from robustness.

The security of CBIRS is challenged when pirates mount attacks after having accumulated a deep knowledge about a particular system, focusing on very specific parts where flaws have been identified.

Security is different from robustness in many points. A pirate is of course operating with the malicious purpose of deluding the system. He doesn't use classical image processing tools such as the ones provided by a photo editor software. He mounts his own attack, a process dedicated to delude a particular content identification technique. He doesn't blindly lead an attack, instead he first observes and accumulates knowledge about the details of the CBIRS techniques and then focuses attacks on very specific parts of the system where flaws have been identified. Because security attacks are designed based on some knowledge about system, they have in general bigger success rates than robustness attacks. Two following examples might help capturing the differences between robustness and security in the context of CBIRS.

**Content Concealment** The goal of the pirate is, in this case, to upload some illegal material inside a UGC platform such that it is not detected, concealed from the content filter. From white papers, from information in the press, from technical blogs, the pirate can learn what specific technique is used for extracting features. In contrast to classical image modifications, such as cropping, cam-cording, severe compressions, etc, that are pretty well absorbed by copyright protection CBIRS, the pirate can produce very specific modifications generating quite different features. Here, the extensive knowledge of the feature extraction method allows for specific attacks. For example, robust features may be extracted according to a two phase process first detecting points of interest in images and then calculating features around each point. Obviously, specifically attacking the point detector by deleting points, adding artificial points or changing their location in images may have a strong impact on the features.

**Abnormally Frequent Identifications** Thanks to photo portals, users can now buy beautiful pictures on-line. Once the picture of interest is found, they can receive a high-quality printed poster in their mail box. The typical process starts with checking a first page of thumbnails, and, with the help of keywords, relevance feedback and/or visual similarity searches, iterative refinements eventually isolate the picture to purchase. Once bought, various people (the photographer, portal keepers, . . .) get payed. Like pirates tweak HTML pages to get ranked higher in textual search engines (this is known as "black hat Search Engine Optimization

(SEO)” attack<sup>1</sup>), a pirate can tweak the visual contents of images such that they always get (artificially) ranked high in the result list of similarity searches, or, in contrast, tweak other images such that their ranks get lowered. In this case, the pirate can be someone working for the portal, and who receives dirty money by secretly favoring a particular photographer or photo agency. Note that similar effect can be produced by a pirate spying the communication channels and inserting well chosen contents on-the-fly in the image result list sent back to the user.

In the next section, we give a brief overview of various problems dealing with security. It adapts the security problems from digital watermarking and cryptography to the special case of CBIRS.

## 2.2 Security threats of CBIRS

First, it is important to clarify who are the actors playing a role in the system, then it must be clear what are the goals of the pirates, what are measurements the success or failing of an attack, what type of knowledge on the system’s internal the pirate can get, and whether the pirate can get access to some of its building blocks found off-the-shelf (or elsewhere). Even if the exact same core technology was used in all systems, the conclusions of the security analyses would differ depending on the nature of the application, on the target chosen by the pirate, on the level of details he has on the specific techniques of the system and whether he has any role inside the system itself (as in the photo portal example above). There are at least the following four important classes of assumptions that need to be clarified.

### 2.2.1 Trust model

Any security analysis relies on a trust model which lists the actors playing a role in the system and whether they can be trusted or not. There are typically four different actors in any scenario involving a CBIRS. The actors may be real persons or key software components. These actors are:

1. The image right-holder who is entitled to upload his works or their low-level features in the database.
2. The image server where a collection of features (computed from uploaded images) is indexed in a database used for building the answers of content-based queries.

---

<sup>1</sup>“black hat SEO” attack is a technique to rise the rank of web pages in an unethical manner. For example, an attacker can stuff in his website many key words or invisible texts such that his site gets a high rank when a textual search engine (eg. Google) searches for a user query.



3. The querying user who comes up with a particular image to search for.
4. The client software which processes the query image, connects to the server, send requests, receives and processes answers before displaying them.

The trust model states which actors honestly play open cards and which actors want to delude the system. There are a priory many possible trust models as any of the four above-mentioned actors, or even worse, a collusion of several of them, can be suspected of dishonesty. The most classical scenario is when the user is the pirate forging illegal copies to be concealed from the system, all other actors being trusted. An other trust model is, for instance, a right-holder modifying his contents in order to increase the recognition rate because he receives incentives whenever the query is deemed to be a copy of his works.

### 2.2.2 Goals of the pirate

The goals of the pirates might also be very different from an application to the other. In the context of CBIRS, we can easily identify two main goals:

1. Producing false negatives. The pirate manipulates images that the CBIRS fails to detect. This encompasses two situations. Either the query is manipulated, either some documents in the database are manipulated: when the server is not trusted, artificially deleting some relevant matches from the result list before sending it back to the client indeed produces some false negatives.
2. Producing false positives. The pirate manipulates multimedia material such that it will always be detected by the system (even truly innocuous contents). This also encompasses two situations: the pirate produces problematic queries triggering a denial of service, or the manipulated data belongs to the database to artificially multiply matches.

### 2.2.3 Measurements

Hacking CBIRS is quite different from hacking a cryptosystem where the disclosure of the secret key grants full decryption of ciphertexts. Here, the success or failing of an attack cannot be simply measured by a binary answer.

Attacking a protected image results in content manipulation that induces distortion. The PSNR<sup>2</sup> measures this distortion with respect to the original image in terms of Euclidean distance in logarithmic scales (the bigger the PSNR,

---

<sup>2</sup>Peak Signal to Noise Ratio

the lower the distortion). Intuitively, the stronger the distortion, the higher the success chances of an attack.

For the false negative goal, an attack on the query succeeds if the CBIRS can not detect that the query is a quasi-copy of one of the images in the database. In other words, the protected image should not be ranked at the first position in the list of ordered images returned by the retrieval process. The second condition is that the visual quality of the attacked image is acceptable. Therefore, an attack against CBIRS can be characterized by an operating curve that represents the relation of rank of protected image (let this rank be  $r$ ) and the PSNR. The key issue of CBIRS security is whether security attacks have much more powerful operating curves than edition processing, by decreasing  $r$  for a given PSNR. In the experiment, whenever a proposed attack is evaluated (chapters 5, 6 and 7), we will have a look on  $r$  and PSNR.

## 2.2.4 The Knowledge of the pirate

Obviously, the pirate's knowledge about the system have a strong impact. The more knowledge the pirate has gathered, the more malicious attack he can design. In this section, we present the Kerckhoffs' principle which is the main working assumption in cryptography. After that, we illustrate this concept by listing attacks and the required knowledge.

### 2.2.4.1 Kerckhoffs' Principle

Generally speaking, there are two broad approaches to keep things secret: the first approach maintains security of the system through obscurity. The second approach maintains security through the use of cryptographic-based techniques defining a secret key [43]. Security through obscurity relies on the bet that pirates may have very hard time to precisely know what are the algorithms used in the system, what are the implementation details, what are the parameters and what can be their values. In other words, the security through obscurity assumes the pirates are unlikely to find the security flaws due to the great complexity of the system they are attacking. It has been demonstrated that solely relying on obscurity is not reliable. Pirates look for any piece of information in publications, patents, standards or by social engineering. It is impossible to empirically assess how difficult it is to disclose information about a system. The second drawback of security through obscurity is the high cost of changing the algorithm if it gets disclosed. Designers need to re-implement another obscure algorithm, likely way different, with heavy testing phases and a high burden for deploying the algorithm on sites. These problems have been reported since a long time and Kerckhoffs came up with several design principles still applicable today [43]. His

best known principle says that the system must remain secure even if everything about the system, except a secret key, is public knowledge. A secret key is a piece of information that determines the output of a particular algorithm: the same algorithm does not produce the same output if it is fed with the same data but with different keys. This secret key solely guarantees the security of the system. As it is much easier to protect a small piece of information (the secret key) than a complete system by obscurity, secure systems using secret keys are more reliable. A secret key is one or more very large random numbers. Finding its value by an exhaustive search is almost impossible as the key space is very large. If the secret key is discovered, then creating a new key is easy and fast. Real-world secure systems typically include secret keys and/or elements of obscurity.

If the algorithm is known, two neighboring concepts were defined by the digital watermarking community:

- The worst case attack is the most efficient attack when the pirate knows the algorithm but not the secret key [87]. Efficiency is typically measured by the loss of quality versus the increase in probability of watermarking decoding errors.
- The security hack is different in the sense that the pirate first takes time to analyze observations in order to estimate the missing information such as the secret key, and then leads an attack based on this stolen knowledge [9].

#### 2.2.4.2 The Kerckhoffs principle and CBIRS

We discuss here how the Kerckhoffs principle applies to CBIRS. From a bird's eye point of view, CBIRS can be classified in two broad categories depending on whether they use secret-based techniques or not.

**Secret-based CBIRS** Very few CBIRS use a secret key. Among this small group, some use the secret key to generate a private selection of parts of the contents [45] or of parts of its features. Another approach defines a secret transform which extracts some private features [59]. Both [40, 80] define a secret quantifier used to quantize the extracted features. In this context, the worst case attack is the best process in terms of probability of successfully changing the robust hash, and quality of the manipulated content. On the other hand, a security hack is possible by observing pairs of an image and its robust hash. This attack framework can be entitled KIA for Known Image Attack, a terminology coming from the cryptanalysis: in the Known Plaintext Attack, the adversary observes pairs of plain and cipher texts. The assessment evaluates the security level as the number of pairs needed to disclose the secret key with a given accuracy. This is also

known as the unicity distance [60]. Then, a second step is to mount the attack itself which takes full advantage of the disclosed key to forge pirated images.

**Non Secret-based CBIRS** Most CBIRS do not include any secret key. Therefore, there is no security hacks against these non-secret techniques, but just worst case attacks. There are certainly plenty of parameters which are unknown to the pirate. However, the pirate knows that the designer of the algorithm fine-tuned the values of these parameters to provide the best performance. Therefore, those parameters are usually not random, and it is possible to, at least, estimate windows where their true values lie by using common sense, logic and careful thinking.

**Knowing the Contents of the Database** A critical point specific to CBIRS security is whether or not the pirate partially knows the contents of the database. Rising false alarms becomes much easier if the pirate has this knowledge. It can simply be acquired thanks to the reputation of the image server making public the names of solid clients it is working with. Or, specific database probing protocols can be cooked by the pirate to get a glimpse of the contents. This is related to the concept of oracles, detailed next.

### 2.2.4.3 Access to Oracles

In cryptanalysis and digital watermarking security assessments, there is a class of attack named oracle attacks. The pirate has an unlimited access to a piece of software that is part of the system under scrutiny - this piece is said to be an oracle. It may even be a sealed black box process. What matters is the access to the output and the total freedom to run that software on any arbitrary, yet well chosen, input. With CBIRS, one oracle could be the software calculating local descriptors, or the filtering system itself giving a binary decision. There are two families of oracle attacks: the Chosen Image Attack (CIA) [11] and the Closest Point Attack (CPA) [65].

**Chosen Image Attack** In this case, the pirate aims at disclosing a secret by challenging the oracle. The difference with the KIA (see 2.2.4.2) is that the pirate is allowed to choose the images sequentially. For secret-based CBIRS, the pirate iteratively creates an image, observes its robust hash and gain knowledge about the secret key with the minimum number of calls to the oracle. For non secret-based CBIRS, the pirate can tests whether a given image feature, such as the chrominance channels, play any role in the search.

**Closest Point Attack** Here, the pirate is not interested in disclosing any secret. He has a content which is filtered out by the system, and his goal is to forge the least degraded copy of this content which will not be blocked by the filter. In the content space, the content is a point belonging to the acceptance region of the filter. A very degraded copy is another point lying outside this region. The question is about the fastest iterative process increasing the quality of the pirated copy. For instance, as a toy example, within few oracle trials and the help of a dichotomy search, the pirate easily finds the content which lies at the boundary of the acceptance region. Interesting results are known in the watermarking community: there exist algorithms which do not need any assumption about the shape of the acceptance region, or in other words, about the technique used by the filter [65]. The increase in quality is very fast at the beginning (ie. in the first hundreds of trials) and then it is going very slowly requiring millions of trials to get any substantial quality gain [23].

## 2.3 Summary

From the intuition that a pirate can delude a system if he has a deep knowledge about its internal technologies, in this chapter, we have considered CBIRS from a security point of view. First, we provided the definition about the security of CBIRS. After that, we described the key ingredients of a security threats analysis such as the trust model, the goals of pirate, the measurements for an attack, the working assumptions (e.g. Kerckhoffs principle) and also the kind of attacks a pirate can design. A significant part of the work presented in this chapter has been published in [18].

In the next chapter, we present a complete CBIRS as well as the state of the art methods to attack content and detect attacked content.

# Chapter 3

## State of the art

In the previous chapter, we presented the security threats of CBIRS. This chapter now underlines that the security of a CBIRS strongly depends on the knowledge of the pirate about system. It consists of the state of the art of the main components of the system where attacks can appear. Although several actors of CBIRS might be dishonest (section 2.2.1), this thesis assumes that the pirate is the querying user. Under this assumption, the pirate leads his attack on the query image. This chapter also gives the state of the art of the attacks at the query side.

This chapter is structured as follows. Section 3.1 presents the overview of a CBIRS. Section 3.2 gives the state of the art about attacking content methods and also detecting attacked contents methods. Section 3.3 concludes the chapter.

### 3.1 Overview of CBIRS

Many CBIRS are proposed in the literature. The general schema of a CBIRS is shown in figure 3.1.

There are three main components in a CBIRS, which are “feature extraction”, “indexing” and “false positive removal”. The action of a CBIRS system can be divided into two stages. The first stage computes features for all raw images in the database. These features are indexed to create a database. This stage is done off-line. This stage appears in blue dash lines in figure 3.1. At the second stage, or on-line stage, the user queries an image to the system. First, the features of the image are extracted. These features are used to query the database of indexed features computed at first stage resulting in a short list of candidate images. The search process at second stage ranks the database images without exploiting geometric information. The accuracy may be improved by adding a “false positive removal” stage<sup>1</sup> [72]. The short list of candidate images is subjected to the “false

---

<sup>1</sup>“False positive removal” will filters matching descriptors that are not consistent in terms

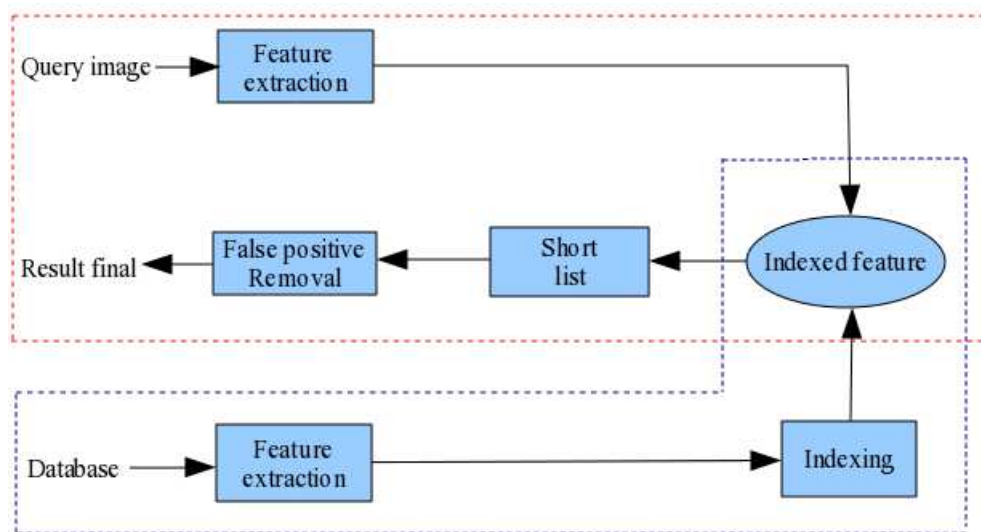


Figure 3.1: General overview of a CBIRS

positive removal” step in order to rerank images. The reranked list is shown to the user. The higher the rank, the more similar to the query image the images are. This second stage is marked in red dash lines in figure 3.1.

The next sections detail each component of the system by its state of the art.

### 3.1.1 Feature extraction

This component extracts a low-level description of the images. The descriptors are typically high-dimensional vectors, which can be compared with a specific metric (often  $L_2$ ). An image is described by a global feature or a set of local features.

**Global feature** Quantization is used to build a histogram for the whole image in the spatial domain (e.g. histogram of color) or in the transformed domain (e.g. histogram of textures, edges, ...). Typical color systems and transforms used in this approach are

- Color: HSV, Lab,... [14, 12, 79, 77]
- Transform: Gabor / Fourier / Wavelet transform applied to image, a histogram is built in the transform domain [67, 44]

The global description is efficient with respect to computation time and storage footprint, but not very accurate. For example, two images can have similar  


---

of angle, scale and positions.

histograms of colors or edges but very different visual content. The global feature in CBIRS tends to generate many false positives. Furthermore, the global descriptors are not so robust against common modifications like affine transformations, cropping . . .

**Local feature** Recently, many CBIRS use a local description because it is more robust against affine transformations, cropping, illumination adjustment . . . The local feature extraction is in two steps. At the first step, some local interest regions in the image are localized by the region detector. At the second step, the descriptors of these local regions are computed. Some state of the art feature detectors are Harris corner detector [30], FAST corner detector [74], DOG [56], LOG, Harris-Laplace, Harris-Affine, Hessian-Laplace, Hessian-Affine [61, 62]. Some local regions description are SIFT [56], PCA-SIFT [41], SURF [5], DAISY [82, 83], or GLOH [63]. SURF, GLOH and DAISY are extensions of SIFT.

In [63], the authors make an evaluation of common local descriptors like SIFT, GLOH, shape context [6], and spin image [47]. Given an image, they applied some modifications such as scale changes, image rotation, image blur, JPEG compression, or illumination changes to the image. After that, they perform a matching between original and modified images. The results show that SIFT and GLOH have a bigger number of correct matches than the other competitors.

### 3.1.2 Indexing

In a CBIRS using global descriptor, only one vector describes an image, however, when the size of the database is big (e.g. 100M images), finding the nearest neighbors of a query vector by the exhaustive search is very time consuming. This problem becomes more significant for CBIRS based on local descriptors where each image can have thousands of local high dimensional descriptors (e.g. 128 dimensions for SIFT). This results in billions of vectors in the database. For each local descriptor vector of the query image, the exhaustive search looking for the nearest neighbor vectors is impractical. An indexing scheme is designed to yield a faster search. Indexing techniques are divided into two categories. The first category performs an exact search method such as KDTree [25] or R-Trees [28]. The second category performs an approximate search such as  $K$ -means [58] and its variations (Approximate  $K$ -means (AKM), Hierarchical  $K$ -means (HKM) [72]), Bag Of Feature [76] and its improved versions [37], Locality-Sensitive Hashing [15, 27], or NVTTree [48].



### 3.1.3 False positive removal

Many CBIRS use various techniques to improve the quality of search results. A common approach is to check the geometric consistency between the query image and the images of the short list. The usual methods are:

- Generalised Hough Transform(GHT) [3, 56],
- RANSAC [24, 52, 72].

The general idea of these methods is to detect an affine transforms between two images (the query image and one image in the short list). The number of local features complying with the detected transformation is the final score of the image. This produces a re-ranking in the short list.

## 3.2 Security of contents

As said in section 1.2, we consider in this thesis the security of CBIRS under the assumption that the pirate is the querying user who comes up with a particular image to be searched for. With this assumption, the pirate modifies / attacks the query image to achieve his goal. In this section, we present a state of the art of such attacks and also some counter-attacks. These attacks are divided into two categories. The first category gathers the attacks designed to break the robustness of CBIRS. The other category is composed by the attacks breaking the security of CBIRS.

### 3.2.1 Robustness attacks

The pirate creates quasi-copies by applying some standard modifications such as rotation, shearing, resampling, resizing, cropping, filtering, loss compression ... and their combination. These modifications can be found in many image processing softwares such as Adobe Photoshop, Microsoft Paint ...

Another tool to create quasi-copy is Stirmark [71]. Originally, StirMark is a generic tool for benchmarking the robustness of still image watermarking algorithms. Basically, StirMark simulates a resampling process, i.e. it introduces the same kind of distortions than one would expect if we print an image on a high-quality printer and then scan the document with a high-quality scanner. The algorithm applies a minor geometric distortion, i.e. the image is slightly and locally stretched, sheared, shifted, and/or rotated by an unnoticeable random amount and then resampled by a bi-linear interpolation. Figure 3.2 illustrates the StirMark attack with the Lena image.



Figure 3.2: StirMark attack on Lena image

Another kind of attack, which does not apply geometric transformation is image tampering. In this attack, some parts in the quasi-copy are copied from other images or other parts of the protected image are cloned or an object/person is covered, or a combination of these operations. An illustration of tampering attack is given in figure 3.3.



Figure 3.3: An example of tampering attack that appeared in press in July, 2008. The tampered image on the right shows four Iranian missiles but only three of them are real. Two different sections (marked in red and purple) are replicates of other image parts. (Figure is from [2])

All attacks mentioned above are basic and may be used against various systems. They are not specific of any system. They challenge the robustness of

CBIRS and do not give any information about security angle.

### 3.2.2 Security attacks

As presented above, there are many attacks challenging the robustness of CBIRS. In contrast, very few works propose attacks to evaluate the security of the system. In [33], Hsu et al. proposed an attack to an image authentication system hashing SIFT features. Their attack tries to remove SIFT keypoints inside the image. In brief, given the location of an interest point in the image, the idea is to locally add a patch  $\epsilon$  to this interest region  $I_m = I + \epsilon$ , such that there is no longer a unique local extremum at this point and therefore the keypoint has been successfully removed. They claim that their attack succeeds in breaking the targeted authentication scheme.

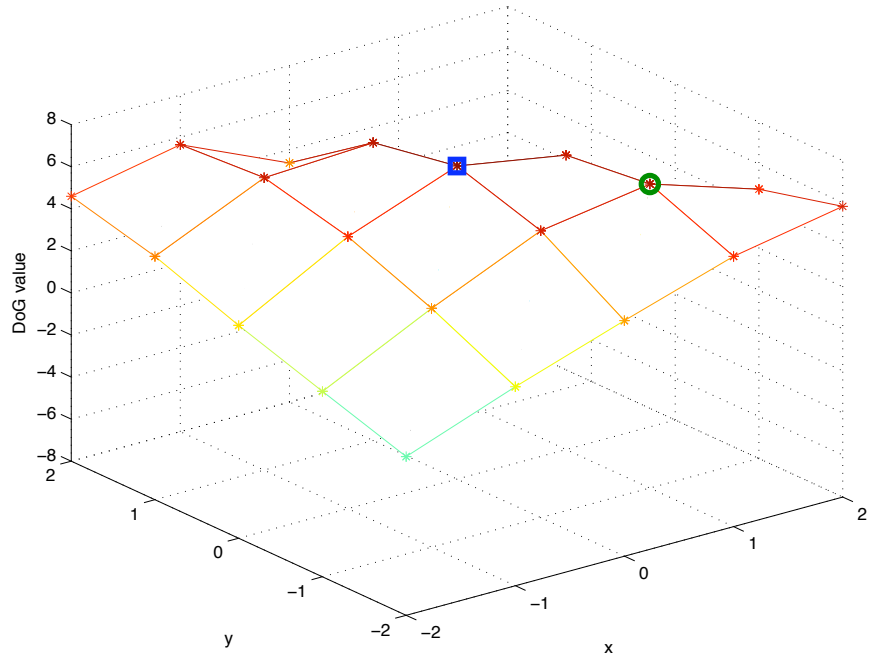
We study their attack in our copy detection scenario using a CBIRS instead of the hashing. After carefully implementing their anti-SIFT attacks and running experiments, we have the conclusion that their attack can not break CBIRS [20]. The attacked images are still matching their original version. After some investigations, the drawback of their method is that removing keypoints triggers the creation of new and undesired ones that match to the original keypoint. One example is presented below.

We apply the patch  $\epsilon$  using their method at a particular position  $(x_1, y_1)$  of the Lena image. It was originally detected as a keypoint because of its local maximum at scale  $\sigma = 1.37$ . Fig. 3.4(a) shows the Difference of Gaussian (DoG) local extremum originally detected, identified on the figure by a blue square, and the second extrema in its neighborhood (green circle). After attack  $(x_1, y_1)$  is no longer an extremum as the original first and second local extrema values are now equal (Fig. 3.4(b)). However, a side effect of their method is the creation of a new local extremum in the neighborhood of  $(x_1, y_1)$  as indicated by a red triangle on Fig. 3.4(b).

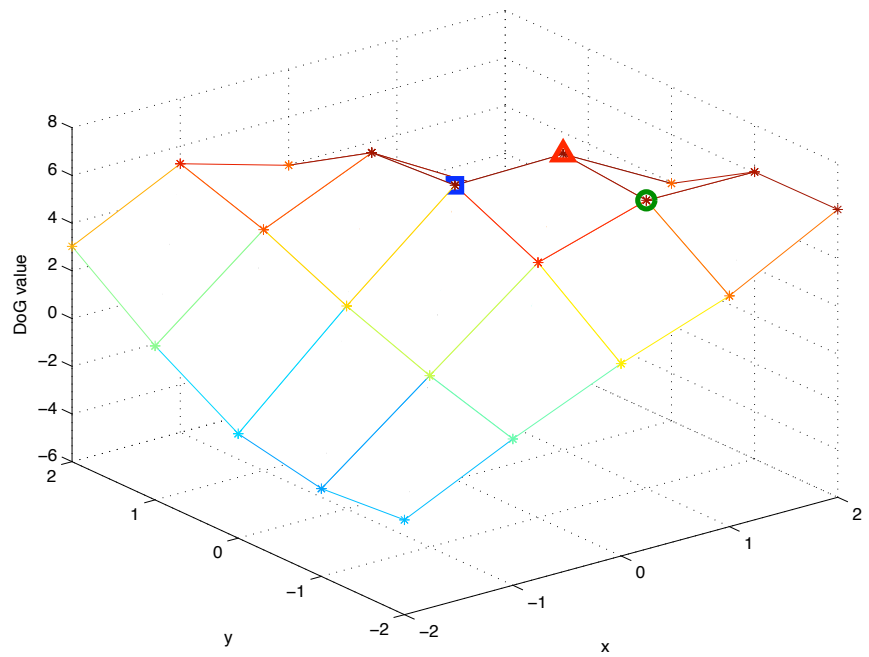
In our experiment, we found that the local description of these new keypoints is too similar to the description of the removed keypoints. Hence, these descriptors are easy to match. The protected image is still identified when quasi-copy image is queried. Hence, this attack is not at all a threat for CBIRS based on SIFT.

### 3.2.3 Competitive technologies

Digital watermarking is a competitive technology for monitoring content consumption and detecting fraudulent use of copyrighted material. It embeds metadata in multimedia content in a robust and imperceptible way [13]. The metadata are of different nature depending on the application. Examples are name of the copyright right holders, identity of the consumer / buyer or the provider, iden-



(a)



(b)

Figure 3.4: Effect of Hsu's attack on the 5x5 neighborhood of a particular key-point: (a) original DoG values, (b) DoG values after attack.

tifier of the content, licensing term of usage. The embedder and the decoder share a secret key, which prevents dishonest user to access to the watermarking channel.

The main advantages of digital watermarking are the following:

- There is no need of a database. The hidden message should be self sufficient to screen the contents. This allows to place watermark decoders in many different point in the content distribution chain. For instance, the Blu-ray disc players look for watermarks in the audio channels to spot illegal camcording of theater movies.
- Two versions of a Work can be watermarked with two different messages. This allows to identify, for instance, which distribution chain a piece of content went through (IPTV, DVD, theater etc.)

Its main drawbacks are as follows:

- The content is modified by the watermark embedder. It resorts to complex human perception models to shape the embedding distortion under the just noticeable difference to remain invisible / inaudible to humans.
- Placing watermark decoders here and there rises the issue of keeping the secret key secret.

As far as security is concerned, this concept and its differentiation with robustness has been studied by the watermarking community during almost a decade. The main conclusion is that, for a given embedding distortion, there is a trade-off between security and robustness. As a matter of fact, it is quite difficult to design a watermarking technique robust to both geometric and valuometric (coarse compression, filtering, noise addition etc.) attacks. Some watermarking techniques by the way use local features like SIFT [57]. This broadens the scope of application of the results of this thesis. Add the requirement about security on top and one finds out where the watermarking community stands: there is no clear consensus whether a secure and fully robust watermarking technique might exist one day.

A way to deal with the lack of security of robust watermarking is to periodically change the secret key. This stems into the issue of how the watermark decoder can synchronize its secret key with the one used at the embedding stage. This is where digital watermarking and CBIRS can cooperate: a CBIRS first recognizes the content under scrutiny and then gives the associated secret key to the watermark decoder. The security of this hybrid system then relies on the CBIRS component ; this is all grist for our mill.

There are also some typical attacks in watermarking which may not have any reality in the field of CBIR.

- Oracle attacks: if a consumer electronics device has a watermarking decoder, a dishonest user can challenge it as many times as he wishes to deduce some knowledge about the secret key. Section 2.2.4.3 discusses the topology of oracles attacks
- Collusion attacks: if two versions of the same content are watermarked with different metadata, what is decoded when one mixes them (say by a sample-wise averaging)? The application “active fingerprinting” or “traitor tracing” where the metadata is the identifier of the consumer is robust against such type of attacks. However, the cost to pay is a dramatic decrease of the embedding rate.
- Known Original Attack: it is clear that if a pirate has the original version of a piece of content, there is no need to attack its watermarked version. But this is even worse: the difference signal may reveal information on the watermarking technique and the secret key.

### 3.2.3.1 Some related applications

Some methods are developed to detect tampered image attack. In [31], the authors proposed an efficient method to detect tampering and to locate tampered regions in the image under scrutiny. This forensics analysis is based on double quantization effect. This effect appears when JPEG compression is applied to an image two times. However, this method only works when original images is a JPEG image.

In [1, 2], SIFT is used for image tampering detection in which the image is modified by a copy-move attack (see Fig. 3.3). The SIFT feature are first extracted from the image. Each descriptor is matched to its nearest neighbor and the matching points are clustered based on their spatial location. A transformation is estimated between clusters to determine if one region is the copy of the other. This is another context where our security analysis finds a key role.

In image hashing, a binary signature, so called hash or perceptual hash, is computed from the visual content of an image and a secret key. This signature is stored as a metadata in the header of the file. If the image undergoes an innocuous transformation like a light JPEG compression, the perceptual hash should not change. On contrary, if the image is tampered, his hash changes and no longer equals the signature in the header. The pirate doesn't know the secret key and this prevents him to update the signature. His goal is thus to create a forgery while keeping the same perceptual hash, or whose perceptual hash is the same as another authenticated image. The features hashed in the signature can be global [80] or local feature [34, 42, 75]. The security is gauged by the entropy of the hashes, which is somehow questionable. A big entropy confirms that the

probability of finding two images with the same hash is very weak. But it does not say how sensible is the hash with respect to tampering. It also does not prove that the pirate observing signed images cannot disclose the secret key.

### **3.3 Summary**

This chapter gives an overview of CBIRS. We present the state of the art of the main components of the system. After that, we present some attacks proposed in literature for evaluating the robustness and the security of the system. We see that most of attacks challenge the robustness. They are basic modifications and do not require any knowledge about the system. On the other hand, few attacks are security oriented. We did an evaluation of the anti-SIFT attack proposed in [33]. Our study shows that this attack is not at all a threat for CBIRS based on SIFT. A detailed version of our evaluation presented in section 3.2.2 has been published in [20].

We also present state of the art methods for detecting copies of a content. Beside CBIRS, there are other techniques relying on watermarking and hashing. However, as said in section 3.2.3, watermarking approaches weakly resist to geometric attacks while hashing approaches based on local feature are not robust if pirates remove or insert local features in the image. This is one of the attacks that we present in the next chapter.

# Chapter 4

## Overview of solutions

In chapter 3, we have discussed the structure of CBIRS and the state of the art of its main components: feature extraction, indexing, and false positive removal. In this chapter, first, we detail the components of our own system. The assumptions about the security as well as the criteria to evaluate an attack are also presented. Then, we propose three key ideas for designing attacks.

This chapter is structured as follows. Section 4.1 details the experimental setup used in the remaining of the thesis. Section 4.2 briefly describes three ideas to delude the recognition of the system. Section 4.3 concludes the chapter and introduces the work of the next chapter.

### 4.1 The experiment setup

#### 4.1.1 Overview of our system and the evaluation criteria

A large database of images to be protected has been created, offline. Given a protected image, we apply a specific modification resulting in an attacked image. The attacked image is used as query image. The query image is submitted and checked against the database. The system decides whether this query is a quasi-copy of the protected original image, which belongs to the database.

We consider that the system succeeds in recognizing the attacked content when the original version is on the top of the result list returned by the retrieval process. In contrast, the system fails (it means that the attack succeeds) when the original version of the attacked image is not on the top of result list. It is “hidden” behind other images. In the experiments of the next chapters, we look at the score of the original image and the first best non-original images. This measures the strength of the attack. An “strong” attack yields a big gap between the score of best non-matching image and the one of the original version.

In [81], the authors show that the PSNR is a good measure of the variation



of the image quality when the visual content are fixed across the test conditions. In our scenario, the pirates aims at forging an attacked image while keeping the same visual content. Hence, we compute the PSNR to measure the quality of the attacked images. The PSNR between an attacked image  $K$  and its original version  $I$  is computed as

$$\text{PSNR} = 10 \cdot \log_{10} \left( \frac{255^2}{\text{MSE}} \right) \quad (4.1)$$

where MSE is the mean squared error between two images, computed as

$$\text{MSE} = \frac{1}{m n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2 \quad (4.2)$$

Intuitively, the more distortion the attack introduces, the higher its chance of success.

### 4.1.2 Description

This thesis focuses on the SIFT descriptor proposed by Lowe [56]. SIFT is a state of the art descriptor for CBIRS [63].

Describing a gray image using the SIFT method includes three steps: (i) the keypoint detection, (ii) the main orientation computation, (iii) the descriptor computation.

In the first step, a pyramid of Gaussian images is built. The Gaussian images are grouped by octaves. Each octave is a group of 6 Gaussian images having the same size. To build the first octave (called octave  $-1$ ), the original image is doubled and smoothed by a Gaussian kernel of standard deviation  $\sigma_1$ . In [56],  $\sigma_1$  is set to  $1.6^1$ . This smoothed image is the first image of the octave  $-1$ . The next five Gaussian images of this octave are formed by smoothing the first image with Gaussian kernels with  $\sigma_2 = 1.6k$ ,  $\sigma_3 = 1.6k^2$ ,  $\sigma_4 = 1.6k^3$ ,  $\sigma_5 = 1.6k^4$ ,  $\sigma_6 = 1.6k^5$ , respectively, where  $k$  equals  $2^{\frac{1}{3}}$ . The consecutive smoothing of an image by two Gaussian kernels with standard deviation  $\sigma_a$  and  $\sigma_b$  is equivalent to smoothing with a Gaussian kernel with  $\sigma = \sqrt{\sigma_a^2 + \sigma_b^2}$ . In the implementation, the  $i^{\text{th}}$  Gaussian image is made by smoothing the previous image by a Gaussian kernel whose standard deviation equals  $\hat{\sigma}_i = \sqrt{\sigma_i^2 - \sigma_{i-1}^2}$  where  $i = \{2, \dots, 6\}$ .

To build the next octave, its first Gaussian image is formed by the down-sampling by a factor 2 of the fourth Gaussian image of the previous octave. The

---

<sup>1</sup>In [56], the author assumes that the original image has a blur of at least  $\sigma = 0.5$  (the minimum needed to prevent significant aliasing) and that therefore the doubled image has  $\sigma = 1.0$  relative to its new scale. Hence, in the implementation, the real  $\sigma$  used for smoothing of doubled image is  $\sqrt{1.6^2 - 1^2}$

$i^{\text{th}}$  Gaussian images of the octave is built by smoothing the previous adjacent Gaussian image with  $\sigma = \hat{\sigma}_i$  as for the octave  $-1$ .

Once the pyramid of Gaussian images is created, the pyramid of Difference-of-Gaussian functions  $D(x, y, \sigma)$  is built. In detail,  $D(x, y, \sigma)$  is calculated by the subtraction of two adjacent Gaussian images:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) \otimes I(x, y) \\ &= \Delta G_{\sigma}(x, y) \otimes I(x, y), \end{aligned} \quad (4.3)$$

where  $\otimes$  is the 2D convolution operator, and  $G(x, y, \sigma)$  is the Gaussian kernel of standard deviation  $\sigma$ . Because there are 6 Gaussian images per octave, there are 5 DoG functions per octave. Figure 4.1 illustrates the pyramids of Gaussian images and DoG functions.

The detection of keypoints relies on the local extrema of the Difference-of-Gaussian function  $D(x, y, \sigma)$ . To detect these local maxima and minima, each sample point of the DoG functions is compared to its eight neighbors in the current scale and its nine neighbors at scales above and below. This extrema detection is processed only on the three inner DoG images of each octave. Figure 4.2 illustrates this detection. Once a keypoint candidate has been found by comparing a coefficient to its neighbors, a substep performs a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. This information allows rejection of candidates that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge.

Overall, a keypoint is detected at location and scale  $\mathbf{x} = (x, y, \sigma)^T$  if the following three conditions hold:

- $D(\mathbf{x})$  is a local extrema over a neighborhood of  $\mathbf{x}$ ,
- a sustainable contrast is present, i.e.,  $|D(\mathbf{x})| > C$  where  $C$  is a threshold hard-coded in the algorithm,
- the keypoint is not located on an edge, which can be detected by

$$\frac{\text{tr}(\mathbf{H})^2}{\det(\mathbf{H})} < \frac{(r+1)^2}{r}$$

where  $\mathbf{H}$  the 2x2 Hessian matrix, computed at the location and scale of the keypoint.  $r$  is the ratio between the largest magnitude eigenvalue and the smaller one of  $\mathbf{H}$ .

The second step finds one or more main orientations for each keypoint based on the directions of the gradient. An orientation histogram is formed from the gradient orientations within a region of size  $[-4\sigma, 4\sigma]^2$  around the keypoint, called

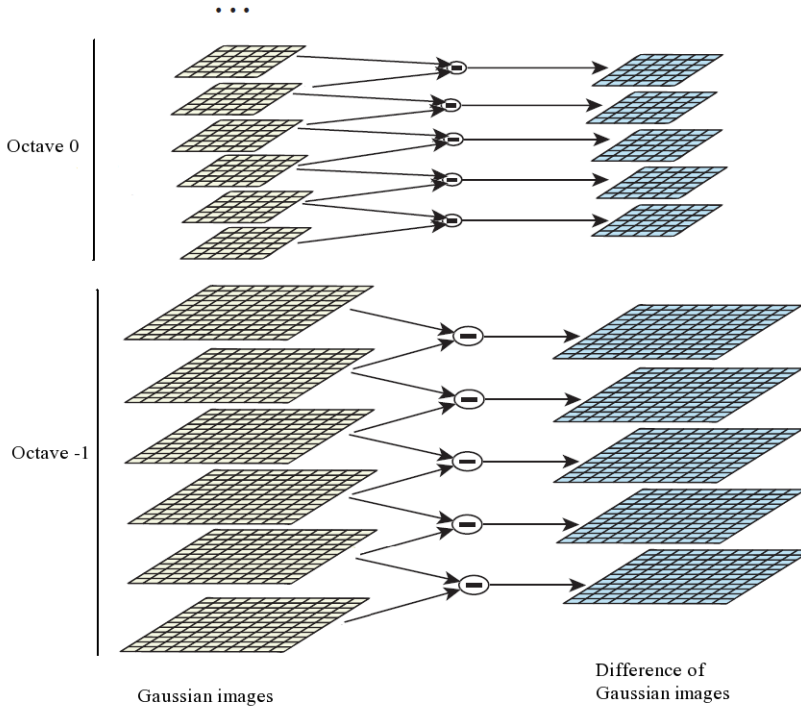


Figure 4.1: For each octave of the scale space, the initial image is iteratively convolved with Gaussian kernels to produce the set of scale space images shown on the left. For the next octave, the Gaussian image is down-sampled by a factor of 2, and the process is repeated. Two adjacent Gaussian images are subtracted to produce the difference-of-Gaussian functions on the right.

support region of the keypoint. The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window whose deviation is  $1.5\sigma$ . After collecting the data in the bins, the histogram is smoothed by a moving average filter. The keypoint orientation is obtained as the maximum of this histogram. In addition to the global maximum, a local maximum with a value above 0.8 of the maximum is retained as well. Thus for each location and scale, multiple orientations might be generated. After orientation computation, each keypoint  $\mathbf{x}$  is presented by four value:  $\mathbf{x} = (x, y, \sigma, \theta)$ , where  $\theta$  is keypoint orientation.

The final step computes a descriptor for each keypoint over its support region. The descriptor layout is a  $4 \times 4$  grid. First the image gradient magnitudes and

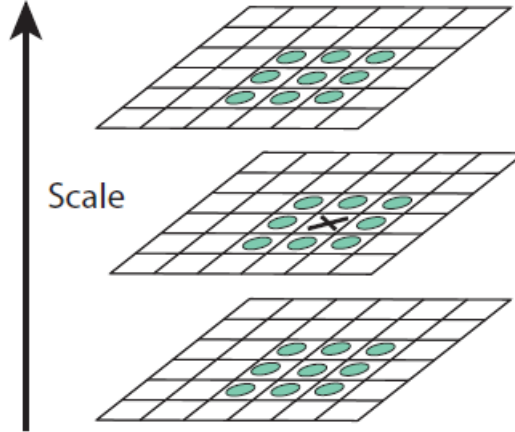


Figure 4.2: The local maxima and minima of the DoG images are detected by comparing a coefficient (marked with X) to its 26 neighbors in a  $3 \times 3$  region at the current and adjacent scales (marked with circles). Figure is from [56]

orientations are sampled around the keypoint location. In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint main orientation.

The gradient orientation then is quantized into 8 bins. It means that each bin covers a 45 degree range of gradient orientation. The gradient magnitude of each sample point is further weighted by a Gaussian function whose standard deviation equals one half of the width of the descriptor window, and these values are accumulated into the bins. At the end, each local patch is described by a vector having  $4 \times 4 \times 8 = 128$  dimensions.

We compute the local SIFT descriptors using the open-source SIFT-VLFeat code by Vedaldi [86]. This open-source software is widely used by the research community (it has roughly 200 citations according to Google Scholar at the time of writing). By using SIFT-VLFeat software, data type of descriptor is unsigned 8-bit integer. It means that value of each element in the descriptor is in interval  $[0, 255]$ . We did several experiments to get descriptors that are as close as possible to the original SIFT description based on Lowe’s binary [55]<sup>2</sup>, both in terms of number of descriptors and of spatial location in images. In our case, the best configuration is when  $C = 4$  and  $r = 12$ . The number of octaves ( $O$ ) depends on size of images. It is determined by

$$O = \lfloor (\log_2(\min(M; N))) \rfloor - 2,$$

<sup>2</sup>SIFT is protected under a US patent. David Lowe only provided binary version to compute SIFT descriptor.

where  $M, N$  is the height and the width of the original image. It means that the smallest image on the top of the pyramid is approximately 4 pixels long on its shortest edge.

### 4.1.3 Indexing

In [48], Lejsek et al. proposed a new indexing method for approximate nearest neighbor search in very large high-dimensional collections, called NVTree (Nearest Vector Tree). This tree is similar to a hierarchical  $K$ -means [72]. However, instead of using  $K$ -means for the partition into clusters at each level of the tree, they used random projection. All SIFT descriptors of the image collection are projected to a single projection line. The projected values are partitioned into disjunct sub-partitions. An overlap partition is created for each pair of adjacent partitions to avoid boundary problem. The projection and partition are repeated to create the sub-levels of NVTree. The process stops when the number of descriptors in a sub-partition is below a threshold designed to be disk  $I/O$  friendly. This sub-partition is called a leaf node. At this time, a projection line is used to order the descriptor identifiers in a leaf node. After that, the ordered identifiers are written on disk.

At the query stage, the query descriptor first visits the inner nodes of NVTree. At each level, the query descriptor is projected onto the projection line associated with a node. The projection value is used to decide the sub-partition of the next level whose center point is the closest to projection value of the query descriptor. This process is repeated and stopped when a leaf node is reached. The ordered values of this leaf node are loaded into memory. The query descriptor is projected onto the line associated with the leaf node. The two closest descriptor identifiers on either side of the projected query descriptor are returned as the nearest neighbors. Then the two nearest neighbors of these two above descriptors are returned and so on. In our experiment, the 1,000 nearest neighbors for each given query descriptors are considered for the voting. Because the search results are based on projection onto a single line at leaf node, this may result in false positives. This can be avoided by applying some ranking method to combine results from several NVTrees. In our experiments, three NVTrees are used.

For each query descriptor, two voting mechanisms of NVTree are considered in our experiments. One is multiple voting: each image in the database receives a number of votes equal to the number of its descriptors belonging to the 1,000 nearest neighbors of the query descriptor. The other is single voting: each image in the database receives one vote if number of its descriptors belonging to the 1,000 nearest neighbors of query descriptor is bigger than zero.

The total number of votes after processing all query descriptors is used to rank the images. When several NVTrees are used, each image in the database

will receive votes from each NVTree. The final score is the sum of these votes.

#### 4.1.4 The geometric verification

The geometric verification is a filtering stage performed on the candidate images returned by the search engine. It checks the geometric consistency of matching keypoints to eliminate false positives. It relies on the affine transformation estimation between the query and a candidate image. Once the affine transformation is estimated, the candidate images are reranked according to the number of their keypoints that verify this affine transformation.

To estimate the affine transformation, keypoints of both images are first matched, providing a list of matching points. For a given keypoint  $k_q$  of the query, its nearest neighbor in the candidate image is defined as the keypoint  $k_1$  with minimum Euclidean distance for their SIFT descriptors denoted by  $d(k_q, k_1)$ . One possible matching criterion proposed by Lowe [56] compares the distance of the closest neighbor  $k_1$  to that of the second-closest neighbor  $k_2$ . Two keypoints  $k_q$  and  $k_1$  match if  $d(k_q, k_1) \leq 0.8d(k_q, k_2)$ . This measure performs well because the correct pairing needs to have the closest neighbor significantly closer than the closest incorrect match to achieve a reliable matching [56]. We will use this matching criterion in chapter 7 and it will be referred as “Lowe criterion” in the following. However, a problem appears if the candidate image have very similar local descriptors: for example if there are two identical logos in the image, the query descriptors will not match any of them due to “Lowe criterion”.

The estimation of affine transformation is performed by a Generalized Hough Transform (GHT) on the pairs of matching points [56]. After that, the least square estimation [56] is computed on matched pairs with the best transformation to remove outliers. The number of inlier points satisfying the estimated model is used as the score to rank the images. We use the open source software developed by A. Jepson [39] for the GHT and least square estimation. Because the geometric verification is time consuming, it is usually performed on a short list of top results returned by the search.

#### 4.1.5 Dataset and queries

Our image collection is composed of 100,000 random pictures downloaded from Flickr that are very diverse in contents. All images have been resized to 512 pixels on their longer edge. Because SIFT works on gray images, the images are converted to gray scale if they are color images. Intensity value for each pixel in image is in interval  $[0,255]$ . This collection yields 103,454,566 SIFT-VLFeat descriptors indexed by the NV-Tree.

We then randomly picked 1,000 of these images as protected images. All

attacks in the next chapters are performed on these 1,000 images. The attacked images are used as queries. For each query, we keep track of the scores of the 100 best matching images. The final result is averaged on the 1,000 queries. Additionally, the Lena image is used to illustrate the visual impact of the different attacks.

### 4.1.6 Security assumption

In chapter 3, we listed the actors in the systems potentially dishonest: the image right-holder, the image server, the query user or the client software. In this thesis, we analyze the security threat on the system under the assumption that only the query user is dishonest.

## 4.2 Three angles of attacks

The structure of a CBIRS of chapter 3 suggests 3 ideas to delude recognition based on local descriptor. This section gives a short introduction, and the next chapters provide their in-depth study.

### 4.2.1 Attacking the keypoint detection

When a pirate attacks this step of the system, its goal is to produce specific modifications of the images entirely driven by the deep understanding of the properties of the detection scheme, such that matches with descriptors of the database will subsequently fail. To attack the keypoint detection, a pirate may want to artificially add new keypoints or delete keypoints by applying well-chosen local modifications. Having new keypoints by patching the images in turn creates new descriptors, increasing the score of irrelevant images. Deleting detected keypoints decreases the score of original image. Hence it lowers the probability that the original version of the pirated image appears at the top of the short list. Chapter 5 will present in details attacks at this step of system.

### 4.2.2 Attacking the keypoint description

Based on a deep knowledge about the descriptor computation, a pirate may modify the support region of the keypoint such that its descriptor is changed. Hence, a modified descriptor will be unlikely to match with its original version in database, and the score of the original image will decrease. Similar to the above attack, this lowers the probability that the original version of the pirated image appears at the top of the short list. The attack at keypoint description step will be the main content of chapter 6.

### 4.2.3 Attacking the false positive removal

When pirates face to geometric verification problem of CBIRS, a way to attack geometrical consistency is to insert into the pirated image distractors extracted from existing images of the database (let this image be  $I$ ), such that the number of matches between pirated image and  $I$  is greater than the number of matches retained by the geometric verification between the pirated image and its original version. Such that,  $I$  will be ranked higher than the original image. Chapter 7 will focus on attacks at this step of the system.

## 4.3 Summary

This chapter gives some details on the setup of the system used in the remaining of the thesis. Additionally, we introduce the three main angles of attacks. Especially, the first two angles focus on the feature extraction step while the last one focuses on the false positive removal step. Each of the three angles of attack is the subject of the following chapters, the next chapter detailing how to design attack against the keypoint detection step.





# Chapter 5

## Attacking keypoints

The performance of a CBIRS is mainly related to the ability of the description scheme in use to match images despite distortions. Without loss of generality, the search process eventually builds a ranked result list of images, such that highly ranked images are more likely to be true positive matches. True positives tend to have a particularly high score compared to the other images in the list. Two strategies can therefore be designed to delude the recognition of a system. First, it is possible to attack the image to be concealed such that its score gets dramatically reduced. Second, it is possible to attack the image by introducing in the picture visual elements that often match with other images from the database, such that the scores of these images get increased. It is of course possible to combine these two strategies. In this chapter, we present some attacks to delude the recognition of CBIRS based on SIFT descriptor.

This chapter is structured as follows. Sections 5.1 and 5.2 detail the two typical strategies to delude a system, respectively concealing an illegal image from identification and polluting the search results with false positives. It visually illustrates their impact using the well known Lena image. Section 5.3 details properties of new keypoints in attacked image. Section 5.4 evaluates the impact of these strategies against a large scale database used together with a state-of-art CBIRS using a multiple voting mechanism also single voting mechanism. It shows which strategies are effective in breaking the security of the SIFT description scheme. Section 5.5 concludes the chapter, suggests some counter measures, and works in the next chapter.

### 5.1 Reducing Scores by Keypoint Removal

Concealing an image from recognition by reducing its score translates into reducing the number of local descriptors that match. This can obviously be achieved by eliminating some of its keypoints.

As explained in section 4.1.2, keypoints are detected when local extremum are found in DoG with enough contrast and away from edges. Avoiding the detection of a keypoint at  $\mathbf{x} = (x, y, \sigma)^T$  is possible by applying on the image a visual patch of a certain size centered at  $(x, y)$  such that at least one of the three conditions for existing of a keypoint does not hold anymore. Patching images introduces visual artifacts, and, thus, a side effect of keypoint removal is the creation of new, undesired keypoints. We study in the following two keypoint removal strategies that are extreme in the sense that one is designed to minimize the local distortion on images, ignoring any potential keypoint creation, while the other takes little care of the distortion but rather eliminates as many keypoints as possible while lowering the number of creations.

### 5.1.1 Removal with Minimum Local Distortion (RMD)

For the keypoint  $\mathbf{x} = (x, y, \sigma)^T$ , this approach determines a patch  $\epsilon$  to be applied on the support region of the keypoint such that it minimizes the local distortion. Since, at scale  $\sigma$ , the Difference of Gaussian kernel  $\Delta G_\sigma$  has a limited support  $\mathcal{S}_\sigma$  in the spatial domain,  $\epsilon$  defined over  $(x, y) + \mathcal{S}_\sigma$  modifies the quantity  $D(x, y, \sigma)$  of a given amount  $\delta$ . In other word, for  $(u, v)$  in the neighborhood of  $(x, y)$ , the image is modified in  $I'(u, v) = I(u, v) + \epsilon(u, v)$  so that  $D'(\mathbf{x}) = D(\mathbf{x}) + \delta$ . The patch should be of minimal Euclidean norm to reduce the perceptual degradation. This resorts to an optimization under constraint:

$$\epsilon = \arg \min_{\epsilon: D'(\mathbf{x})=D(\mathbf{x})+\delta} \frac{1}{2} \|\epsilon\|^2. \quad (5.1)$$

The constraint being affine and the function to be minimized being convex, a simple Lagrangian resolution yields that

$$\epsilon = \frac{\delta}{\|\Delta G_\sigma\|^2} t_{(x,y)}(\Delta G_\sigma) \quad (5.2)$$

where  $t_{(x,y)}$  is the 2D translation operator of a shift  $(x, y)$ . This patch succeeds in controlling  $D(\mathbf{x})$ , however, as illustrated in Figure 5.1, it also modifies the DoG values in the neighborhood such as  $D(x+u, y+v, \sigma)$  with  $(u, v) \in [-4\sigma, 4\sigma]^2$ .

The first condition for the existence of a keypoint is that its DOG value is bigger than a contrast threshold. Let this threshold be  $C$ . We control the Removal with Minimum local Distortion (RMD) attack power by targeting a limited number of keypoints to be erased. We introduce a value  $\delta^+ > 0$  that defines the subset  $\mathcal{E}_{\delta^+} = \{\mathbf{x} : C < |D(\mathbf{x})| < C + \delta^+\}$ . Erasing keypoints in  $\mathcal{E}_{\delta^+}$  means that we decrease  $|D(\mathbf{x})|$  by an amount  $\delta = -\delta^+$  such that its new value is below the threshold  $C : |\delta^+| = |D(\mathbf{x})| - C$ . Obviously, when  $\delta^+$  grows, we deal with a bigger subset  $\mathcal{E}_{\delta^+}$ , and more keypoints are removed.

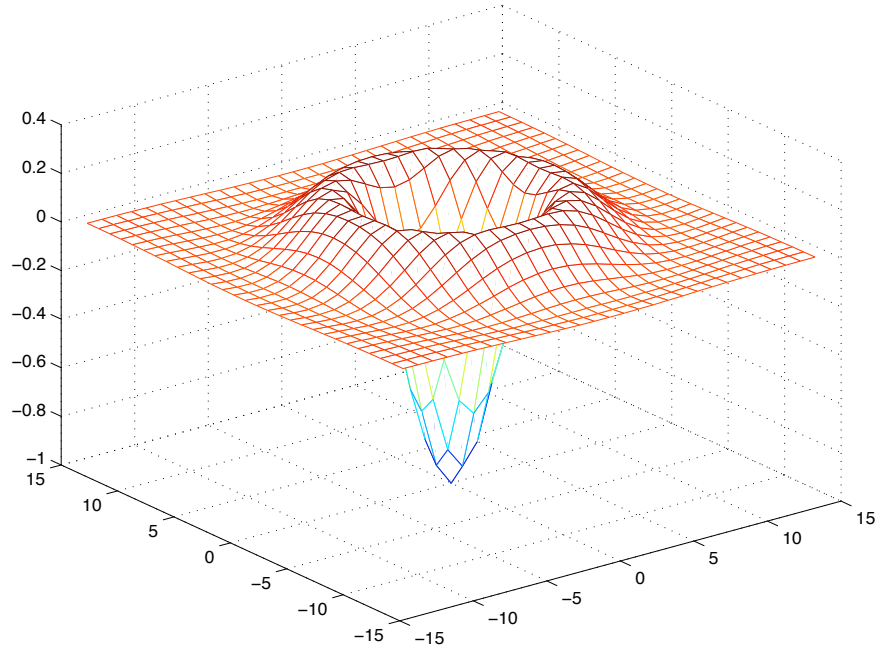


Figure 5.1: A RMD patch has been applied at scale  $\sigma = 2.54$  with  $\delta = -1$ . The z-axis represents the difference  $D'(x+u, y+v, \sigma) - D(x+u, y+v, \sigma)$ . If the patch succeeds to lower the targeted DoG coefficient of the amount  $\delta$ , it also impacts the DoG values around.

The most visible artifacts are caused by the removal of keypoints at higher scales, mostly because the patches have a bigger support and a stronger amplitude, as shown in Figure 5.2(a). Note with RMD, it is the local distortion around each keypoint that is minimized, not the global distortion on the whole image. The image attacked by RMD is displayed in Figure 5.2(b).

Table 5.1 summarizes the results when comparing the number of keypoints between the attacked and the original images. From Table 5.1, RMD with  $\delta^+ = 7$  removes 90.48% keypoints on Lena images and 77.49% on average over 1000 images. The attack on keypoints where  $|D(\mathbf{x})|$  is big requires a bigger  $\delta^+$ . Hence it will distort more the image (by equation 5.2). The maximum value of  $|D(\mathbf{x})|$  on Lena image is 19.

However, a undesired side effect of RMD is that it creates many new keypoints. Applying the patch  $\epsilon$  creates discontinuities in this local region. Table 5.1 shows that 72.91% keypoints (resp. 53.22%) in the attacked image for Lena (resp. on average over 1000 images) are indeed new keypoints. As illustrated on Figure 5.2(c), these new keypoints are located near the original keypoints, and hence potentially match with the original ones. A detailed analysis of these new key-

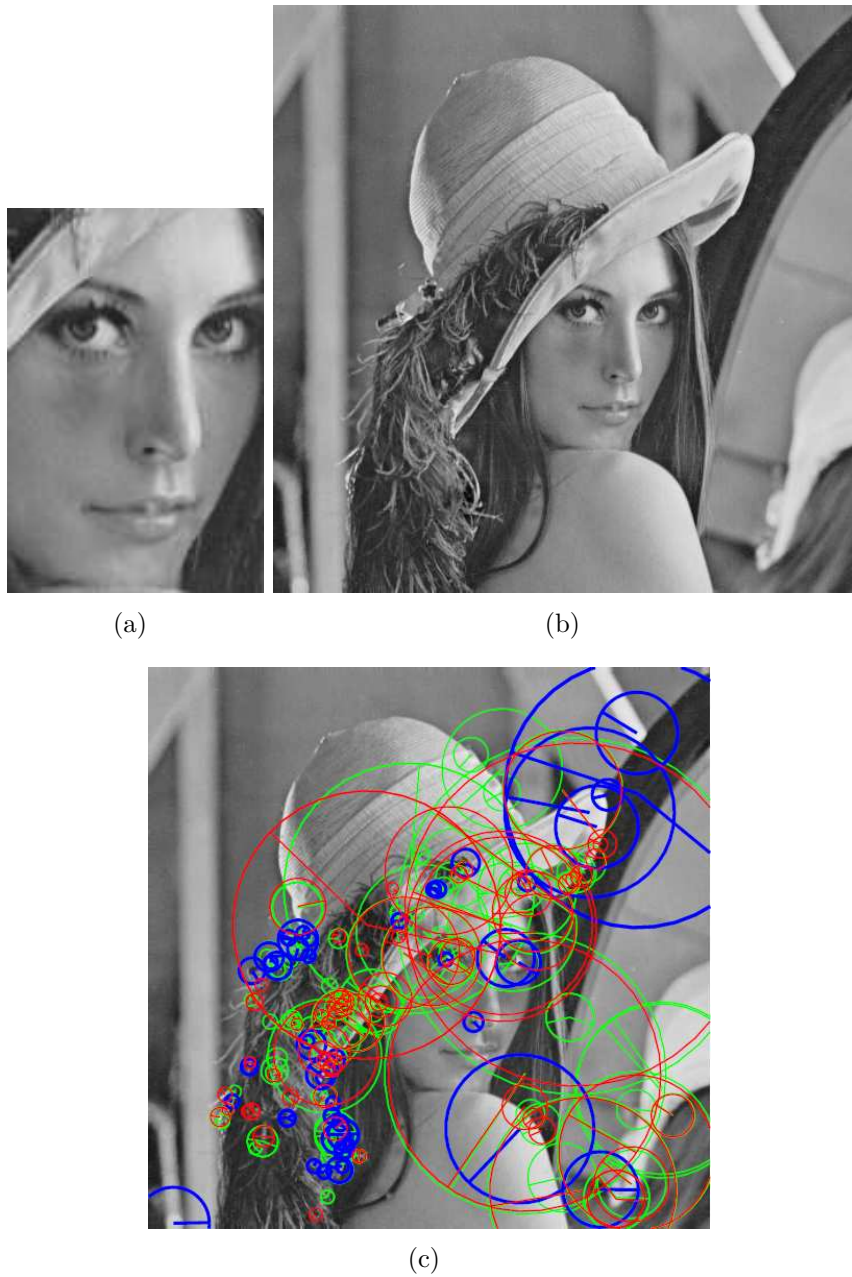


Figure 5.2: Visual distortions caused by RMD with  $C = 4$ ,  $\delta^+ = 4$  for keypoints at different scales: (a) close view of the face (see distortion on the lips and the shadow under the eye on the left) ; (b): whole image ; (c): the RMD attack with  $\delta^+ = 7$  on a subset of keypoints: unchanged (blue), deleted (green), and new (red) keypoints.

points properties is presented in the section 5.3. Another disadvantage of the RMD attack is that we can only control DoG value of each keypoint separately. It is then difficult to control the DoG values of a local region having a high density of keypoints.

### 5.1.2 Removal by smoothing

Local extrema of the DoG correspond to points located on significant discontinuities in the image. A straightforward way to avoid their detection consists in smoothing the image.

**Global Smoothing (GS)** Performing a smoothing on the whole image reduces the number of keypoints while avoiding the creation of new ones, as it does not introduce strong discontinuities. Experiments show this global smoothing is quite effective even with a Gaussian kernel of small variance. The value of  $\sigma$  is set empirically by conducting many experiments described in section 5.4;  $\sigma = 1.3$  is found as a good trade-off between visual and score (or number of unchanged keypoints) of the attacked images (see section 5.4). A greater  $\sigma$  would in turn remove more keypoints, but the quality of the resulting images would be worse. This global smoothing strategy is referred to as the GS strategy. Almost all keypoints not removed by the GS attack have high absolute DoG value.

**Local Smoothing (LS)** To increase the number of removed keypoints, a second step of smoothing can subsequently be applied, this time on a local basis. After a GS attack, a local smoothing (LS) phase is ran on each remaining keypoint: it replaces the  $n \times n$  region around the current keypoint by its smoothed version with a Gaussian kernel whose variance equals the keypoint scale, and checks whether this keypoint is still detected or not. This is in fact performed iteratively using regions of growing sizes ( $n = \{1, 3, 5, 7\}$ ), until the keypoint is no longer detected. Having  $n$  larger than 7 introduces too severe distortions in the images. Therefore, if the keypoint is still detected when  $n = 7$ , then the LS phase is aborted for that keypoint. This usually corresponds to keypoints with large scales. The strength of the LS attacks is controlled by the maximum value of  $n$ : the strongest LS attack, denoted LS7, is when  $n$  can take all values, while in the milder LS5 attack  $n$  varies in  $\{1, 3, 5\}$ . The visual distortions introduced by LS7 are shown in Figure 5.3. Note that we can attack with GS only, or with both GS and LS referenced to as GS+LS.

As shown in Table 5.1 and Figure 5.3(c), applying LS after GS results in a smaller number of created keypoints, as keypoints created by GS may be subsequently removed by LS. GS+LS7 removed 96.22% keypoints on Lena image and 94.25% on average over 1000 images. These figures are bigger than with the RMD

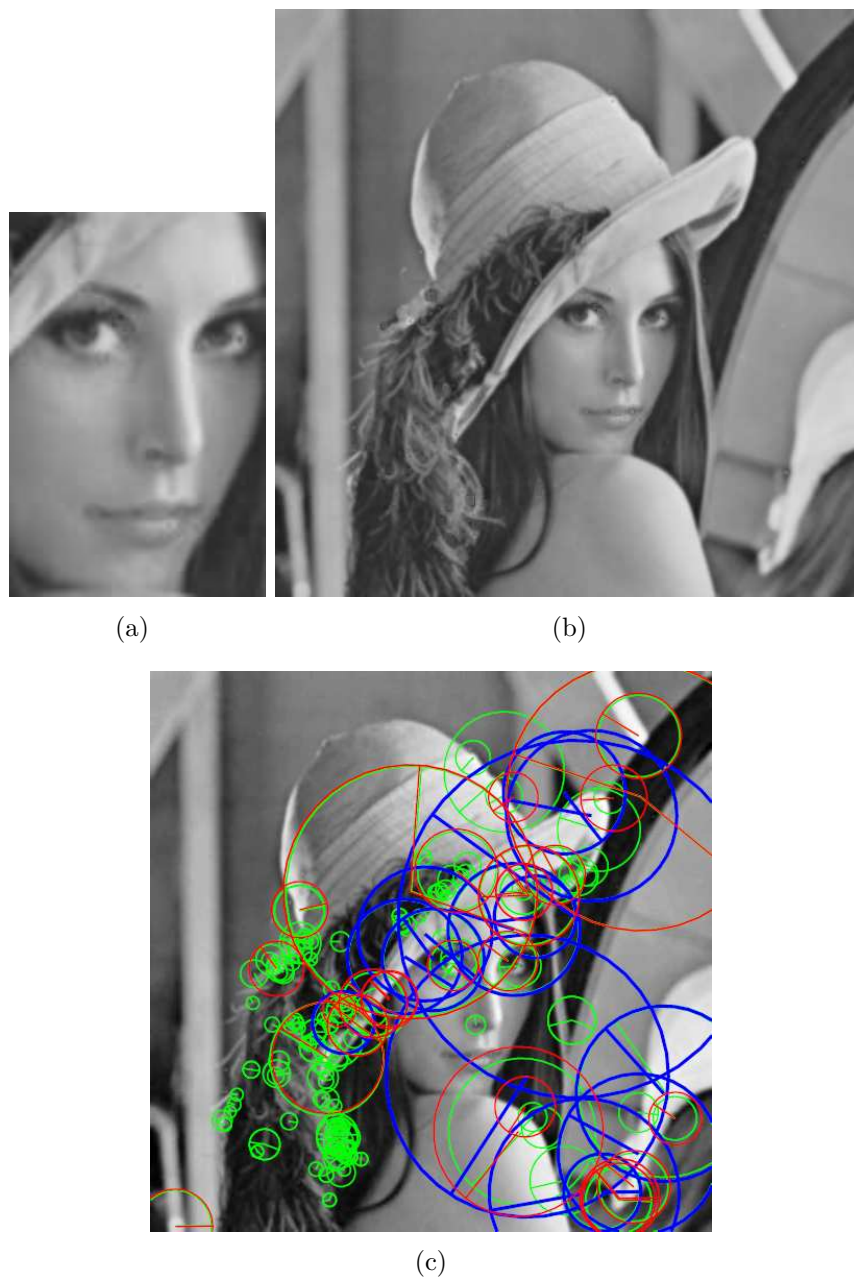


Figure 5.3: Visual distortions caused by GS+LS7: (c) close view of the face ; (b) whole image ; (c): GS+LS7 on a subset of keypoints: unchanged (blue), deleted (green), and new (red) keypoints.

attack. Both global and local smoothing also create new keypoints. The number of created keypoints, however, is by far smaller than with the RMD attack.

Table 5.1: Number of deleted and created keypoints by RMD with  $\delta^+ = 7$ , GS, GS+LS7, DS and GS+LS7+FMD. Original Lena image has 1218 keypoints; and the set of 1,000 images has 1026 keypoints on average.

Image	Attack	# KP deleted	% KP deleted	#KP created	% KP created	# KP after attack	PSNR in dB
Lena	RMD	1102	90.48	888	72.91	1004	27.78
	GS	1112	91.30	339	27.83	445	31.17
	GS+LS7	1172	96.22	270	22.17	316	30.31
	DS	1005	82.51	455	37.36	668	33.95
	DS+LS7	1007	82.68	437	35.88	648	33.90
	GS+LS7+FMD	1174	96.39	383	31.44	427	30.12
1,000 imgs	RMD	795	77.49	546	53.22	777	30.70
	GS	903	88.01	395	38.50	518	29.17
	GS+LS7	967	94.25	321	31.29	380	28.41
	DS	698	68.03	582	56.73	910	34.07
	DS+LS7	934	91.03	528	51.46	620	32.43
	GS+LS7+FMD	967	94.25	389	37.91	448	28.23

**Density-based Smoothing (DS): A variant of GS attack** GS attack is efficient to remove keypoints in the image. However, there are some regions/images which do not have any keypoints or have a sparse density of keypoints. Smoothing such regions/images is not necessary: few keypoints are affected while much visual distortion is introduced. It is more efficient to only smooth regions presenting a high density of keypoints. In this section, we use a variant version of the GS attack which takes into account the keypoints density and is referred to as Density Smoothing (DS).

The image is segmented to dense regions and non-dense regions by sliding a window of size  $50 \times 50$ . A smaller window makes DS more time consuming. A bigger window makes DS less efficient in spotting high keypoint density regions. A region is considered as dense if the number of keypoints in the window is more than 60. It means that there are approximately 1 keypoint per  $7 \times 7$  pixels square. Dense regions are smoothed by the Gaussian kernel with  $\sigma = 1.3$ , like the GS attack. Figure 5.4(a) illustrates dense regions in Lena image. The attacked image after applying DS is shown in Figure 5.4(b).

On average over 1,000 images, the number of remaining keypoints after the DS attack is 910. It is higher than 518 which is number of keypoints after applying GS (Table 5.1). However, the average PSNR after DS attack is 34.07 dB. It is significantly higher than the average PSNR for GS (29.17 dB in Table 5.1).



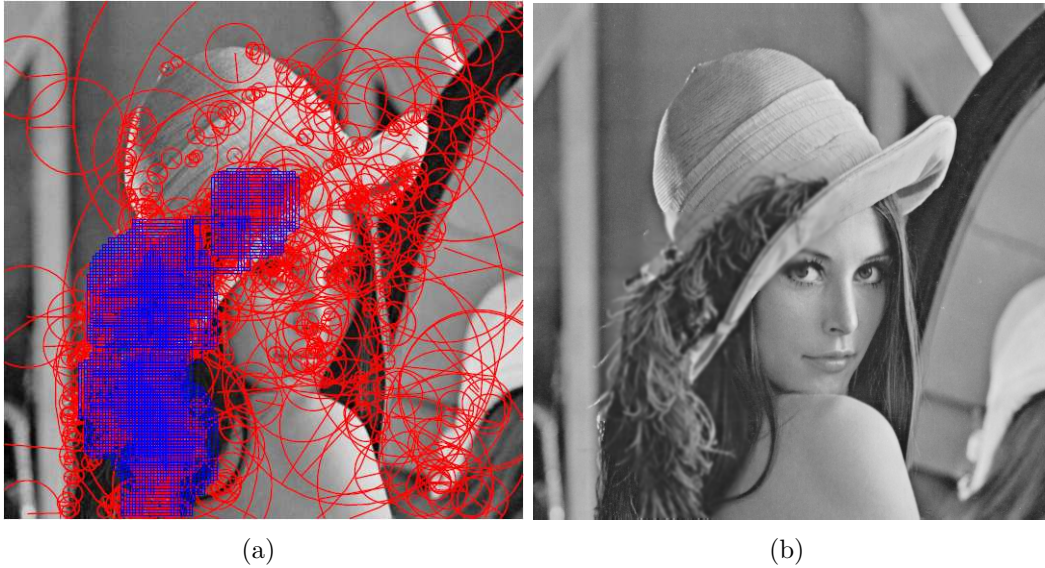


Figure 5.4: Density-based Smoothing on Lena image: (a) red circles are original keypoints. Dense regions are marked with blue rectangles ; (b) visual distortion after DS.

## 5.2 Forging New Keypoints

As mentioned in the chapter introduction, the other strategy is to deliberately create new keypoints so that their descriptors will match with wrong images in the database, increasing their scores. Hopefully, querying with the attacked image will bring at the top of the result list some other images, while the original one may either be further away in the list or not in the list at all. This also raises false alarms spoiling the trust that users have on the system.

The strategy we use for forging new keypoints with Minimum local Distortion (FMD) is symmetric to the RMD attack. In this case, we address the local extrema in the subset  $\mathcal{F}_{\delta^-} = \{\mathbf{x} : C - \delta^- < |D(\mathbf{x})| < C\}$ , and add patches that strengthen the contrast in the neighborhood of keypoints. This also reduces the gap between the absolute values of the first and second eigenvalues, such that the condition on the Hessian matrix gets verified most of the times.

To be created with FMD, new keypoints must meet two conditions: (i) belong to the two first octaves, such that the introduced distortion is small, (ii) be located relatively far away from existing keypoints such that the resulting descriptors are rather different from the existing ones. We thus create keypoints where the neighborhood of size 8 pixels is free from original keypoints. This size of 8 pixels ensures that the new keypoints are not near to some original keypoints and their descriptor is far enough from the descriptor of the original keypoints. The distance

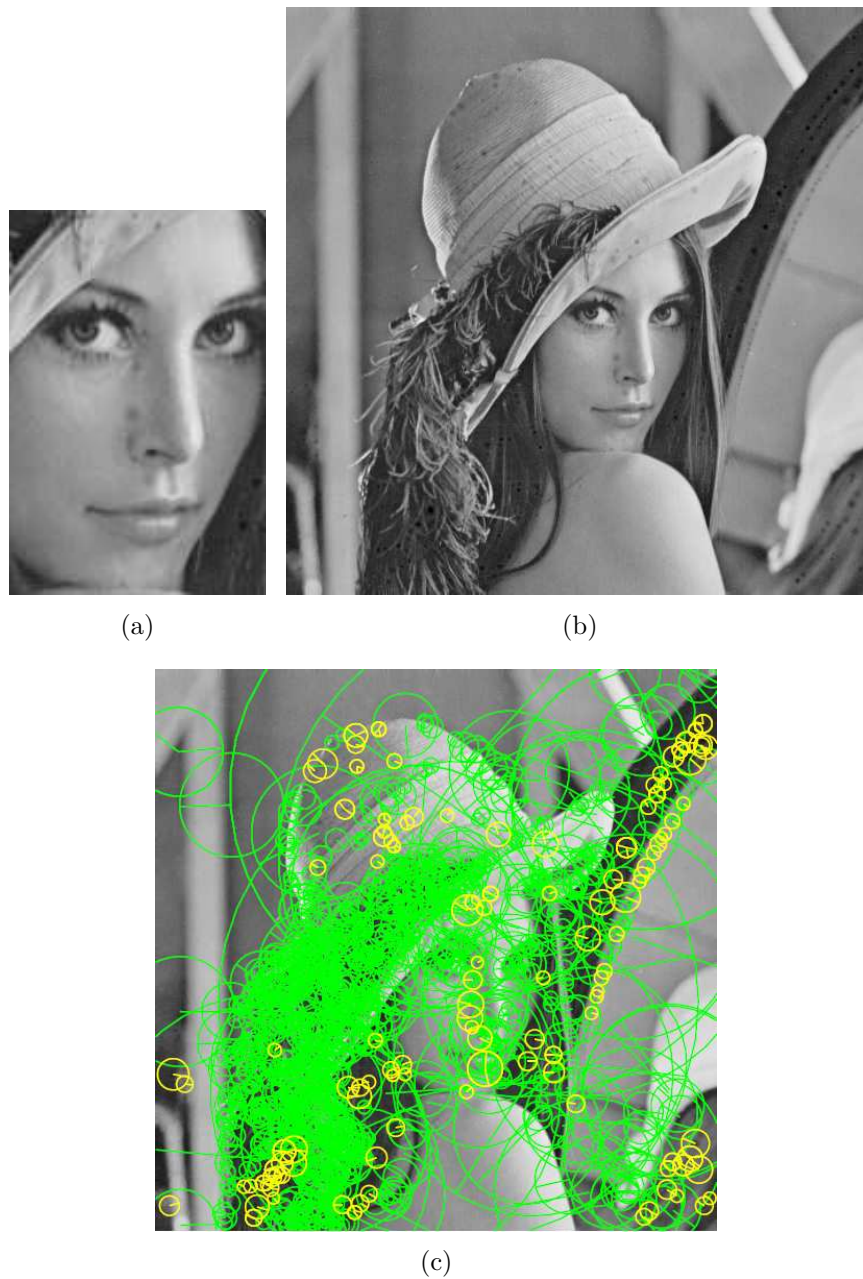


Figure 5.5: Visual distortions caused by FMD with  $C = 4$ ,  $\delta^- = 3$ : (a): close view of the face; (b): whole image. The keypoints created under (i) and (ii) conditions; (c): Illustration of MFD on a subset of keypoints: original (green) and new (yellow) keypoints

between descriptors is bigger than 450 for keypoints on octave -1 and 0 on Lena

image (see figure 5.6). The new keypoints yield salt and pepper noise artifacts. This is shown in figure 5.5.

The goal is achieved if the score of some images in the database are higher than the score of the original image. However, we can not forge many new keypoints in the attacked image to limit the introduced distortions. Hence, we lead GS+LS attack to remove some keypoints before applying FMD. The combination is referenced to as GS+LS+FMD.

Tables 5.2 and 5.3 show that there are 111 keypoints created by FMD on the first two octaves of Lena image (68 on octave -1, and 43 on octave 0), and 68 created keypoints on average over 1000 images (36 on octave -1, and 32 on octave 0). Table 5.1 shows the number of deleted and created keypoints with GS+LS+FMD. On average over 1000 images, the number of removed and created keypoints are 94.25% and 37.91%, respectively. The average PSNR between attacked and original images is 28.23 dB.

### 5.3 Properties of New Keypoints

As presented in sections 5.1 and 5.2, the new keypoints in the attacked image come from two sources. They are created as a side effect of the removal method (RMD or GS+LS) or deliberately by FMD. This section details the properties of these keypoints and their impact on recognition. Tables 5.2 and 5.3 give indications on the locations and scales of the keypoints created as a collateral effect of RMD or GS+LS. It focuses on the new keypoints belonging to the first two octaves (-1 and 0), because it is where the majority of creations takes place. For each such new keypoint, we compute the distance from its location to the nearest original keypoint in the same octave. Averaging over all created keypoints gives the “avg distance” line in Tables 5.2 and 5.3. Those tables also give the average scale factor between new keypoints and their nearest original keypoints, regardless of the octave. It can be seen that the new keypoints created by LS+GS are farther away in location than those created by RMD. However, the keypoints created by RMD have an average scale factor smaller than those created by LS+GS. In other words, the new keypoints created by RMD are moved farther away in scale than those created by GS+LS. Table 5.2 and Table 5.3 also show average distance of the new keypoint created by FMD from the nearest original keypoint. As expected, this distance is big. However, their scale factors are similar than the nearest original keypoints.

To understand how the shift in space or in scale impacts on keypoint descriptors, we carefully track the whole descriptor creation process for all keypoints of the Lena image. We then replay the creation for these keypoints, but we artificially shift each keypoint away from its original location in space (or in scale)

Table 5.2: Properties of the new keypoints: average distance per octave and scale factor between a new keypoint and the nearest original keypoint. The last column only shows the creation of the keypoints thanks to FMD, after applying GS+LS7. Lena image

Attack Octave	RMD, $\delta^+ = 7$		GS+LS7		GS+LS7+FMD	
	-1	0	-1	0	-1	0
# KP created	613	188	11	147	68	43
avg distance	3.1	9.1	4.1	7.8	36.3	34.0
avg scale factor	0.21	0.62	0.47	0.80	0.56	0.79

Table 5.3: Properties of the new keypoints: average distance per octave and scale factor between a new keypoint and the nearest original keypoint. The last column only shows the creation of the keypoints thanks to FMD. Average on 1000 images

Attack Octave	RMD, $\delta^+ = 7$		GS+LS7		GS+LS7+FMD	
	-1	0	-1	0	-1	0
# KP created	329	141	28	196	36	32
avg distance	4.1	8.7	4.8	10.7	34.6	36.3
avg scale factor	0.26	0.55	0.45	0.77	0.51	0.75

and measure the euclidean distance between the descriptor associated with that shifted keypoint and its original version. Figure 5.6 shows the result of this experiment when considering spatial shifts while Figure 5.7 is for scale shifts. Figures 5.6 and 5.7 shows that shifting keypoints in the image indeed increases the distances in the high-dimensional space.

For a given distance between descriptors, keypoints on small octaves need less location shift. This is expected as the corresponding support regions are smaller, so rather different with a small shift. For scale shift, the behavior of the distance is different. The shift in scale increases or decreases the support region size. However, the weighting during the descriptor computation give more influence to the central area of the support regions. Hence, the shift in scale on different octaves has similar impact on distance.

From Table 5.3, the distance between the new keypoints created as a side effect of GS+LS and RMD and the original keypoints on the same octave is 4.8, 4.1 for octave -1 and 10.7, 8.7 for octave 0. According to Figure 5.6, these new keypoints are likely to match with these original keypoints. We also see that the distance between a keypoint created by FMD and the nearest original keypoint (on same octave) is 34.6 and 36.3 for octave -1 and 0, respectively. Hence, the new keypoints created by FMD are unlikely to match to these nearest original keypoints. The scale factors between the new keypoints created by FMD and the

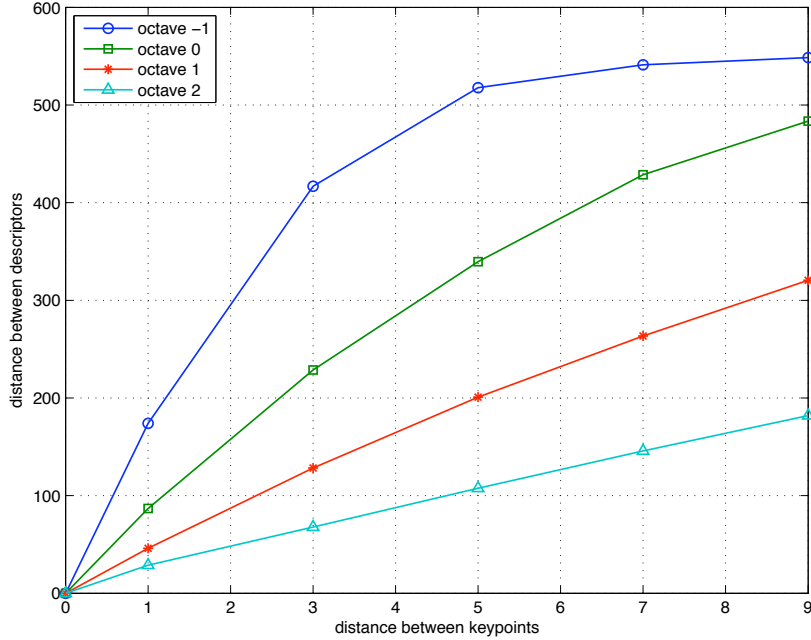


Figure 5.6: Euclidean distance between descriptors vs. distance between keypoints (Lena image).

nearest original ones (regardless the octave) are ranging from 0.51 to 0.75. These scale factors are not very far from the nearest original, but these new keypoints are ensured to be far away (at least 8 pixels) from the nearest original keypoints (by condition (ii) in section 5.2). Hence, in conjunction with Figure 5.6, we draw the conclusion that the keypoints created by FMD are also unlikely to match with some original keypoints.

## 5.4 Large Scale Experiments

We evaluate the efficiency of the above mentioned attacks using a large scale image collection and a real CBIRS. The experiment setup is given in section 4.1. The search process uses the multiple voting scheme of NVTree. The geometric verification is not considered in this experiment.

### 5.4.1 Results

The first experiment illustrates the efficiency of the keypoint removal. We average the scores of 1,000 original images when searched with 4 RMD attacks

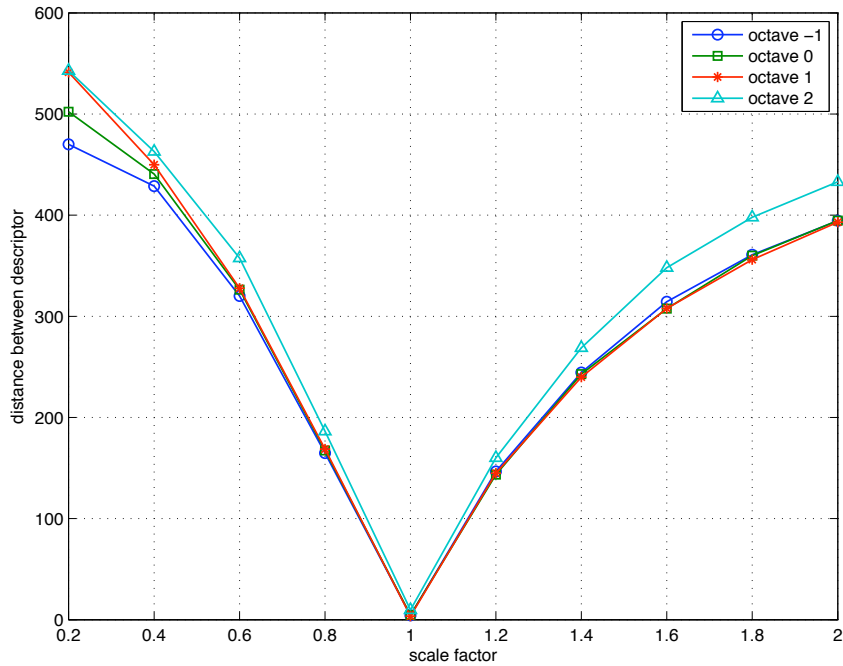


Figure 5.7: Euclidean distance between descriptors vs. scale shift of a given factor (Lena image).

corresponding to  $\delta^+ = \{3, 4, 6, 7\}$  and 4 GS+LS attacks with  $\sigma = 1.3$  for GS and  $n = \{1, 3, 5, 7\}$  for LS.

We also collect the average PSNR observed on the attacked images for each family of attack. The results of this experiment are on Figure 5.8, which additionally depicts scores and PSNR for JPEG attacks with varying quality factors, for comparison.

As expected, both the score and the PSNR drop down as the attack strength increases (through increasing parameters  $n$  or  $\delta^+$ ). It also confirms that the RMD attack yields a better PSNR than GS+LS, as it was designed to introduce the minimum distortion. RMD is less efficient than GS+LS at lowering the matches, as it creates more descriptors that are not so different from their original counterpart.

Figure 5.9) shows the results of the experiment, focusing on 10 “security” attacks. Along the X-axis, the attacks differ by the keypoint removal process with varying parameters as described in sections 5.1 and 5.2 and whether the FMD attack is turned off (left side) or on (right side). The Y-axis of the figure shows, for each attack, the average scores (over the 1,000 queries) of the original images that are expected to match with the attacked copies. It also shows the scores of the first, second and third best matching images that are different from the

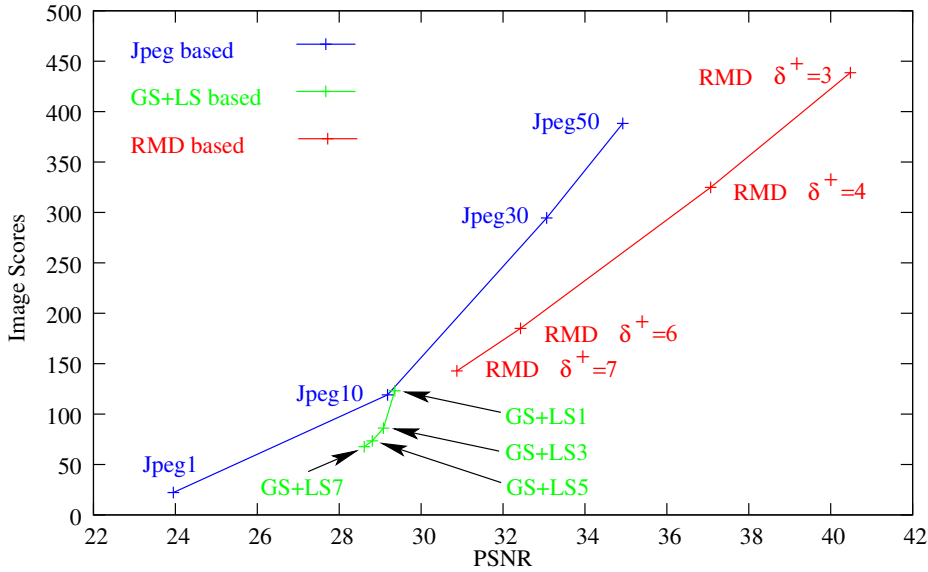


Figure 5.8: Average score vs. PSNR (dB). JPEG-, GS+LS and RMD based attacks.

original images. When the image having rank #1 in the result list is the original image, then the system succeeded in recognizing that image. When the image at rank #1 is not the original image, then the system failed. For comparison, the score of the original images when searched with themselves is around 1,000 and the second best is below 100.

### 5.4.2 Analysis

Figure 5.9 shows that solely removing keypoints (the 5 attacks on the left) fails at deluding the system. There are mainly two reasons for this. First, the gap in scores between the original image and the best non-matching image is very large, and therefore, at least 90% of the keypoints should be removed to shrink that gap, which is impossible in practice due to the severe visual distortion this would cause. Second, keypoint removal creates new keypoints in their vicinity and it turns out the corresponding vectors match with the original ones. We also see in this figure that the gap in score between the original image and the best non-matching image for GS or GS+LS attack is smaller than RMD attack. This consolidates the fact that keypoints removal by smoothing is stronger than RMD.

When the FMD attack is turned on then things are changing. With the GS+FMD attack, although the original image is found, the score of the best other image jumps and get much closer. A detailed examination of the matching shows that the new forced keypoints are created at small scales and tend to match



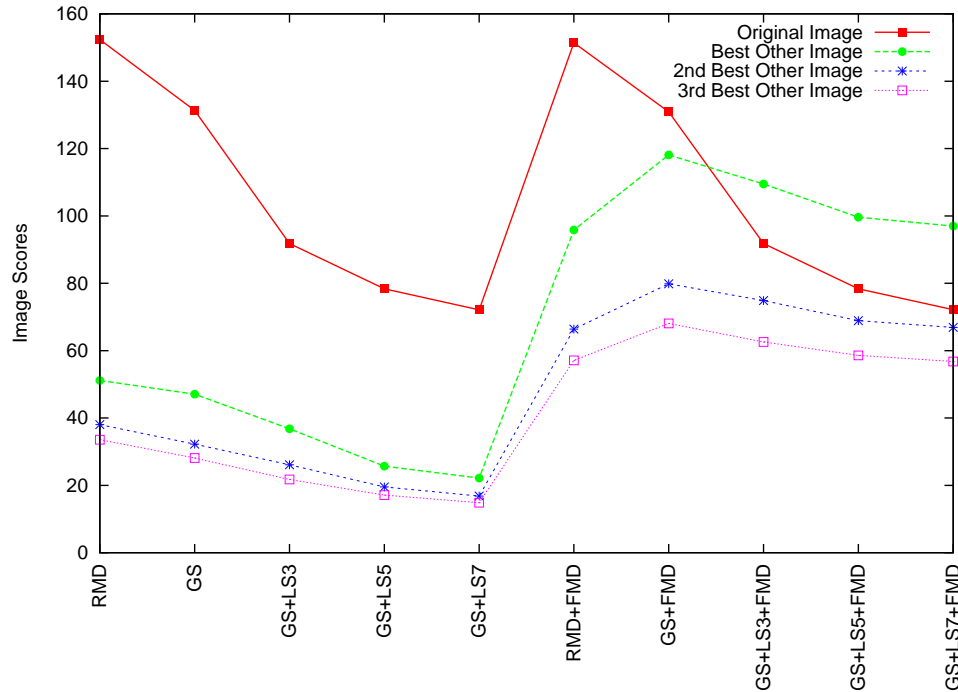


Figure 5.9: Image scores in realistic settings. X-axis: 10 selected “security” attacks. Y-axis: for each family of attack, the average scores over the 1,000 queries of the original images and of the three other best matching images.

with regular images from the database that have repetitive visual patterns such as bricks, small windows on large facades, tiles, venitian blinds, . . . Interestingly, despite the size of the collection, few images have such patterns (see Figure 5.10 for an example) and therefore they concentrate the votes when scoring. This is due to the multiple voting scheme used in these experiments, that allow multiple matches for one descriptor. Indeed, only with a small number of keypoints created by FMD, the attacked image can get many votes from keypoints belonging to regular images of the database.

Increasing the strength of these attacks by adding local smoothing succeed in deluding recognition. With the last three attacks on the right of the Figure 5.9, not only few other images concentrate matches (thanks to FMD) but the number of keypoints in the attacked image drops (thanks to GS+LS7), reducing the number of true matches. It is essential to note that the attacked image is not concealed from the CBIRS (as its unmodified copy has rank #2), but it gets “hidden” behind another image that better matches. This is a key result.





Figure 5.10: Images of the database often ranked #1 when the attack succeeds.

Overall, these experiments are a proof of concept that “security” attacks can conceal an image, at the cost of a distortion of around 30 dB in PSNR (see Table 5.1). A “robustness” attack, such as a JPEG compression with  $Q = 1$ , achieves the same goal but with a PSNR of 23.68dB (see Figure 5.8).

### 5.4.3 Discussion

The main result of the proposed attacks is that the score of the original is now lower than the scores of some other irrelevant images of the database (Figure 5.9). This result comes from the strength of the GS, LS and FMD attacks, and from the multiple voting mechanism in the system. An additional result with single voting for three attacks GS + LS3 + MFD, GS + LS5 + MFD, GS + LS7 + MFD is shown in Figure 5.11. From this figure, we see that the original image are still on the top of the result list. This is because each best competitor image in database only receives a maximum of one vote per each keypoint created by FMD <sup>1</sup>. The score of the best competitor image fails being bigger than the score of original image.

## 5.5 Summary

In this chapter, we show that a “security” attack dedicated to a given CBIR technique is more efficient than “robustness” attacks. A significant part of the work presented in this chapter has been published in [19].

<sup>1</sup>Let number of created keypoint be  $n$ , the best competitor image will have maximum  $n$  votes.

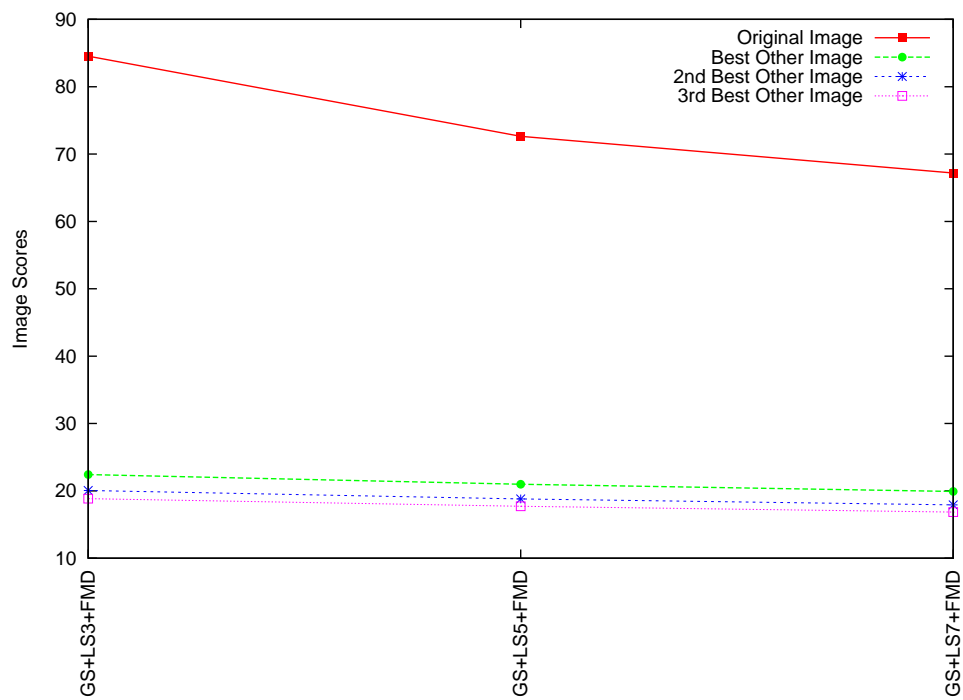


Figure 5.11: Image scores with single voting mechanism. X-axis: three selected “security” attacks. Y-axis: the average scores over 1,000 queries of the original images and of the three other best matching images.

There are some lessons learned from this work.

- Removing keypoints is not sufficient to delude a system in real settings. All attacks on the left of Figure 5.9 fail. Although we removed 90% of keypoints, the gap between the scores of the original image and the best competitor is so huge. This is due to the robustness of the system. CBIRS still works well even with a small number of unchanged keypoints remaining in the attacked image.
- The RMD attack introduces little distortion but it creates new keypoints in the vicinity so that they still match with the original. Hence, RMD is not used in the remaining of the thesis. GS+LS are much more efficient but still not enough: removing so many keypoints would severely degrade the image.
- The forgery of new keypoints is necessary, but on the condition that the new keypoints massively match with a restricted set of images in the database. In addition, this will happen only on systems using multiple voting.
- The geometric verification is not used in the experiments. If geometric verification is used (see later in next chapter), it can remove keypoints created by FMD. In that case, FMD can be useless.
- This chapter tries to delude the recognition of CBIRS attacking the keypoint detection step. Another possibility is to attack the keypoint description step. In the next chapter, we propose some attacks changing the orientation of keypoints such that it changes support regions. It is likely to change the final descriptor so that it is unlikely to match with the original descriptor.

# Chapter 6

## Attacking descriptors

The previous chapter introduces a complex mixtures of keypoint removal and keypoint creation. Removing keypoints reduces the number of matches ; creating keypoints produces false positives. Overall, the previous chapter exploits the way the Difference of Gaussian (DoG) value of a keypoint is used in the SIFT extraction process. However, we can not remove all the keypoints in the image, in particular those at high scales: their modification severely degrades the image over their big support region.

To further disrupt the matching between attacked image and original image, we change the list of descriptors found by the  $K$  nearest neighbors search at retrieval time. In other words, we want to preclude the appearance of descriptors of the original image in  $K$ -nn list. This is done by pushing the descriptors of the attacked image far way from their original version. Hence, an attacked descriptor unlikely match with the original one. One way to do this is to change the orientation of keypoints, as this has a direct impact on the SIFT description.

This chapter explores how this strategy can be put into practice. The orientation of a keypoint is changed by modifying its support region. The attack includes two steps. In a first step, a learning process based on SVM learns the hyperplane <sup>1</sup> separating support regions of different orientations. After that, given a keypoint in one side of the hyperplane, we modify its support region with the minimum distortion such that it is pushed to the other side of the hyperplane.

This chapter is structured as follows. Section 6.1 demonstrates the impact of the keypoint orientation on its description. Section 6.2 details the method to change the orientation of keypoints. Section 6.3 evaluates the proposed method at patch level, image level and large scale recognition. Section 6.5 concludes the chapter.

---

<sup>1</sup>In this chapter, we use SVM with RBF kernel. Hence, when we say hyperplane, it is implication of the hyperplane in kernelized space.

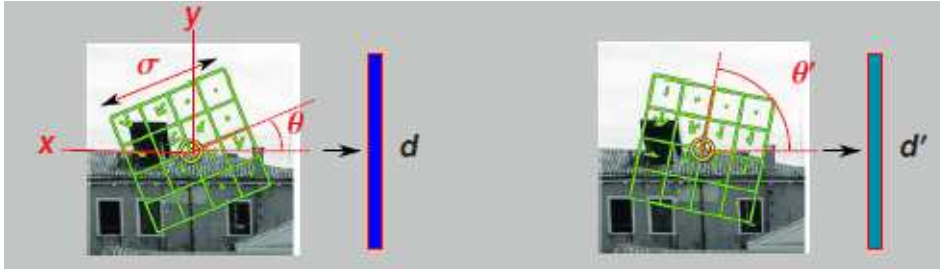


Figure 6.1: The role of the orientation in the description of the keypoint

## 6.1 SIFT: impact of keypoint orientation on keypoint descriptor

This section first shows that the keypoint orientation has an important role in the descriptor computation. An experiment then quantifies the impact of orientation on descriptors.

### 6.1.1 From orientation to descriptor

As presented in section 4.1.2, at description step, the coordinates of the descriptor and the gradient orientations are rotated relatively to the main keypoint orientation to gain orientation invariance. Hence, shifting the orientation of keypoints is likely to change the final high-dimensional descriptor. This strategy does not remove the keypoints but reduces the likelihood that they will match with their original vectors. This is illustrated in Figure 6.1. The change of orientation from  $\theta$  (left figure) to  $\theta'$  (right figure) change the original descriptor  $d$  to a new descriptor  $d'$ . Note that this is different from rotating the whole support region, which has no impact on description. Intuitively, the shift must be large enough in order for the final descriptor to be sufficiently different and to preclude the matching.

### 6.1.2 Changing Orientations Impacts Descriptors

To see the relation between orientation and descriptor, we first describe Lena using the open-source VLFeat package [86]. We then patch the code of VLFeat to artificially change the orientation of the detected keypoints by a multiple of  $\pi/18$ . We then launch 35 descriptions of Lena, each time increasing the orientation change from  $\pi/18$  to  $35\pi/18$ . To foresee an acceptable visual quality of the attacked image, we focus on the keypoints having a small support region. In other words, only the keypoints on first three octaves  $\{-1, 0, 1\}$  are considered. Figure 6.2 shows the impact of this forced change of orientation by plotting the distance in the feature space between the original descriptors and the descriptors

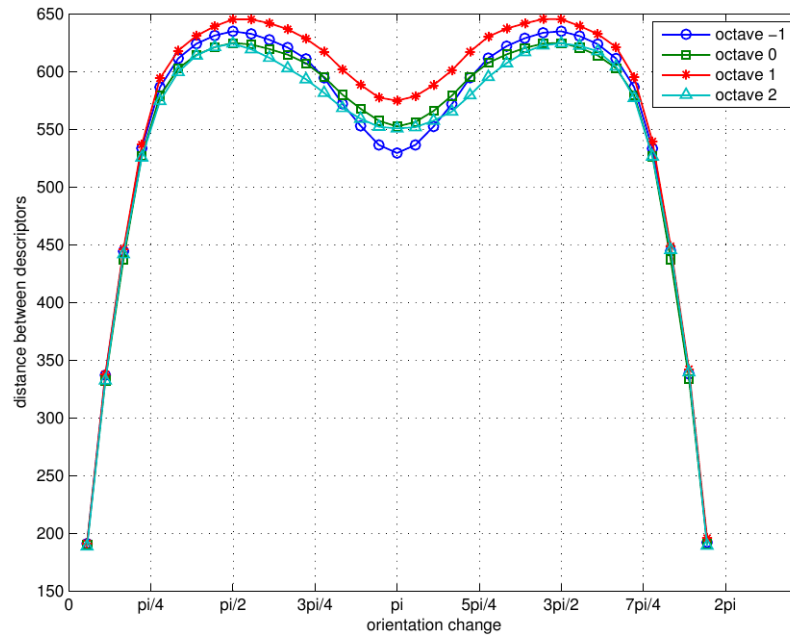


Figure 6.2: Euclidean distance between descriptors as a function of the distance in radian between the keypoint orientation.

after the orientation changes. We see that the larger distances in feature space are reached when the keypoint orientation is changed by  $\pi/2$  and  $3\pi/2$ . In other words, if the new orientation is orthogonal to the original one, then the maximum difference between descriptors is obtained. This applies to other images as well.

## 6.2 SVM for Changing Orientations

This section describes our method to force the orientation of keypoints. In short, we learn which patch  $\epsilon$  should be applied to a specific support region to change the orientation by  $\pi/2$ . As mentioned in section 4.1.2, the main orientation of a keypoint is computed based on distribution of gradient orientation of pixels in the support region. Intuitively, the keypoints having similar orientation should have the same distribution of gradient orientation. In other words, they should belong to the same class in view of gradient orientation distribution. If we consider the keypoints having the same orientation as a class, now, the problem of changing the keypoint orientation is turned into a problem of moving a keypoint from a given class to another class. A classical method to separate two classes is SVM. Our method relies on a collection of SVMs. Each SVM determines the hyperplane separating the keypoints having the orientation  $\theta_1$  from the keypoints with orientation  $\theta_2 = \theta_1 + \pi/2$ . To facilitate learning, reduce the noise and

be more effective, the orientation space is quantized in bins of length  $\pi/18$ . It means that after quantization, there are 36 bins covering all orientations. We use 18 SVM classifiers trained for the 18 pairs of orthogonal orientations (e.g., from orientations ranging in  $[0, \pi/18]$  to orthogonal orientations ranging in  $\pi/2 + [0, \pi/18]$ ). Table 6.1 illustrates patches belonging to different orientation bins.

To train the SVMs, we first determine all the keypoints and their orientation for a set of 1,000 queries images (same as section 4.1). We only keep keypoints belonging to the octaves  $\{-1, 0, 1\}$  as their support regions are small, facilitating patching them visually while not too severely distorting the images. There are 954,285 keypoints on these octaves and 83% belong to the octave  $-1$ . We then normalize all support regions to be of size  $(12 \times 12)$ , which is the average size of patches on octave  $-1$ . A larger patch size requires a longer time to train the SVM and to find the patch  $\epsilon$  (as presented below). The keypoints from octaves 0 and 1 are also resized to  $(12 \times 12)$ . After resizing, we map all of them from grayscale to range  $[0, 1]$  and then stored each resized and normalized patch in a vector  $\mathbf{r}$  of  $L = 12 \times 12 = 144$  components.

The set of keypoints is then divided into classes according to their orientations  $\theta$ . Let  $\mathcal{X}_1 = \{(\mathbf{r}_i, \ell_i)\}_i$  be the training set of normalized support regions  $\mathbf{r}_i$  of a given orientation  $\theta_1$ , forming the class labelled by  $\ell_i = +1$ .  $\mathcal{X}_2 = \{(\mathbf{r}_j, \ell_j)\}_j$  is the training set of normalized support regions  $\mathbf{r}_j$  whose orientation is  $\theta_2 = \theta_1 + \pi/2$ , forming the dual class labelled by  $\ell_j = -1$ . At training time, the SVM in charge of  $\theta_1$  and  $\theta_2$  learns the hyperplane parameters  $(\mathbf{w}, b)$  separating  $\mathcal{X}_1$  and  $\mathcal{X}_2$  as solution of:

$$\begin{aligned} \ell_k \cdot (\langle \mathbf{w}, \Phi(\mathbf{r}_k) \rangle + b) &\geq 1 \quad \forall \mathbf{r}_k \in \{\mathcal{X}_1, \mathcal{X}_2\}, \\ \text{with } \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle &= \sum_{k:\alpha_k > 0} \alpha_k \ell_k K(\mathbf{x}, \mathbf{r}_k), \end{aligned}$$

where  $\Phi$  maps  $\mathbf{x}$  to an higher dimensional space,  $\alpha_k$  are the Lagrange multipliers, and  $K$  is the Radial Basis kernel Function (RBF):

$$K(\mathbf{x}, \mathbf{r}_k) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{r}_k) \rangle = \exp\left(-\frac{\|\mathbf{x} - \mathbf{r}_k\|^2}{2\sigma^2}\right). \quad (6.1)$$

The RBF kernel is chosen because it is tuned by only one parameter  $\sigma$ . In our experiments,  $\sigma$  equals 1 (the default value in MATLAB library). Once trained, the SVM is used to determine the patch  $\epsilon$  of minimum norm to be added to a region  $\mathbf{r} \in \mathcal{X}_1$ , such that  $\mathbf{r} + \epsilon \in \mathcal{X}_2$ . This asks to solve the following optimization:

$$\min \frac{1}{2} \|\epsilon\|^2 \quad (6.2)$$

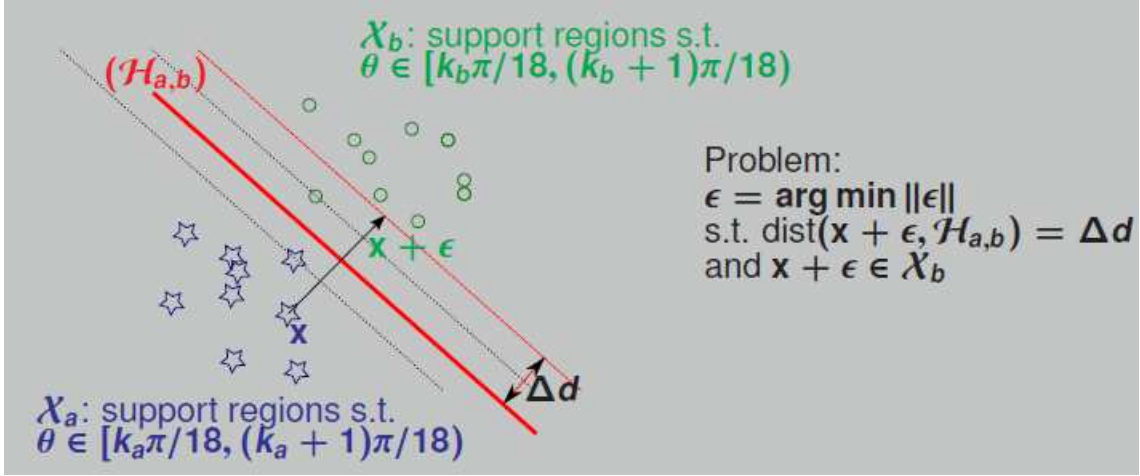
$$\text{s.t. } \sum_{k:\alpha_k > 0} \alpha_k \ell_k K(\mathbf{r} + \epsilon, \mathbf{r}_k) + b = -\Delta d, \quad (6.3)$$

$$\text{and } 0 \leq r_i + \epsilon_i \leq 1, \quad \forall i \in \{1, \dots, L\} \quad (6.4)$$

orientation	$[0, \pi/18]$ 	$(\pi/18, 2\pi/18]$ 	$(2\pi/18, 3\pi/18]$ 	$(3\pi/18, 4\pi/18]$ 
orientation	$(4\pi/18, 5\pi/18]$ 	$(5\pi/18, 6\pi/18]$ 	$(6\pi/18, 7\pi/18]$ 	$(7\pi/18, 8\pi/18]$ 
orientation	$(8\pi/18, 9\pi/18]$ 	$(9\pi/18, 10\pi/18]$ 	$(10\pi/18, 11\pi/18]$ 	$(11\pi/18, 12\pi/18]$ 
orientation	$(12\pi/18, 13\pi/18]$ 	$(13\pi/18, 14\pi/18]$ 	$(14\pi/18, 15\pi/18]$ 	$(15\pi/18, 16\pi/18]$ 
orientation	$(16\pi/18, 17\pi/18]$ 	$(17\pi/18, 18\pi/18]$ 	$(18\pi/18, 19\pi/18]$ 	$(19\pi/18, 20\pi/18]$ 
orientation	$(20\pi/18, 21\pi/18]$ 	$(21\pi/18, 22\pi/18]$ 	$(22\pi/18, 23\pi/18]$ 	$(23\pi/18, 24\pi/18]$ 
orientation	$(24\pi/18, 25\pi/18]$ 	$(25\pi/18, 26\pi/18]$ 	$(26\pi/18, 27\pi/18]$ 	$(27\pi/18, 28\pi/18]$ 
orientation	$(28\pi/18, 29\pi/18]$ 	$(29\pi/18, 30\pi/18]$ 	$(30\pi/18, 31\pi/18]$ 	$(31\pi/18, 32\pi/18]$ 
orientation	$(32\pi/18, 33\pi/18]$ 	$(33\pi/18, 34\pi/18]$ 	$(34\pi/18, 35\pi/18]$ 	$(35\pi/18, 36\pi/18]$ 

Table 6.1: Illustration of patches in 36 orientation bins used for SVMs



Figure 6.3: The way we find  $\epsilon$ .

where  $\Delta d > 0$  is the distance from  $\mathbf{r} + \epsilon$  to the hyperplane  $(\mathbf{w}, d)$ . The value of  $\Delta d$  is somehow related to the probability that we achieve our goal:  $\mathbf{r} + \epsilon \in \mathcal{X}_2$ . If  $\Delta d$  is small,  $\mathbf{r} + \epsilon$  will likely lie on the boundary between  $\mathcal{X}_1$  and  $\mathcal{X}_2$ . In the other hand, if  $\Delta d$  is big,  $\mathbf{r} + \epsilon$  will be pushed deep inside  $\mathcal{X}_2$ . Figure 6.3 illustrates the problem of finding  $\epsilon$  by using SVM.

Eq. (6.4) ensures that the modified region remains in the range  $[0, 1]$ . Defining scalars  $a_k = \alpha_k \ell_k K(\mathbf{r}, \mathbf{r}_k)$ , and vectors  $\mathbf{c}_k = 2(\mathbf{r} - \mathbf{r}_k)$ , Eq. (6.3) can be rewritten as:

$$\sum_{k:\alpha_k>0} a_k \exp\left(-\frac{\mathbf{c}_k^\top \epsilon + \|\epsilon\|^2}{2\sigma^2}\right) + b + \Delta d = 0. \quad (6.5)$$

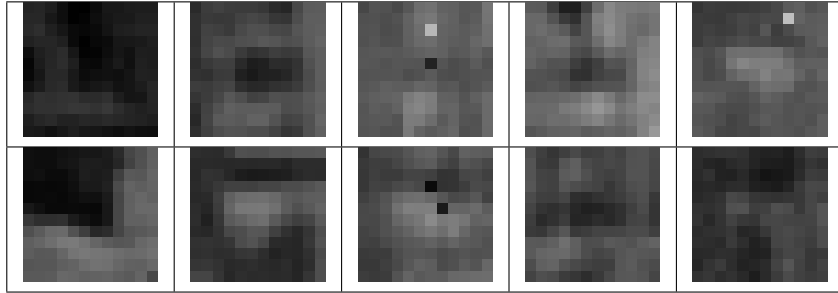
The derivative of constraint (6.5) is

$$\sum_{k:\alpha_k>0} -\frac{a_k(\mathbf{c}_k + 2\epsilon)}{2\sigma^2} \exp\left(-\frac{\mathbf{c}_k^\top \epsilon + \|\epsilon\|^2}{2\sigma^2}\right). \quad (6.6)$$

This minimization problem under constraints is solved using an interior-point method [7, 8], resulting in the desired  $\epsilon$  to be applied. Once  $\epsilon$  is found for a particular  $\mathbf{r}$ , we reshape it to  $12 \times 12$ . After that, it is resized to size of corresponding support region by bicubic interpolation method, rescaled to  $[0, 255]$ , and finally added to the corresponding support region. Table 6.2 shows visual of patches  $\epsilon$ .

### 6.3 Evaluation of the orientation attack

We first evaluate the effectiveness of the above method for changing the orientation of keypoints. We show that while the orientation of many keypoints indeed

Table 6.2: Visual examples of patches  $\epsilon$ 

change, some remain un-impacted. We evaluate the method at the image level and show that new keypoints appear as a side-effect of visual distortions. We finally benchmark the effectiveness of the method when querying the database of 100,000 random images (see section 4.1) with orientation-attacked quasi-copies.

The SVMs were trained using 1,000 random images from that set, as described earlier. Note that this amounts to 954,285 samples, and the number of samples per orientation class ranges from 19,567 to 45,060.

### 6.3.1 Ability to Change Orientations

We apply the method to the keypoints belonging to octaves  $\{-1, 0, 1\}$  of the 1,000 images earlier described. To check whether or not orientations changed, we observe for all keypoints the angle between the original and attacked orientation, expecting a change of  $\Delta\theta = \pi/2$ . However, this is not always verified. With  $\Delta d = 2$ , Figure 6.4(a) counts the number of keypoints as a function of the observed orientation change  $\Delta\theta$ . Each bin on the x-axis covers a range of  $\pi/18$  from 0 to  $\pi$ : the first bin corresponds to keypoints with  $\Delta\theta \in [0, \pi/18]$ , the second one to  $\Delta\theta \in (\pi/18, 2\pi/18]$ , ... It appears that for most of the keypoints, the orientation is changing by  $6\pi/18$  to  $8\pi/18$  ( $7^{th}$  and  $8^{th}$  bins). The value of  $\Delta d$  drives this phenomenon: a larger value for  $\Delta d$  increases the number of  $\pi/2$  changes but in turn causes more severe and visible distortions in the patches. This will be shown in the following where results using  $\Delta d = 2$  and  $\Delta d = 3$  are compared.

Figures 6.4(c) and 6.4(d) show the average of the Euclidean distances between the original and attacked descriptors, as a function of  $\Delta\theta$ . The observed distances are fairly constant between  $4\pi/18$  and  $13\pi/18$ . While Figure 6.2 suggests enforcing  $\Delta\theta = \pi/2$  would be best, Figures 6.4(c) and 6.4(d) show that, in practice, a value for  $\Delta\theta$  ranging from  $4\pi/18$  to  $13\pi/18$  pushes the attacked descriptors far away in the feature space.

For  $\Delta d = 2$ , 79% of the keypoints have their orientation changed from  $4\pi/18$  to  $13\pi/18$ , moving as far as possible descriptors in the feature space, and making matches potentially problematic. When  $\Delta d = 3$ , this number is even higher:

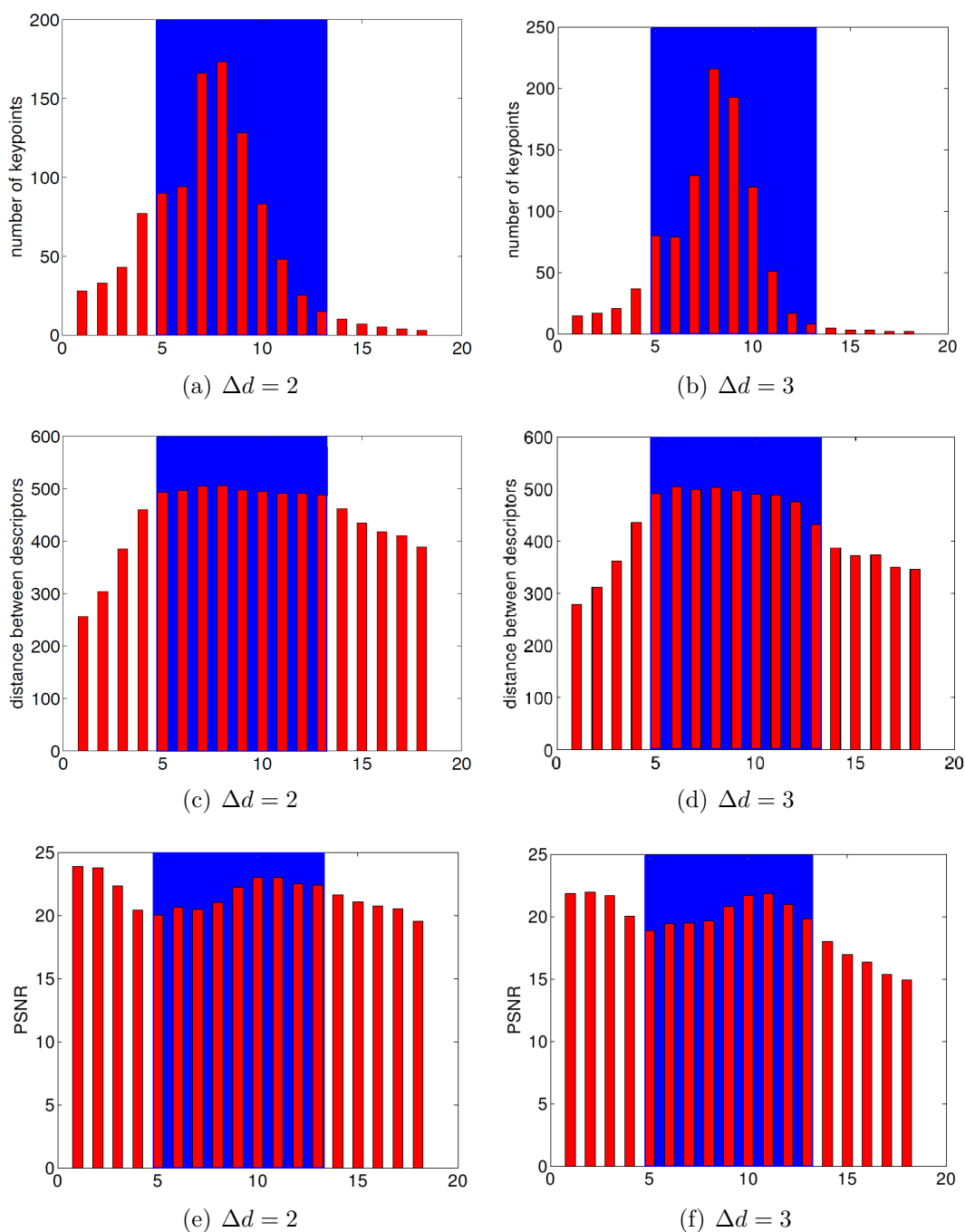


Figure 6.4: Orientation changes analysis for  $\Delta d = 2$  and  $\Delta d = 3$ . X-axis: each bin  $i = 1, \dots, 18$  corresponds to an observed orientation change  $\Delta\theta = [\frac{(i-1)\pi}{18}, \frac{i\pi}{18}]$ . (a)-(b): Number of keypoints per orientation change bin; (c)-(d): Distance between the original and modified descriptors; (e)-(f): PSNR between the original and modified support regions.  $\Delta\theta \in [\frac{4\pi}{18}, \frac{13\pi}{18}]$  is marked as blue region.

about 89%, as summarized in Table 6.3.

However, the increase of  $\Delta d$  introduces more distortion. The average PSNR between the original and attacked support regions (computed over the 79% of keypoints having their orientations changed from  $4\pi/18$  to  $13\pi/18$ ) is 21.64 dB for  $\Delta d = 2$ . For  $\Delta d = 3$ , it drops to 19.44 dB for the same set of keypoints. This phenomenon is also visible on Figures 6.4(e) and 6.4(f) representing the PSNR between original and attacked support region.

To keep a acceptable PSNR for attacked patches,  $\Delta d$  is set to 2 in the following experiments of this chapter.

	%keypoint	PSNR
$\Delta d = 2$	79	21.64
$\Delta d = 3$	89	19.44

Table 6.3: Number of keypoints whose orientation has changed from  $4\pi/18$  to  $13\pi/18$  and average PSNR of patches for different value of  $\Delta d$

### 6.3.2 Cropped Patches Center

It is possible to preserve even more the PSNR by running a small variant of the method. Instead of applying the patch to the whole support region, only its central region is added. The size of the latter is proportional to the size of the support region, meaning it is equal to  $7 \times 7$ ,  $9 \times 9$ , or  $11 \times 11$  for keypoints belonging to octave -1, 0, or 1, respectively. These sizes are chosen from empirical experiments, trying to get the best trade-off between the number of orientation changes from  $4\pi/18$  to  $13\pi/18$  and the PSNR of the attacked patches.

Although we only update the center of patches, it is quite effective thanks to the weighting enforced when determining the orientation, the central area of the support regions having more influence. Reproducing all experiments with this variant gives an average PSNR of 23.84dB as shown in Figure 6.5(f). It does, however, change the effectiveness of the method as fewer keypoints have their orientation changed. Figure 6.5(b) counts the number of keypoints as a function of  $\Delta\theta$  observed with this variant. It clearly shows many orientations could not be changed (see the left-most bin); most of the keypoints that changed orientation have a  $\Delta\theta \in [4\pi/18, 10\pi/18]$ . The distances between descriptors are shown in figure 6.5(d), which is quite similar to the ones observed on Figure 6.5(c).

There are about 62% number of keypoints having their orientations changed from  $4\pi/18$  and  $13\pi/18$ . Table 6.4 summarize results when comparing the full and cropped patches updating for  $\Delta d = 2$ .

Because the number of modified keypoints having  $\Delta\theta \in [\frac{4\pi}{18}, \frac{13\pi}{18}]$  is still high and while increasing the overall PSNR, this variant is used in the sequel.

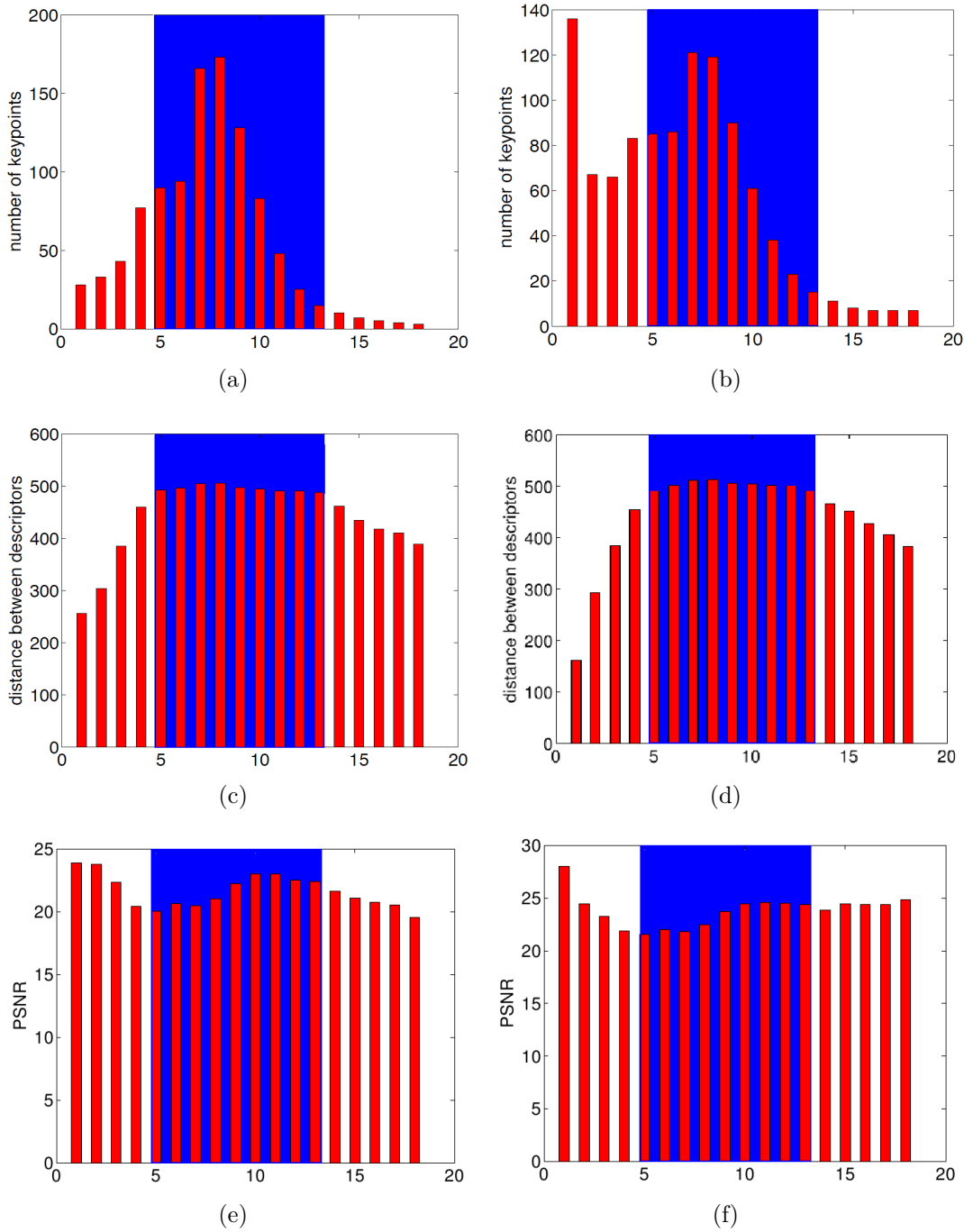


Figure 6.5: Orientation changes comparison for  $\Delta d = 2$  when applying a full patch (left column) and a centered cropped patch (right column). X-axis: each bin  $i = 1, \dots, 18$  corresponds to an observed orientation change  $\Delta\theta = [\frac{(i-1)\pi}{18}, \frac{i\pi}{18}[$ . (a)-(b): Number of keypoints per orientation change bin; (c)-(d): Distance between the original and modified descriptors; (e)-(f): PSNR between the original and modified support regions.  $\Delta\theta \in [\frac{4\pi}{18}, \frac{13\pi}{18}]$  is marked as blue region.

	%keypoint	PSNR
Full patches	79	21.64
Cropped patches	62	23.84

Table 6.4: Number of keypoints whose orientation has changed from  $4\pi/18$  to  $13\pi/18$  and average PSNR of patches for  $\Delta d = 2$ .

### 6.3.3 Impact at Image-Level

To get an acceptable visual distortion at the level of the whole attacked image, the variant modifying the center of support regions is used and applied only if the PSNR between the original and patched support regions is bigger than a given threshold  $t_{PSNR}$ . We apply the method to 1,000 images for 3 different  $t_{PSNR}$  values: 15.3, 16.3, and 17dB. The Table 6.5 shows the average PSNR computed at image level over the 1,000 images for the different  $t_{PSNR}$  values. As expected, the PSNR increases with  $t_{PSNR}$ , as fewer keypoints are modified.

$t_{PSNR}$	15.3	16.3	17
Avg PSNR on 1,000 images	28.39	29.24	29.93

Table 6.5: Average PSNR vs  $t_{PSNR}$

Having a closer look on how the keypoints are modified by the orientation attack shows that keypoints can be divided into three categories:

1. unchanged keypoints (slight change (or unchange) in location, scale and orientation)
2. keypoints having their orientation changed (slight change (or unchange) in location and scale, but significantly change orientation)
3. new keypoints created as a side-effect of the distortions introduced by the attack.

In detail, we evaluate how many keypoints fall into each class as follows. Let  $kp_o = \{x_o, y_o, \sigma_o, \theta_o\}$  be an original keypoint,  $(x, y, \sigma, \theta)$  a keypoint in attacked image, and  $d(., .)$  the Euclidean distance.

- A keypoint falls into the first category if there is a  $kp_o$  such that  $d((x, y), (x_o, y_o)) \leq 5$  and  $0.7 \leq \sigma/\sigma_o \leq 1.3$  and  $|\theta - \theta_o| \leq \pi/18$ . These values have been determined because any keypoint in that class remains pretty close to its original keypoint in the feature space (at a distance lower than 200, see Figure 6.5(d)), allowing easy matching.

Table 6.6: Number of keypoints for each category in attacked images over all octaves; PSNR between original and attacked images. In 5<sup>th</sup> column, some keypoints are unchanged only because they belong to higher scales, and then are not attacked.

Image	# KP before attack	$t_{PSNR}$	# KP after attack	# KP cat.1	# KP cat. 2	# KP cat. 3	PSNR
LENA	1218	15.3	1388	258 (18.6 %)	712 (51.3 %)	418 (30.1 %)	27.88
		16.3	1360	287 (21.1 %)	688 (50.6 %)	385 (28.3 %)	28.57
		17	1321	321(24.3 %)	668 (50.5 %)	332 (25.2 %)	29.14
AVG 1000 IM	1026	15.3	1224	229 (18.7 %)	653 (53.3 %)	342 (28 %)	28.39
		16.3	1198	266 (22.2 %)	623 (52 %)	309 (25.8 %)	29.24
		17	1177	290 (24.7 %)	604 (51.3 %)	283 (24 %)	29.93

- A keypoint falls into the second category if there is a  $kp_o$  such that  $d((x, y), (x_o, y_o)) \leq 5$  and  $0.7 \leq \sigma/\sigma_o \leq 1.3$  and  $|\theta - \theta_o| \geq \pi/18$ .
- The remaining keypoints fall into the third category. They can be seen as new keypoints as they are far in position or scale with respect to the original keypoints.

Table 6.6 shows the number of keypoints belonging to each category. It also shows the PSNR between original and attacked images. The value of  $t_{PSNR}$  has not a big impact on the number of keypoints having their orientation changed: whatever the value  $t_{PSNR}$ , almost half of keypoints belongs to the second category. However, as mentioned previously, the higher  $t_{PSNR}$ , the more unchanged keypoints, and the better the global PSNR. Corollary, the number of new keypoints decreases when  $t_{PSNR}$  increases. As presented, the attack is applied to keypoints belonging to octaves  $\{-1, 0, 1\}$  so some keypoints are unchanged because they belong to higher scales.

Figure 6.6 illustrates the unchanged keypoints, keypoint changed in orientation and nearest original keypoints (in same scale) to keypoints changed in orientation on Lena image. We can see that keypoints changed in orientation (green) are located near to original keypoints (yellow) but have a significantly different orientation.

### 6.3.4 Impact on Large-Scale Recognition

This section evaluates the proposed attacks on the system given in section 4.1. The search process uses multiple voting scheme of NVTree. The geometric verification is not considered in this experiment. The same 1,000 images are used as queries and we ran the proposed orientation attack on them resulting in quasi-copies. The variant modifying the center of support regions is used and controlled by the  $t_{PSNR}$  threshold.



Figure 6.6: Illustration for orientation attack of keypoints belonging to octaves  $\{-1, 0, 1\}$  with  $t_{PSNR} = 17$ : unchanged keypoints (blue), keypoints changed in orientation (green), nearest original keypoints (in same scale) to keypoints changed in orientation (yellow).

Each query probes the system which returns the top 100 images with associated scores. We then compute the average score of the original image.

Figure 6.7 shows average score of original image (red line) and the three top matching images for some different  $t_{PSNR}$  values. From right to left, the gap between original image and best competitor scores decreases, as the strength of the attack increases. The attack succeeds for  $t_{PSNR} = 15.3$  dB. Even if the attacked image is not completely concealed, the original image has not the best score anymore, and gets hidden behind another image that better matches.

## 6.4 Combination of Attack

In chapter 5, we presented some strategies to delude the recognition of CBIRS system, focusing on attacks at keypoint detection step. It includes keypoint removal methods as GS, LS and forging new keypoints method as FMD. This chapter proposed an attack at keypoint descriptor computation step that tries to change orientation of keypoints. Hence, it is likely to change keypoint descriptor. To craft stronger attacks, these single attacks should be combined together. We first



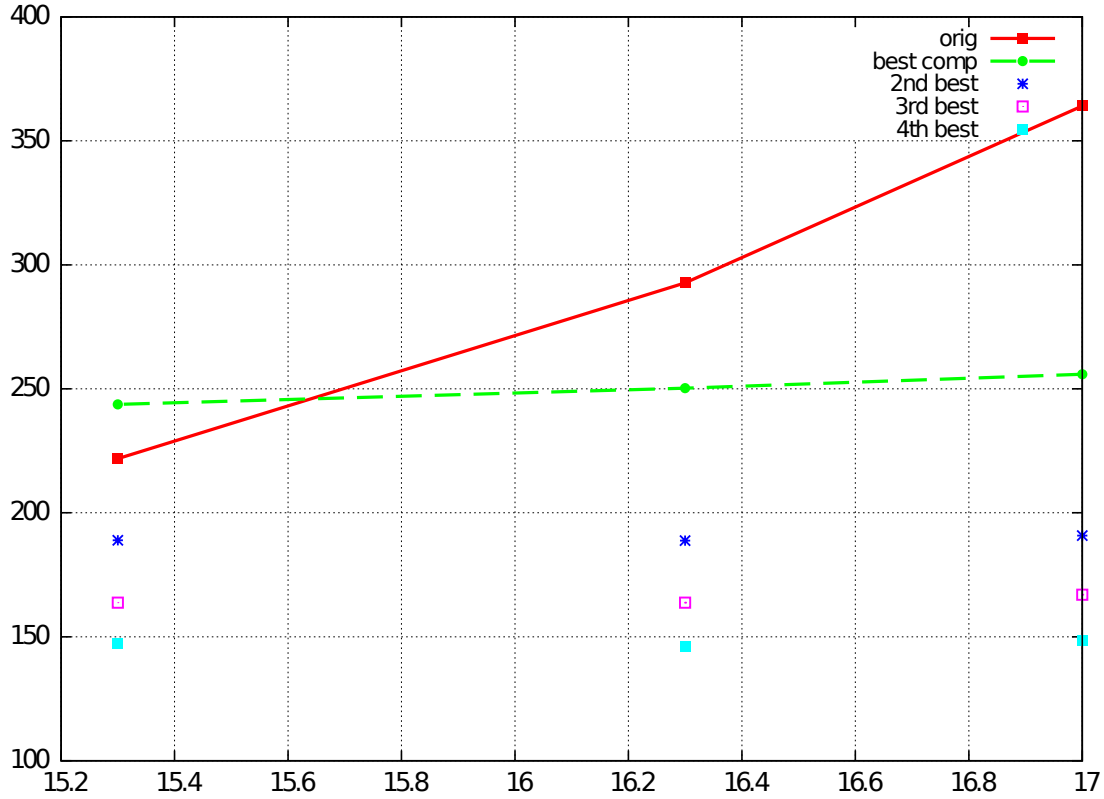


Figure 6.7: X-axis: orientation attacks with  $t_{PSNR} \in \{15.3, 16.3, 17\}$  in dB. Y-axis: the average scores over the 1,000 queries of the original images and of the three other best matching images.

detail the way we combine them. We then evaluate the attack combination on the system using both multiple and single voting. We also evaluate this combination when geometric verification is included in the system.

### 6.4.1 Global System

We combine the attacks DS, LS and FMD proposed in chapter 5 and the orientation attack. To keep an acceptable visual of the forged image, the attacks are applied to keypoints belonging to octaves  $\{-1, 0, 1\}$ .

At first, the DS attack is applied on the image, because it removes most of the keypoints, while minimizing the distortion as seen in section 5.1.2. Keypoints unchanged by the DS attack are then subjected to an orientation attack. The orientation attack is less time consuming than LS attack<sup>2</sup>. Hence, the orientation

<sup>2</sup>Reminder: The LS attack replaces the  $n \times n$  region around the current keypoint by its

attack is applied before the LS attack. For every keypoint, the patch  $\epsilon$  is applied to its support region if this changes its orientation by at least  $\pi/6$ . This, in turn, ensures that the descriptors computed from this tweaked support region and from the original support region are far enough in the high-dimensional space.

The keypoints unchanged by orientation attack are then subjected to a LS attack. Finally, once the image has been washed from as many keypoints as possible, the FMD attack is performed to forge new keypoints.

The visual impact of these different attacks is shown in Figure 6.8. We can see that the DS attack blurs regions (Lena’s hair) in the image. The orientation and LS attack have the same effect. They produce some very small square noise regions. The new keypoints by FMD are resulted in small salt and pepper noise artifacts.

For each single attack, the keypoints are modified iff the PSNR computed between the original and modified support regions does not drop below a threshold, called  $t_{PSNR}$ . A different  $t_{PSNR}$  is used for each attack, given in Table 6.7. These values of  $t_{PSNR}$  were set empirically by conducting many experiments. They yield a good trade-off between visual and score (or number of unchanged keypoints) of attacked images. In the previous section,  $t_{PSNR}$  equals to 15.3, 16.3, or 17 for the orientation attack. However, in the combination of attacks, we can set a higher threshold  $t_{PSNR}$ . The keypoints not impacted by orientation attack are likely removed by the LS attack after. Hence we can get same results<sup>3</sup> as in section 6.3.4 even for a higher  $t_{PSNR}$ .

Attack	orientation attack	LS	FMD
$t_{PSNR}$	19	19	21

Table 6.7:  $t_{PSNR}$  in dB for the single attacks of the combination.

Table 6.8 gives the average number of detected keypoints after each single attack is applied sequentially on 1,000 images. It also gives the average PSNR between attacked and original images. We can see that the DS attack removes about 11.3% of keypoints. However, after applying the orientation attack, the number of keypoints slightly increases because some new keypoints are created. After applying the LS attack, the number of keypoints is 694. It equals 67.6% of keypoints in the original image. Because the FMD creates some new keypoints, the number of keypoints increases again, resulting in 896 keypoints in the final attacked image.

---

smoothed version and checks whether this keypoint is still detected or not. This is performed iteratively using regions of growing sizes, until the keypoint is no longer detected. In orientation attack, the equation 6.2 is fastly solved by interior-point method

<sup>3</sup>The original image is not on the top of result list



Figure 6.8: Visual rendering along the successive attacks.

Attack	# keypoints	PSNR
Original image	1026	$\infty$
DS	910	34.07
+Orientation attack	961	31.77
+LS	694	29.99
+FMD	896	29.60

Table 6.8: Number of keypoints on average over 1,000 images. Attacks are applied in sequel.

### 6.4.2 Experiments With Multiple Voting and Single Voting

With the same setup as described in section 4.1, we now evaluate the improvement of attacks combination against the results obtained when the attacks are considered independently (the geometric verification is still not used for the moment).

Table 6.9 shows the average score of the original image and the three top matching images when using multiple or single voting, respectively. When multiple voting is on, the combination of attacks is more efficient than both the orientation and the GS+LS+FMD attack: The original image is ranked below the 3 best competitor images, while with the latter attack, the original image is ranked 2nd in the result list (see Figure 5.9 and Figure 6.7).

Furthermore, the average PSNR of attacks combination is 29.60 dB, while with GS+LS+FMD and the orientation attack, it is 28.23 dB (see Table 5.1) and 28.39 dB (see Table 6.6), respectively. However, when single voting is on, we can see in Table 6.9 that the original images are still on top of the result list for the same reasons as those discussed in Sect. 5.4.3: each best competitor image in the database just gets a maximum of one vote when each keypoint created by FMD is queried. In general, the systems using multiple voting is easier to circumvent than the systems with single voting.

	original image	1 <sup>st</sup> competitor	2 <sup>nd</sup> competitor	3 <sup>rd</sup> competitor
Multiple voting	83	214	143	124
Single voting	75	43	39	36

Table 6.9: Scores of the original images and of the three other best matching images, averaged over the 1,000 queries, when attacks are combined. The geometric verification is not considered.

### 6.4.3 Geometric Verification

In this section, we evaluate the combination of attacks when a geometric verification is used as a post-filtering to rerank images in the result list. The description of the geometric verification and the reranking are described in section 4.1.4.

Table 6.10 shows the average score of the original image and the three top matching images for multiple voting and single voting, respectively. When geometric verification is used, the original image is still at the top of the result list for both multiple voting and single voting. There is a very big gap between the scores of the original and the best competitor images. The main reason is that the new keypoints created as side effect by removal method GS, LS or orientation

attack or deliberately by FMD do not mimic the geometrical pattern present in the best competitor images. Therefore, these competitor images are pruned out. In the other hand, although many keypoints in the attacked image are removed, the unchanged keypoints are still geometrically consistent with the original image and they are enough of them to vote for the original image. Hence, only the original image stays on the top of the list.

	original image	1 <sup>st</sup> competitor	2 <sup>nd</sup> competitor	3 <sup>rd</sup> competitor
Multiple voting	52	5	4	3
Single voting	52	5	3	2

Table 6.10: The average scores over the 1,000 queries of the original and of the three other best matching images. The geometric verification is considered.

## 6.5 Summary

In this chapter, we proposed a new angle of attack of CBIRS based on SIFT descriptors. The proposed attack focuses on the influence of the orientation disturbance on the recognition of an image. The orientation shift in descriptor computation is accomplished by introducing locally non-affine modifications, through the addition of patches that are learned by an SVMs process. The main results of proposed orientation attack has been published in [17]. We also evaluate the combination of this attack and single attacks presented in chapter 5.

There are some lessons learned from this work.

- The results in Figure 6.7 shows that the orientation attack lowers enough the score of the original image so that it is no longer returned at first position by the system. To be truly effective, this attack must be combined with other attacks presented in chapter 5.
- Based on results in section 6.4.2, attacking the system using multiple voting is easier than attacking system using single voting.
- As presented in section 6.4.2, the single voting improves the security of the system even if it is attacked by a combination of many attacks. **By this conclusion, in next chapter, we will focus on attack system using single voting scheme.**
- When geometric verification is included in the system as a post-filtering, the attacks combination can not delude the recognition of the system anymore. This is because the new keypoints created by FMD are not geometrically

consistent with those of the original image. Hence, FMD is not used in the remaining of the thesis.

- **In the next chapter, we introduce a strategy to attack the system when geometric verification is included.** The attack is similar to Picture in Picture modification. Some small pictures are inserted into the attacked images. These patches are likely to share a geometric consistency with patterns present in some images of the database. Hence, these images are likely to have a higher rank than original image.



# Chapter 7

## Attacking geometry

In the previous chapter, when geometric verification is included in the system as a post-filtering, the proposed attacks can not delude the recognition of the system. The main reason is that the new keypoints in the attacked image do not have a geometric consistency with the lure image. To circumvent the geometric verification, the attacked image should contain numerous keypoints that can be matched in a geometric consistent way with another image of the database than the original one.

A straightforward way to proceed is to insert in the attacked image distractors such that those distractors must contain a number of matches (with some images in database) greater than the number of matches retained by the geometric verification for the original image. This attack is a kind of Picture in Picture (PiP) attack. A PiP visual modification tries to preclude the recognition of a copyrighted picture by modifying that picture by two ways defined in TRECVID<sup>1</sup> [84] as follows. The first one (referred to as PiP-Type 1) embeds the severely downscaled version of the protected image in a distractor content, which dominates recognition. In the second one (referred to as PiP-Type2), a small distractor image is embedded in the protected image. In this chapter, we propose a PiP attack similar to PiP-Type 2. However, in PiP-Type 2, the distractor image is chosen randomly, while in the proposed attack, the small distractor is carefully determined such that it will catch almost all matches avoiding the full resolution image to be identified, hence deluding recognition.

This chapter is structured as follows. Section 7.1 briefly describes how PiP visual modification schemes are created and how existing CBIRS recognize contents despite PiP. Section 7.2 describes how these security-oriented PiP visual modifications are produced. Section 7.3 shows the effectiveness of the proposed scheme through large scale image recognition experiments. Finally, Section 7.4

---

<sup>1</sup>TRECVID defines PiP attack on video but PiP problems in video raises very similarly to image. The PiP term we use here refers to both video and image.



concludes.

## 7.1 PiP Visual Modifications

This section gives a brief overview of the typical process that eventually creates a Picture in Picture visual modification. It then discusses about the two families of solutions found in the literature recognizing contents despite PiP attacks.

### 7.1.1 Producing PiP Visual Modifications

The way TRECVID produces PiP modifications is quite generic and works as follows [84, 70]. In PiP-Type 1, first, an image (or video) is randomly picked from a collection of distracting contents, in practice unlikely to match with the database of protected material. Then, the image (or the video) that the modification process tries to hide from recognition is scaled down and is inserted in the distracting contents. TRECVID specifies the rules driving the downscaling as well as the location of the inserted contents. The downscaled size ranges from 0.3 to 0.5 of its original size; the downscaled protected content can be inserted at five locations, the four corners or the center of the distracting image. Furthermore, the aspect ratio of the inserted image/video is typically fixed. In PiP-Type 2, a small distractor image (or video) is inserted in front of the protected image. The size of distractor ranges from 0.2 to 0.5 times the size of the protected image. Again, it can be inserted at five location as for PiP-Type 1. TRECVID benchmark includes several hundreds of such PiP material. Figure 7.1 shows two examples from TRECVID for PiP-Type 1 and PiP-Type 2 modifications.

It is interesting to compute the PSNR between the original and modified versions. PSNR is naturally very small as pixels are very different. For the examples given in Figure 7.1,  $\text{PSNR}(7.1(a),7.1(b))=12.67$  dB,  $\text{PSNR}(7.1(c),7.1(d))=15.72$  dB

### 7.1.2 Recognizing Contents Despite PiP

Overall, there are two families of techniques trying to identify the protected contents. The first family heavily relies on some prior knowledge to facilitate identification, because it knows the PiP procedure sticks to the predefined rules. The second family has no particular ad-hoc mechanism but it rather finely analyzes the candidate list of similar material returned by the database search during a post-processing step. We detail these two families below.



Figure 7.1: Two examples from TRECVID for PiP-Type 1 and PiP-Type 2: (a) and (c) are the images to be protected, kept inside the database; (b) is TRECVID PiP-Type 1 visual modifications; (d) is TRECVID PiP-Type 2 visual modifications

### 7.1.2.1 Prior Knowledge for Separating Images

The first family makes use of the PiP construction rules to separate the two images, isolating the distracting image from the one possibly protected. Once separated, the protected image is typically up-scaled and then used to query the CBIRS.

Some image separation techniques use edge/line information to detect the boundaries of the inserted image. The detection typically relies on a Canny edge detector or on a Hough transform [69, 53, 78]. In [53], to detect PiP in video, they first segment the attacked video into shots, where each shot contains homogeneous content. The first frame within each shot is selected as a keyframe. All next processes are applied on keyframes<sup>2</sup>. A Canny edge detector is run to locate horizontal and vertical edges which can possibly be at the boundaries of the

<sup>2</sup>At this time, PiP detection in video is turned to PiP detection in image

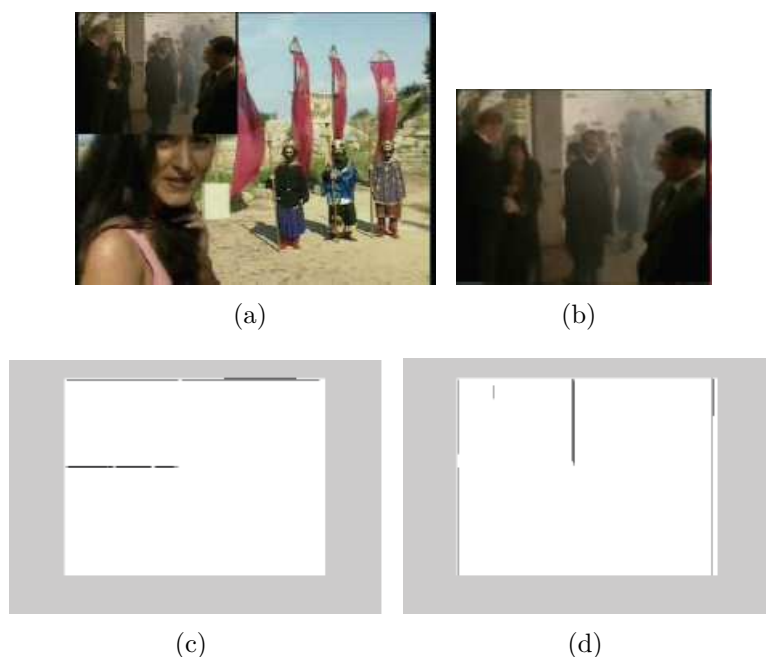


Figure 7.2: PiP detection: (a): protected video is embedded in a distractor video; (b): enlarged version of protected video; (c): horizontal edge image; (d) vertical edge image. Figures are from [53].

inserted image. Candidate regions which will later be used as queries are obtained by grouping four edges in pairs made of two horizontal and two vertical edges. Some candidate regions are then eliminated according to some additional criteria such as their aspect ratio as well as their location. This process is illustrated in Figure 7.2 [53]. Finally, the remaining candidate regions are extracted from the global image, up-scaled and they probe the system. The one with the highest similarity score is returned as the final answer. [69] uses a Hough transform on keyframes to detect persistent strong horizontal lines from which the candidate regions are determined.

In [78], both Canny edge detector and Hough transform are used for PiP detection. In their work, instead of applying Canny edge detector on keyframes, the detector is directly run on every single frame of the video, which in turns creates a series of detected edge images. A time-sliding window processes consecutive edge images and produces average edge images. A Hough transform is then applied on these average edge images to detect persistent lines, and short lines as well as non horizontal or vertical lines are eliminated. The remaining lines create candidate regions, probing the system one after the other.

Some other methods detect corners of regions instead of lines. In [10, 4], a Sobel operator and a Laplacian kernel are applied to video frames. Pixels in

frames that are found to be part of either horizontal or vertical edges are flagged. The corners of candidate regions are thus the pixels in images where the maximum accumulation of edge points on horizontal and vertical directions are found.

Other methods make an even more rough use of the prior knowledge of the rules driving the creation of PiP visual modifications. For example, [50] extracts from the whole image a series of candidate regions located at the four corners as well as around the center, having sizes of 30%, 40% and 50% of the whole image. These regions are then used to probe the database.

Overall, these methods too much rely on the prior knowledge to be usable in practice from a security perspective. While these PiP detection techniques are somehow robust, they are not at all secure. Once the TRECVID rules are known, it is easy to create a PiP visual modification that diverges enough from these rules to make recognition more problematic. Inserting the protected content at a random location, rotating it by few degrees or stretching it by different scaling factors are sufficient to delude recognition.

### 7.1.2.2 Identification Without Prior Knowledge

In contrast to the approaches mentioned above, some other techniques do not separate the images. They rather compute feature vectors over the whole image and then query the database with every single descriptor, without separating anything [35, 26, 66, 89]. Once all the query descriptors have probed the database, the list of candidate images is then post-processed in order to identify the potential protected contents. This post-processing step typically checks the geometry consistency between the query and all images in the candidate list. Very robust tools estimating the geometrical consistency are used, such as RANSAC [24, 52, 72] or the Generalized Hough Transform [3, 56].

Three comments are in order, however. First, the feature vectors (typically SIFT) are so powerful that the search process is likely to put in the candidate list the protected image corresponding to the one quasi-copied in the PiP, despite its downscaling. Finding that image inside the list of candidates is thus a matter of eliminating the other candidate images that are false positives. Second, the robustness of the geometrical verification process is so high that if the query indeed contains some pieces of protected contents, then it will be classified as containing inliers (mostly, if not only). Third, because the distracting image in the query does not match with any image in the database, then no consistency will be found there and the distracting part of the query image is thus classified as containing outliers only.

These techniques strongly assume that the distracting part of the image/video query does not belong to database—this is a very serious drawback. Otherwise, when the distracting part is in (or turns out to be in) the database, then the

result returned by the CBIRS is much more ambiguous.

## 7.2 CBIRS-aware PiP Creation

The proposed PiP attack has the same philosophy as PiP-Type 2. However, in contrast to the generic way of creating PiP visual modifications, we now detail a technique producing PiP attacks deluding the recognition capabilities the system.

The main ideas of the proposed attack are the following ones:

- To make the distractor part dominates the recognition,
- To make it difficult to separate distractor and protected content.

The first idea includes two steps. The first step is to “wash” the image. The goal is to reduce the number of matches between the forged and the original image and in order to make the protected content more difficult to be recognized. The second step is to insert some specific distractors into the washed image. The goal is to make the distractor strongly recognized and so that the protected content is badly ranked. The details of two steps are presented in the next sections.

### 7.2.1 Image Washing Before PiP

Because there are approximately 1,000 keypoints in a protected image, a forged copy of a protected image should be “washed” before doing PiP attack to reduce as much as possible the number of matches between the forgery and its original version. We apply the combination of DS, orientation attack and LS attack in the sequel as presented in the previous chapter (see Section 6.4.1) to reduce the number of matches satisfying Lowe’s criterion (see section 4.1.4) between the attacked image and its original version. The number of keypoints in the washed image, the number of matches between and the PSNR of the washed images are given in Table 7.1.

Attack	# keypoints	# matches	PSNR
Original image	1026	1026	$\infty$
DS	910	833	34.07
+Orientation attack	961	505	31.77
+LS	694	115	29.99

Table 7.1: Number of keypoints and number of matches between the washed images and their original version. Average over 1,000 images. Attacks are applied in sequel.

As mentioned in the previous chapter, the FMD attack is useless when geometric verification is included in the system. Hence, it is not considered in this chapter. In Table 7.1, after the washing, the number of matches between the forged and the original image is 115. It means that only 11.2% of the matches remain. The number of points compliant with the geometric verification (section 4.1.4) is thus decreased. Overall, after the washing, the forged copy has (i) few descriptors that perfectly match with its original version, (ii) many descriptors that are unlikely to match since they have been moved away in the high-dimensional space and (iii) several new keypoints created as a side effect of visual artifacts that may or may not match. Querying the CBIRS with such a washed image is likely to identify the original but with a very low score, making the system less confident.

## 7.2.2 Creating the PiP Attack

Washing even more reduces the number of keypoints that still match and it breaks recognition. Yet, so severely washed images are much too distorted to remain usable in practice and to keep any commercial value. In contrast, as highlighted in the Section 7.1, making sure the distracting part dominates the recognition is one option; making it difficult to separate the two images is another. For these reasons, our scheme for producing CBIRS-aware PiP attacks adopts the following guidelines:

- The relative locations of the protected contents and the distracting part have to be as free as possible. Corners and centers should not be the only options.
- The frontier between the protected and the distracting contents must not boil down to straight segments (this includes not being strictly vertical or horizontal).
- The distracting part must always strongly match with some contents that is protected in the database, even if the contents of this database remain absolutely unknown. Making sure the recognition has ambiguities is key to this attack scheme.
- The PSNR computed between the original image and its forgery must be high to preserve visual quality.

From these guidelines, our PiP attack inserts inside the image to be forged a small visual distractor carefully determined such that it is extremely likely to strongly match with some of the (possible unknown) contents of the database. The sequel describes how such visual distractors are determined, how they are inserted into the images and why more than one distractor might be inserted.

### 7.2.2.1 Determining Candidate Visual Distractors

After the washing, the forged image still contains several keypoints matching with their original counterparts (eg. 115 matches in Table 7.1). To make sure this original image does not get ranked first by the CBIRS, more matches than this number must be created between another image from the database and some visual elements in the query image. Furthermore, there must be a good geometrical consistency between these matching keypoints to pass the geometrical verification.

In general, the distractors are selected such that the forged image (after distractor insertion) should have strong matches passing the geometric verification with some images (not original version of forged image) that are protected in the database. The creation of distractors can be seen under two situations.

First situation, it is easy to identify such matches when the database of protected contents is known. The contents to be protected might be copyrighted material such as blockbusters or other images that are somehow known from everyone. In another case, the CBIRS might return images that are similar to the ones used for querying the system. In this case, it is possible to patiently send queries and accumulate results.

Analyzing these disclosed images bootstraps the creation of CBIRS-aware PiP attacks. Given these images, we crop the densest keypoint regions at various sizes eg.  $10 \times 10$  pixels,  $15 \times 15$ ,  $20 \times 20$ ,  $\dots$

This, somehow, creates a dictionary of visual distractors that are sorted by the number of keypoints still matching their origin image after cropping. Note that, a  $50 \times 50$  pixels distractor containing  $m$  keypoints can have number of matches with its origin image less than  $m$ . The difference comes from the keypoints near the borders of the distractor which can not match anymore. Figure 7.3 illustrates the matching between a distractor and its original image.

Then, we select a distractor from this dictionary containing more keypoints than the number of remaining matching keypoints after the washing of the forgery. Inserting this distractor in the washed image is going to produce enough geometrically consistent matches to push the original image down in the result list (at least with rank #2, see 7.2.2.4).

Second situation, in some applications, it might be impossible to disclose what is inside the database. It is however possible to create a set of visual distractors that are extremely likely to match with some images from this unknown database. Many images include textures that are very repetitive, such as tiles, bricks, tree leaves, window shutters, windows on a facade seen from far enough, friezes, fabrics, clothes, etc. Downloading a fair number of pictures from Flickr provides an easy way to bootstrap the construction of a dictionary of keypoint-dense visual distractors, as described above. These distractors typically contain regular and

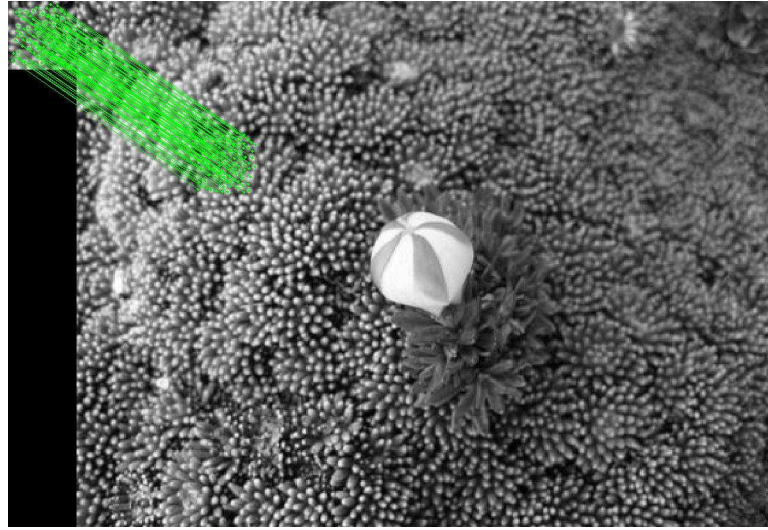


Figure 7.3: Matching between a distractor and the image it is extracted from. The number of keypoints in the distractor is 148. There are 120 matches between the distractor and its original image.

repetitive patterns. The pirate bets that these patterns will match with some of the database images even if no images from the database are known. Furthermore, it is likely that these distractors will be geometrically consistent with some of the database contents.

Overall, our strategy starts by building a dictionary of visual distractors. Examples are given in Table 7.2. The distractors have different sizes since their keypoint density varies. It then picks from the dictionary the distractors that have more keypoints than what remains in the washed image. This gives a list of candidate distractors. The next sections describe the selection of one or more distractors in that list and their insertion in the washed image.

### 7.2.2.2 Inserting a Visual Distractor

Given a distractor, we define four policies for determining where the distractor is inserted inside the washed image. In contrast to TRECVID, the insertion is not driven by a simple rule, but rather by the content of the washed image, and therefore its place significantly varies from one image to the other. For simplicity, we detail the policies assuming there is only one distractor candidate for insertion.

**Policy #1: PSNR-Oriented** The first policy finds the place in the washed image where the distractor can be inserted while reducing the PSNR as little as possible. Let  $I_w$  be the washed image into which the distractor  $I_p$  has to be





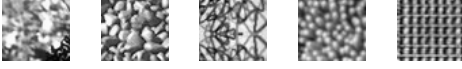



# of keypoints	Visual Examples
20–50	
50–100	
100–200	
200–300	
300–400	
400–500	

Table 7.2: Dictionary of keypoint-dense visual distractors.

inserted, eventually producing the image  $I'_w$ .  $I_p$  is a  $p \times p$  pixel square. First,  $I_p$  is centered  $\tilde{I}_p = I_p - \bar{I}_p$ , with  $\bar{I}_p$  denoting the average luminance.  $\tilde{I}_p$  is then sled over the washed image. Because SIFT is invariant to illumination changes (provided the local contrast is not too small), it is possible to adjust the illumination of  $\tilde{I}_p$  to be as close as possible to  $I_w(x, y, p)$  the  $p \times p$  region of  $I_w$  around position  $(x, y)$ . This boils down to finding parameter  $a$  and  $b$  such that:

$$\min_{a,b} \|a\tilde{I}_p + b - I_w(x, y, p)\|^2, \text{ subject to: } a \geq a_{min}. \quad (7.1)$$

The constraint  $a \geq a_{min}$  avoids small values of  $a$  which flattens  $\tilde{I}_p$  removing most (if not all) of its keypoints. Solving (7.1) gives  $a$  and  $b$ :

$$\begin{aligned} b &= \bar{I}_w(x, y, p), \\ a &= \max \left( a_{min}, \frac{\tilde{I}_p^T I_w(x, y, p)}{\|\tilde{I}_p\|^2} \right). \end{aligned} \quad (7.2)$$

Once the best illumination of  $\tilde{I}_p$  is determined at position  $(x, y)$ , the PSNR between  $a\tilde{I}_p + b$  and  $I_w(x, y, p)$  is computed. Once  $\tilde{I}_p$  has been sled all over  $I_w$ , it is inserted at the place where the maximum PSNR was observed. At that time,

after having inserted the distractor  $\tilde{I}_p$  with the appropriate values for  $a$  and  $b$ , the washed image becomes  $I'_w$ . Figure 7.4 illustrates a distractor  $I_p$  before and after illumination adjustment.

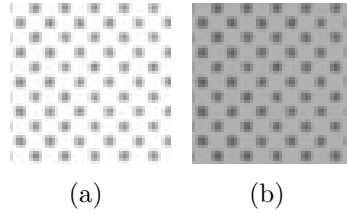


Figure 7.4: Illustration for illumination adjustment: (a) original distractor; (b) after illumination adjustment

**Policy#2: Matching Density-Oriented** The second policy determines where to insert based on the number of matches that still exist between the washed image  $I_w$  and its original, non modified version. It moves a sliding window having size equal to size of distractor over  $I_w$  to identify the region with a maximum of matches. Then,  $I_p$  is centered,  $a$  and  $b$  are determined as in Policy#1 and then  $a\tilde{I}_p + b$  replaces the appropriate region of  $I_w$ . Generating  $I'_w$  this way tends to dramatically reduce the number of matches, making recognition harder. Figure 7.5 shows the number of matches between a washed image and its original version before and after distractor insertion.

**Policy#3: Visual Attention-Oriented & PSNR** Care must be taken to move away the distractor from being inserted in the middle of, or close to, the visual centers of interest that exist in the image. This third policy therefore computes a visual saliency map for  $I_w$  using the Graph-Based Visual Saliency (GBVS) approach [29]. In [29], some saliency regions are extracted in three steps. The first step extracts some feature vectors over the image, resulting in feature images (feature maps). In their experiment, the feature maps are formed by the convolution of input image with some Gabor filters at various scale and orientations. The second step forms an “activation map” using the feature maps. Each feature map  $M$  is resized to a smaller size. Let the new size be  $n \times n$ . A fully-connected graph  $G_A$  is obtained by connecting every node of the lattice  $M$ , labelled with two indices  $(i, j) \in [n]^2$ , with all other  $n - 1$  nodes. The weight for an edge between two nodes is computed as<sup>3</sup>:

$$d((i, j), (p, q)) = \left| \log \frac{M(i, j)}{M(p, q)} \right|$$

<sup>3</sup>In their experiment, the edge weight is further multiplied by a Gaussian function



(a)



(b)

Figure 7.5: Matching density-oriented distractor insertion (Policy#2): (a) The number of matches between a washed and original image is 156; (b) The number of matches between images after distractor insertion is 144.

They run a Markov chain over the graph  $G_A$ , and regard its equilibrium distribution over the map as the saliency values. In the final step, the activation map is normalized using a Markovian algorithm. The normalized activation maps are summed to form a master saliency map. The values of the master saliency map is mapped to  $[0,1]$ . Then, the master map is interpolated to the size of the input image. In our experiments, saliency regions include pixels having saliency value higher than 0.7. Policy#3 then simply determines salient-enough regions and then applies Policy#1, forbidding the sliding window to enter inside these regions, as illustrated in Figure 7.6.

**Policy#4: Visual Attention & Matching Density** Once the salient-enough regions have been determined, then Policy#2 is applied. This inserts the distractor where the largest number of matching keypoints is found, outside the salient regions.

### 7.2.2.3 Blurring Boundaries

The systems that separate pictures to cope with PiP attacks heavily rely on horizontal and vertical segment detection. Challenging their security is possible by

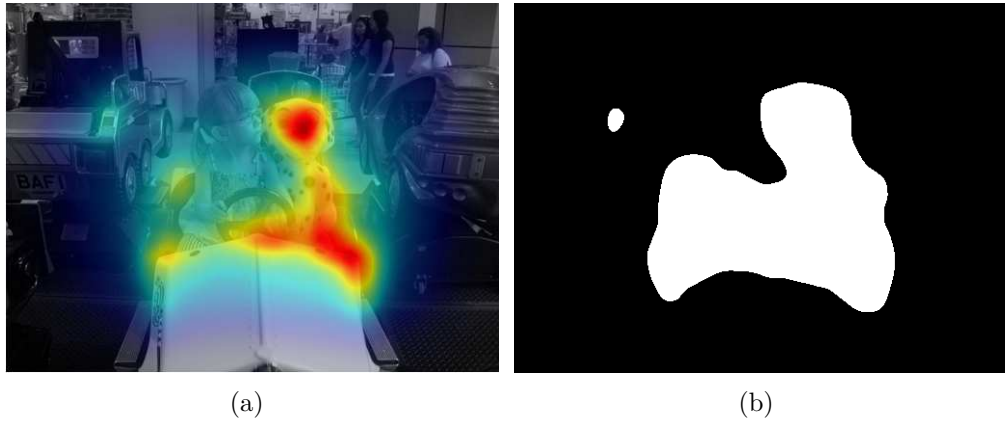


Figure 7.6: (a) shows the map of salient regions according to GBVS. (b) shows in white the area where high-saliency forbids the distractor insertion.

modifying the images such that the detection of segments becomes very difficult with their approaches. One option is to blur the boundaries separating the two images. After insertion of the distractor, a small and local Gaussian ( $\sigma = 1$ ) blurring around the frontier does the job. The blurring makes intensity of pixels on distractor boundary and their neighbors similar. Hence, vertical and horizontal edges are much harder to detect.

The Figure 7.7(a) shows a close view of distractor boundary after blurring. Figure 7.7(b) shows edge image after applying a Canny edge detector on attacked image 7.7(a). We can see that there are neither vertical, nor horizontal edges around the distractor boundaries. Hence, the methods using prior knowledge as presented in section 7.1 may fail.

#### 7.2.2.4 Inserting Multiple Visual Distractor

Inserting one distractor in the washed image pushes the corresponding original image at least at rank #2 in the result list. It might be desirable to push it further. In this case, several distractors might be inserted in the washed image. Two inserted distractors push the original image to rank#3, etc. Of course, it is not possible to push it extremely far without too severely degrading the image. Figure 7.8 shows examples where two distractors have been inserted. We can see that the position of distractors can be anywhere in the image. This is very different from PiP in TRECVID. When the distractor is a regular pattern as in Figure 7.8, for policies #1 and #3, it is usually inserted in uniform regions. For policies #2 and #4, it is inserted in nonuniform regions. This is expected because there are many keypoints at these latter regions. As expected, the PSNR is “preserved” with policies #1 and #3 (higher than 29). We also see the disadvantage

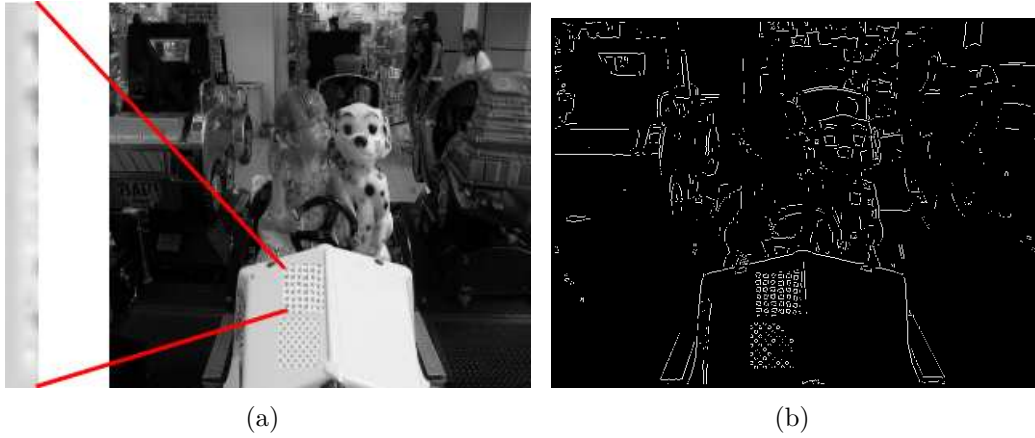


Figure 7.7: Blurring the boundaries: (a) Close view of a distractor boundary ; (b) Edge image after applying the Canny edge detector on the attacked image (a)

of policies not including visual attention in Figure 7.8(b) where the distractor is put in a saliency region (face of dog) of image.

## 7.3 Large-Scale Experiments

This section evaluates the proposed PiP attacks on the system given in section 4.1. The search process use single voting scheme of NVTree, as it is the more difficult to attack than multiple voting scheme (see section 6.5). After that the NV-Tree returns a candidate list of the best 100 matching image, the list is processed in order to re-rank the candidates by using geometric verification, as presented in section 4.1.4.

### 7.3.1 Experiment 1: Inserting One Distractor

In this first experiment, we first wash the 1,000 query images and then determine which distractor should be inserted in each washed image. We then insert distractors in images according to the four distractor-insertion policies defined above, eventually ending-up with 4,000 CBIRS-aware PiP attacked images. Each attacked image is used to query the database. We first discuss the effectiveness of the attacks before giving details on the distractors used to produce these attacks.

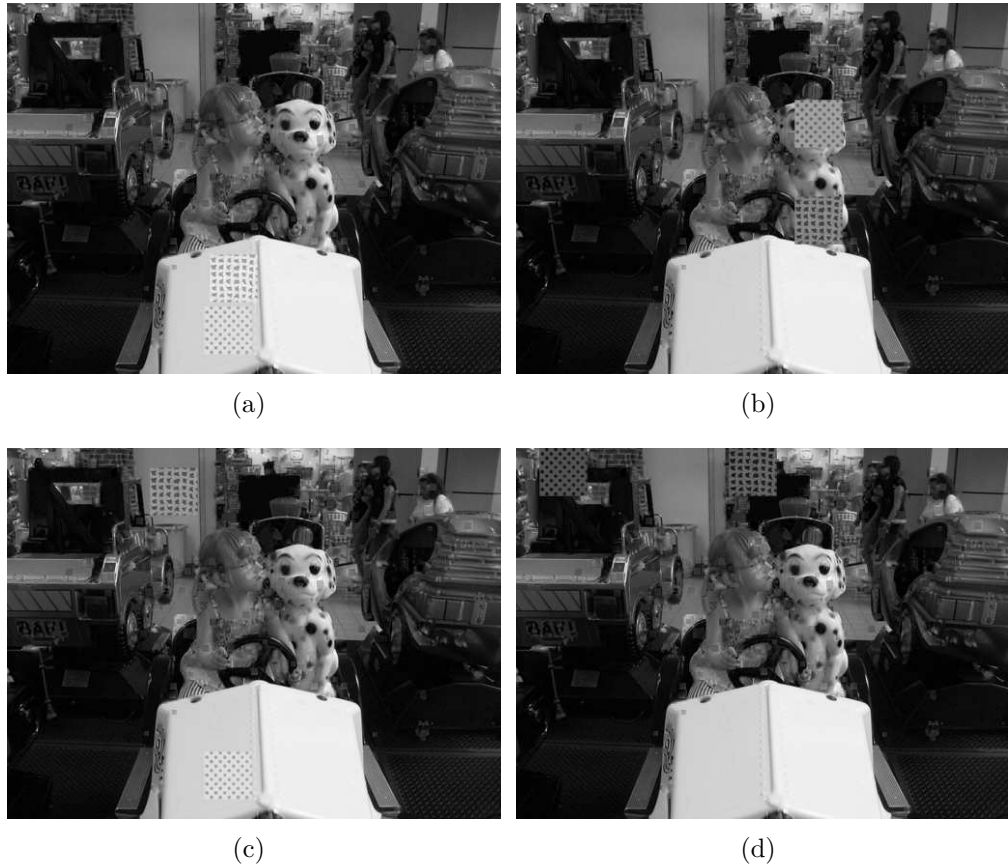


Figure 7.8: The four distractor insertion policies for 2 distractors. (a), (b), (c) and (d) are Policies #1 to #4, respectively. PSNR are respectively 29.13, 24.77, 29.11, and 25.14 dB.

### 7.3.1.1 Effectiveness of the Attacks

Table 7.3 shows the effectiveness of the attack for each of the four policies as given by the first column. The second column gives the average PSNR between the original images and their attacked version. The third column gives the number of times the PiP attack was not successful since the washed and attacked images could still be identified by the system and be ranked first.

It turns out that some images can not be attacked in practice while preserving their PSNR. They are too much textured, and still have too many unchanged keypoints after washing. It would require the insertion of big visual distractors to topple over recognition. An example of such an image is given in figure 7.9.

Note that the failure rate of this attack is very small  $((25+17+20+13)/4000 = 1.88\%$ , roughly, across policies). As expected, policies #2 and #4 have a light

Policies	PSNR	rank=1	rank=2	rank>100
#1-PSNR	29.55	25	923	47
#2-Matching Dens.	28.14	17	921	60
#3-Visual Att. & PSNR	29.52	20	928	46
#4-Visual Att.& Matching Dens.	28.35	13	918	63

Table 7.3: The ranks of the identified images over 1,000 queries with one distractor inserted.



Figure 7.9: This attacked image is still at on the top of the result list after reranking

failure rate (lower than the one of policies #1 and #3), because these policies are designed to remove more matches between washed image and original image. On the other hand, policies #1 and #3 have an average PSNR higher than the one of policies #2 and #4.

The fourth column of the Table gives the number of times the original image is found at rank #2 in the final result. In this case, the PiP attack is successful. The washing was strong enough to dramatically reduce the number of (good) matches and the picked visual distractor triggered enough matches (geometrically consistent) to dominate recognition.

The fifth column gives the number of times the original image is not found in the top 100 most similar images. There are  $(47+60+46+63)/4000 = 5.4\%$  of the original images having rank  $\geq 100$ . This happens when the washing removes enough keypoints and/or triggers matches with other random images from the database in addition to what produces the distractors. In other words, the washing process leaves too few keypoints identical to the original image. On average, there are 4.75 original images ranked between 3 and 100.

Overall, we observe that identifying the original image after the CBIRS-aware PiP attacks is always problematic, the system having to deal with recognition ambiguities or being unable to identify the correct images. This is a very encouraging result.

### 7.3.1.2 Details on the distractors

In addition of keeping track of image similarities to determine the effectiveness of the attacks, we also record some detailed information on the distractors used to create the PiP visual modifications. Figure 7.10(a) shows the number of times distractors of size given by the x-axis were used to produce the 1,000 attacks in the case of Policy#1 (PSNR-oriented), quantized every 5 pixels. Figure 7.10(b) displays the impact on the PSNR of such distractors. Overall, this shows that most distractors are smaller than  $50 \times 50$  pixels, which is to contrast with images that are typically  $512 \times 384$ . Of course this in turn reduces the PSNR. It is interesting to observe that, in some cases, very small distractors are picked from the dictionary: their keypoint density is very high and they tend to contain small repetitive visual patterns such as clothes or bricks. We conducted that same study with the three other policies without seeing any significant changes.

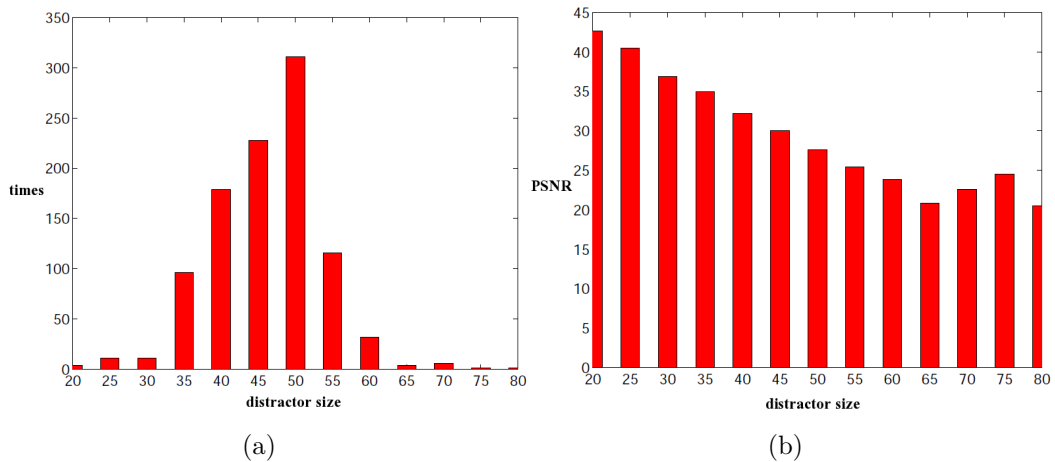


Figure 7.10: Analysis of used distractors for Policy#1: (a) histogram, (b) Impact on PSNR.

## 7.3.2 Experiment 2: Inserting Two Distractors

This experiment demonstrates that it is possible to decrease the rank of the original image by inserting more than one distractor. This allows to make even more



difficult the identification of the protected picture as (potentially) many other unrelated pictures dominate recognition. Table 7.4 gives effectiveness results when inserting two distractors in the images for different policies. This table reads as the one given above. Here, the original image is (almost) never ranked first (col 3), occasionally ranked #2 (col 4), very often ranked #3 as desired (col 5). The failure rate is  $(1+32+15+29+11)/4000 = 2.2\%$ , on average over the four policies. This is very small.

In addition, it is quite often not even found among the 100 most similar images (col 6). There are  $(274+331+282+393)/4000=32\%$  original images having rank  $\geq 100$ . It means that these images are lost. It is impossible to recognize the forged version of these images.

Across the four policies, the policy #3 seems to be the best because of its high PSNR and the conservation of the visual content.

Policies	PSNR	rank=1	rank=2	rank=3	rank>100
#1-PSNR	28.93	0	32	691	274
#2-Matching Dens.	26.80	0	15	653	331
#3-Visual Att. & PSNR	28.89	1	29	686	282
#4-Visual Att.& Matching Dens.	27.10	0	11	593	393

Table 7.4: The ranks of identified images over 1,000 queries with two distractor inserted.

## 7.4 Summary

In this chapter, we proposed an attack against a complete CBIRS. The attack is similar to the PiP-Type 2 modification, with a complex process including washing the image, building the distractor dictionary, inserting a distractor, and blurring the boundaries. The experiment results show that, from a security perspective, the created Picture in Picture visual modifications is seriously challenging the recognition power of a state-of-the-art CBIRS. A significant part of the work presented in this chapter has been published in [16].

There are some lessons learned from this work.

- The attack can delude the recognition of a complete CBIRS even if the system uses a single voting mechanism.
- The proposed attack is tested with various policies. The policy focusing on visual attention and the PSNR of image is a good choice.
- Because the boundaries of the distractor are blurred, it can resist to the PiP detection methods that use a prior knowledge for separating images.

# Chapter 8

## Conclusions and Perspectives

In this thesis, we have presented some techniques to attack a complete CBIRS. The following sections summarize our contributions and discuss the potential extensions of the thesis.

### 8.1 Conclusions

There are some lessons learnt from this work.

- As presented in chapter 2, if the pirates have knowledge about one stage of CBIRS, they can perform attacks targeting that stage. They can also design combination attacks against two or more stages of CBIRS. The better knowledge pirates have, the more chance they have to break the system.
- The three main contributions of thesis are presented in chapters 5, 6 and 7. In particular, chapter 5 presents two techniques to remove keypoints in images. Removal with Minimum Local Distortion (RMD) is designed to minimize the local distortion on images, ignoring any potential keypoint creation, while removal by smoothing (global and local smoothing) takes little care of the distortion but rather eliminates as many keypoints as possible while lowering the number of creations. Chapter 5 also presents the forging of new keypoint with minimum local distortion, called FMD. The goal of this method is to pollute the search results with false positives.

A method to attack keypoint descriptor is presented in chapter 6. The method tries to change orientation of a keypoint such that it changes its final descriptor. A learning approach using SVMs is proposed to solve this problem.

Chapter 7 proposes a method to attack the complete CBIRS where geometric verification is used to rerank the images in the result list. The attack includes some complex stages like determining candidate visual distractors, inserting visual distractors, blurring their boundaries. The results show that after the attack, the original images are not on the top of the result list.

- As presented in chapter 6 and 7, the systems with multiple voting are easier to attack than systems using single voting. The new keypoints created the attacked image (e.g. by FMD) massively match keypoints of regular images in the database thanks to the multiple voting mechanism. With the single voting mechanism, each image in the database votes at most once for each keypoint created by FMD. Hence it is difficult to outnumber the votes of the original image. In other words, the single voting improves the security of the systems.
- The Lowe criterion (section 4.1.4) in geometric verification (section 6.4.3) also improve the security of the system. The keypoints in regular images in the database are usually similar. The distances from a created keypoint (e.g. by FMD) to its nearest neighbor and second-nearest neighbor is almost equal. When the Lowe criterion is used as a pre-filtering of the keypoints used for geometric verification, the matches between the created keypoints and the keypoints of a regular image is pruned out. However, the Lowe criterion has some known drawbacks: if an image in the database has similar local descriptors (e.g. there are two logos in an image), the query descriptors are also pruned out.
- The geometric verification is not only enhancing the robustness but also improving the security of the system because the created fake keypoints (eg. by FMD) do not mimic the geometrical pattern present in regular images.

## 8.2 Perspectives

We discuss in this section further issues to be explored in the future works.

**Attacking copy-move, near identical image detection.** It is difficult to push the original image far away in the result list. However, the attacks proposed in chapter 5 and 6 can be applied to delude systems using local descriptors for object recognition [56], copy-move attack detection [2], or near duplicate image

search [12, 46]. These scenarios often consider interesting objects or small copied images, whence there is a small number of keypoints to be processed. That small number of keypoints weakens the security of the system.

**Attacking database.** In chapter 7, we know that the distractors inserted into the forged image come from images inside or outside the database. The distractors of images belonging to the database triggers many matches between the attacked image and these images. This stems into a high probability of a successful attack. The pirate wants to disclose images from the database. However, a CBIRS rather stores the local descriptors extracted from images than the whole images. The pirate would like to reconstruct image from local descriptors. In [88], the authors show that the content of image can be reconstructed from its local descriptors. To protect the database of descriptors from unauthorized parties, a normal approach is to encrypt the data. This approach opens the problem on how to run a nearest neighbor search like  $K$ -nn on an encrypted database. In [68], the authors propose a distance-preserving encryption scheme: the distance between two encrypted descriptors is same as the distance between the original descriptors. However, this transformation is not secure to known-sample attack and know-plain text attack [51].

**Visual quality enhancement.** While the PiP creation scheme and the various policies detailed in chapter 7 are able to delude the system, substantial additional work is needed to reduce the visual impact when washing images as well as when inserting carefully chosen visual distractors. It is likely that inpainting methods would help preserving the visual quality of the attacked images. It is part of our plans to go one step beyond what is presented in that chapter by integrating our security-oriented CBIRS attacks with inpainting strategies.

**Attacking other indexing schemes.** In the thesis, the database of local descriptors was indexed by the NVTree scheme [48]. However, there are many other state-of-the-art indexing schemes. Many of them use Bag of Feature - BoF model [76, 36, 38]. An attack is to disrupt the quantization step. These schemes perform the quantization with a  $K$ -means. If the pirates know the  $K$ -means visual words, they can modify the local descriptors of a query image to disturb the distance between descriptors and visual words in order that the descriptors are quantized in a wrong cell. The problem of changing distance between a descriptor and a visual word can be summarized as follows.

Consider a descriptor  $\mathbf{p}$ , and denote  $\mathbf{v}_1$  and  $\mathbf{v}_2$  the nearest and second nearest neighbors, with  $d_1 = \|\mathbf{p} - \mathbf{v}_1\|$  and  $d_2 = \|\mathbf{p} - \mathbf{v}_2\|$ , so that  $d_1 \leq d_2$ . We look for the minimum distortion to get the attacked descriptor  $\mathbf{p}_t = \mathbf{p} + \epsilon$  closer to visual

word  $\mathbf{v}_2$  than  $\mathbf{v}_1$ . A simple Lagrangian resolution gives:

$$\epsilon = \frac{d_2^2 - d_1^2}{2\|\mathbf{v}_2 - \mathbf{v}_1\|^2}(\mathbf{v}_2 - \mathbf{v}_1).$$

As often in this thesis, it is easy to find the appropriate modification in the description space, and much more difficult to find the patch in the pixel domain that produces this modification because the mapping is not linear at all.

**Attacking Video Content–Based Copy Detection (VCBCD) systems.**

There are many VCBCD systems based on a local description like SIFT [53, 54, 69, 22]. The processing and indexing of all frames from the query and/or database videos is too costly. The system indeed selects few keyframes from the videos to be processed. Intuitively, the approaches with keyframes can be attacked by applying the type of modifications in the manuscript. But it requires a special handling of the time dimension. As we have taken special care of the geometry in chapter 7, the time robustification would certainly be the main difficulty for attacking VCBCD systems.

**Counter attack.** The final goal of the research is not only to design the attacks. After disclosing the flaws of the system, we should focus on the design of counter-attacks. One of approach is to integrate cryptography techniques in CBIRS. For example, instead of computing local descriptors in the pixel domain, it might be possible to compute description in a protected domain which is the result of a secret keyed transformation of the image. Without the secret key, the pirate doesn't know how to compute a descriptor, and therefore the attacks will be much more difficult.

# Chapter 9

## Résumé étendu

### 9.1 Description du problème

La recherche d'images par le contenu fait référence au problème de recherche de contenus numériques, ici des images, dans de grandes bases de données. Elle a été utilisée jusqu'à présent dans des contextes très coopératifs et conviviaux pour lesquels elle était bénéfique à la fois pour les fournisseurs de contenu et pour les utilisateurs. Récemment, les systèmes de recherche d'images par le contenu (Content-Based Image Retrieval System—CBIRS) ont été utilisés pour filtrer le contenu multimédia afin de protéger la création de quelques-uns face à la piraterie du plus grand nombre [46, 49, 73].

Le filtrage est une application des CBIRS qui est assez différente de son objectif premier : l'environnement est maintenant hostile en ce sens que le filtrage restreint la liberté des utilisateurs, contrôlant et/ou interdisant la distribution de contenus. Dans ce cadre, les systèmes ne magnifient plus toute la richesse culturelle, mais protègent la valeur commerciale du contenu. Parce qu'il y a des biens de valeur à protéger, les pirates vont essayer de contourner ces systèmes. Par conséquent, il est légitime d'étudier soigneusement l'aspect sécurité des CBIRS comme cela est fait dans cette thèse.

#### 9.1.1 Objectif de la thèse

L'objectif de cette thèse est d'examiner ce nouveau problème de caractérisation de la sécurité des systèmes existants de détection des copies basées sur le contenu.

La thèse se concentre sur les modifications visuelles à apporter à une image protégée afin de perturber ses descripteurs de façon à ce que le système ne puisse pas détecter que l'image modifiée est une quasi-copie de l'image protégée lorsque l'image modifiée est soumise au système.

Le modèle de description étudié dans cette thèse est SIFT (Scale Invariant Feature Transform) qui est aujourd'hui considéré comme un descripteur local de référence, à l'origine de nombreux descripteurs état de l'art. Nous appuyant sur une connaissance fine du processus de calcul des SIFT, nous avons proposé quelques techniques pour tromper des CBIRS employant cette technique de description locale. Les modifications peuvent être effectuées à l'étape de détection des caractéristiques, à l'étape de calcul des descripteurs, ou les deux. Nous avons également proposé une technique pour attaquer la vérification géométrique, utilisée comme post-filtrage afin de reclasser les résultats retournés par la recherche.

### 9.1.2 Structure du manuscrit

Le manuscrit se divise en 8 chapitres. Après un chapitre introductif décrivant le contexte général et la problématique abordée dans cette thèse, le chapitre 2 définit ce qu'est la notion de sécurité d'un CBIRS, en clarifiant la différence entre les notions de robustesse et de sécurité. Le chapitre 3 décrit les différents éléments composant un CBIRS et dresse un état de l'art des CBIRS actuels, ainsi que des méthodes permettant d'attaquer du contenu et de détecter du contenu attaqué.

Le chapitre 4 présente le protocole expérimental utilisé et trois manières de créer des modifications dans les images. Ces manières sont détaillées dans les chapitres 5, 6 et 7, formant ainsi les principales contributions de cette thèse. Le chapitre 5 étudie des attaques au niveau de la détection des points d'intérêt. Les attaques visent à supprimer des points d'intérêt et/ou à en ajouter intentionnellement. Le chapitre 6 propose une méthode permettant d'attaquer le calcul du descripteur en changeant l'orientation principale des points d'intérêts. Enfin, une attaque contre un CBIRS complet incluant la vérification géométrique est proposée dans le chapitre 7. Le chapitre 8 conclut et propose un certain nombre de perspectives à ce travail.

Le reste du présent chapitre est un résumé du manuscrit, chapitre par chapitre.

## 9.2 Chapitre 2 : menaces sur la sécurité des CBIRS

Dans le chapitre 2 du manuscrit, nous définissons et discutons la nature des problèmes de sécurité des CBIRS.

La “sécurité” des CBIRS est mise à l’épreuve lorsque des pirates bâtissent des attaques après avoir accumulé une connaissance approfondie sur un système particulier, et s’être concentré sur des parties très spécifiques du système dans lesquelles des défauts ont été identifiés.

Il y a au moins quatre classes importantes d’hypothèses qui influent sur la sécurité des CBIRS.

**Modèle de confiance** : il y a généralement quatre acteurs différents qui sont ou non dignes de confiance dans n’importe quel scénario impliquant des CBIRS : (i) le titulaire légal de l’image qui a le droit de charger ses images ou leurs caractéristiques de bas niveau dans la base de données ; (ii) le serveur d’images où une collection de descriptions est indexée dans une base de données qui est utilisée pour construire les réponses aux requêtes basées sur le contenu ; (iii) l’utilisateur qui interroge le système avec une image particulière, et (iv) le logiciel client qui traite l’image requête, se connecte au serveur, envoie les requêtes, reçoit et traite les réponses avant de les afficher. Le scénario de détection de copie le plus classique pour les attaques est celui où l’utilisateur est le pirate, tous les autres acteurs étant considérés comme tiers de confiance.

**Objectifs** : le pirate peut avoir les deux grands objectifs suivants : (i) produire des faux négatifs en manipulant les images de façon à ce que le CBIRS ne parvienne pas à les détecter par la suite comme étant des quasi-copies d’images protégées ; (ii) produire de faux positifs en manipulant les images de telle sorte qu’elles soient toujours détectées (donc retournées) par le système (même pour les contenus inoffensifs).

**Mesures** : attaquer une image protégée entraîne la manipulation du contenu qui induit une distorsion. Le PSNR mesure cette distorsion à l’égard de l’image originale en terme de distance euclidienne dans des échelles logarithmiques. Une attaque est considérée comme réussie si le système ne peut pas reconnaître l’image quasi-copie et si la qualité visuelle de la quasi-copie de l’image est acceptable.

**Connaissances** : la connaissance qu’ont les pirates sur le système a un impact fort sur les attaques conçues. Plus le pirate a de connaissances, plus l’attaque qu’il conçoit peut être malicieuse.

La connaissance que l’on peut acquérir est limitée lorsque l’on utilise des techniques se fondant sur la présence de clés secrètes [40, 80]. Si un pirate dispose d’un accès complet à un logiciel de calcul de caractéristiques extraites des images, bien que ces caractéristiques et/ou le principe de leur extraction soient



protégés par une clé secrète, il peut alors créer des images et observer leurs caractéristiques résultantes, ce qui confère des informations sur la clé secrète. Le niveau de sécurité du système pourrait alors être mesuré par le nombre de couples (images, caractéristiques) nécessaire pour acquérir assez des connaissances sur cette clé pour le mettre en danger.

La plupart des CBIRS n'emploient pas de clé secrète. En général, le pirate connaît le fonctionnement du système visé car les algorithmes le composant sont connus, publiés. Leurs paramètres ne sont généralement pas pris au hasard, et il est possible d'au moins estimer les fenêtres où se trouvent leurs valeurs en utilisant le bon sens. Un point critique spécifique pour la sécurité des CBIRS est de savoir si oui ou non le pirate connaît partiellement le contenu de la base de données. Générer artificiellement des fausses alarmes devient beaucoup plus facile si le pirate a cette connaissance. Avec les connaissances sur la base de données, le pirate peut concevoir des attaques à base d'oracle.

## 9.3 Chapitre 3 : état de l'art

Dans ce chapitre un état de l'art des composants principaux d'un CBIRS à l'encontre duquel les attaques peuvent être menées est présenté, suivi d'un état de l'art des méthodes liées à la sécurité des contenus eux-mêmes.

### 9.3.1 Vue générale des CBIRS

Il y a trois composants principaux dans un CBIRS qui prennent en charge (i) l'extraction de caractéristiques des images, (ii) le processus d'indexation et de recherche et (iii) l'amélioration du résultat par l'élimination d'autant de faux positifs que possible.

La vie d'un CBIRS peut être divisée en deux étapes. La première étape calcule les caractéristiques de toutes les images de la base de données. Ces caractéristiques sont insérées dans une structure d'index multidimensionnel pour permettre des recherches rapides. Cette étape se fait hors ligne. Dans la deuxième étape, en ligne, une image requête est proposée au système par l'utilisateur. Les caractéristiques de cette image sont d'abord extraites puis utilisées pour interroger l'index. La recherche retourne alors une courte liste d'images candidates, ayant une ressemblance jugée suffisante par le système. Cette liste est analysée ensuite afin de tenter d'y détecter des faux positifs et de reclasser les images de cette liste, mettant en avant celles jugées comme vraiment similaires et en retrait ces faux positifs. La liste est finalement renvoyée à l'utilisateur.

**Extraction de caractéristiques** : une image peut être représentée par une unique caractéristique globale ou un ensemble de caractéristiques locales. La plupart des CBIRS utilisent des caractéristiques locales (encore appelées descriptions locales) pour représenter une image. L'extraction de ces caractéristiques locales comprend deux étapes. Durant la première étape, les régions d'intérêt locales dans l'image sont identifiées par un détecteur de région. Durant la deuxième étape, des descriptions de ces régions locales sont calculées. Les détecteurs de région de l'état de l'art sont principalement le détecteur de coins de Harris [30], le détecteur de coins de FAST [74], l'approche par différences de Gaussiennes DOG [56], LOG, Harris-Laplace, Harris-Affine, Hessian-Laplace, Hessian-Affine [61, 62]. Les techniques de l'état de l'art pour décrire les régions repérées sont SIFT [56], PCA-SIFT [41], SURF [5], DAISY[82, 83], GLOH [63].

**Indexation** : ce processus est chargé d'organiser dans une structure de données performante les millions ou les milliards de descriptions issues des images de la base de données. Il est en plus chargé de trouver le plus efficacement possible les  $k$  plus proches voisins de chaque descripteur calculé sur l'image requête. Les schémas d'indexation peuvent être divisés en deux catégories. La première catégorie contient des méthodes de recherche exactes, comme le KDTree [25] et le R-Trees [28]. La deuxième catégorie contient des méthodes de recherche approximatives comme celles s'appuyant sur le  $K$ -means [58] et ses descendants comme les versions approximatives de  $K$ -means (Approximate  $K$ -means (AKM), Hierarchical  $K$ -means (HKM) [72]). Certaines techniques emploient des approches sacs de mots [76, 37]. D'autres utilisent des approches basées sur des projections aléatoires comme le LSH (Locality-Sensitive Hashing) [15, 27] ou le NVTree [48].

**Élimination des faux positifs** : un CBIRS utilise diverses techniques pour améliorer la qualité des résultats de la recherche. Les approches habituelles vérifient la cohérence géométrique entre l'image requête et les images dans la liste résultat pour modifier le classement des images candidates dans cette liste. Plus il est constaté de cohérence, meilleur est le classement. Les méthodes de l'état de l'art pour ce reclassement sont la transformée généralisée de Hough (GHT) [3, 56] ou encore l'approche RANSAC [24, 52, 72]

### 9.3.2 Sécurité du contenu

De manière générale, les CBIRS actuels arrivent à reconnaître des contenus protégés bien que ceux-ci aient subi diverses distorsions. Cette capacité est d'ailleurs ce qui est vérifié durant les campagnes d'évaluations (TRECVID et autres). Il nous faut toutefois distinguer les distorsions créées dans le but de tester la robustesse des systèmes de celles créées dans le but d'en tester les aspects sécurité.

**Attaquer le contenu pour évaluer la robustesse :** L'image quasi-copie peut être créée par l'application de certaines modifications normalisées telles que la rotation, le rééchantillonnage, le redimensionnement, le recadrage, la compression, ... Des moyens rendant possibles ces modifications peuvent être trouvés dans de nombreux logiciels de traitement d'image comme Adobe Photoshop, Microsoft Paint ou encore StirMark [71]. Toutes ces modifications sont très générales, elles ne prennent jamais en compte les détails des systèmes qui vont ensuite effectuer les reconnaissances. Ce caractère général fait que ces approches se focalisent sur les aspects robustesse des systèmes.

**Attaquer le contenu pour évaluer la sécurité :** a contrario, certaines approches créent des distorsions dans les images qui tentent d'exploiter au maximum les failles des différents composants des systèmes (et c'est ce que nous faisons dans cette thèse). Très peu de travaux explorent cette voie.

Dans [33], Hsu et al. attaquent un système d'authentification d'images basé sur SIFT. Ils tentent notamment de faire disparaître certains points d'intérêt en modifiant les pixels de telle sorte que les maxima locaux disparaissent. Leur approche n'utilise toutefois pas de base d'images et ne s'appuie que sur le hachage des caractéristiques. Ce problème étant très similaire à celui qui nous intéresse, nous avons précautionneusement mis en oeuvre leur solution en utilisant une grande base d'images et un CBIRS. Il s'avère que leurs attaques ne posent aucun problème de reconnaissance au CBIRS. Leurs conclusions ne sont valides que dans leur contexte, et ne s'appliquent pas au nôtre [20].

**Détecter les contenus attaqués :** pour détecter les contenus attaqués en utilisant un CBIRS, il nous faut d'abord créer une "vérité terrain". Autrement dit, nous définissons un ensemble d'images protégées. Ces images là sont ensuite distordues selon divers processus, formant autant de quasi-copies. Chaque quasi-copie est alors utilisée en tant que requête pour interroger le système. Si le système est suffisamment robuste, il retourne alors pour chaque quasi-copie l'image originale ayant servi à sa création, celle appartenant à la vérité terrain. La tâche de détection de copie TRECVID [84] est un bon exemple de ce type de méthode. Notons que le but de cette thèse est de produire des quasi-copies qui ne seront pas reconnues car leur création exploitera les failles liées aux techniques employées dans un CBIRS particulier. Notons qu'il existe d'autres approches pour détecter des contenus attaqués : le tatouage numérique [13, 57], dans une certaine mesure le hachage d'image [80, 34, 42, 75] et les techniques de détection d'images altérées [31, 1, 2].

## 9.4 Chapitre 4 : contexte expérimental et survol des contributions

Ce chapitre présente le contexte expérimental puis les trois manières de créer des quasi-copies destinées à prendre en défaut les capacités de reconnaissance d'un CBIRS.

### 9.4.1 Critère d'évaluation

Le système échoue (c'est à dire, l'attaque est faite avec succès) lorsque la version originale de l'image attaquée n'est pas sur le haut de la liste des résultats. Elle est "cachée" derrière d'autres images. Dans les expériences présentées dans les prochains chapitres, nous regarderons le score de l'image originale et les premières meilleures images qui n'ont aucune raison d'être naturellement mises en correspondance.

Pour mesurer la qualité (la distorsion visuelle) de l'image attaquée, nous calculons le PSNR entre l'image attaquée et sa version originale.

### 9.4.2 Description de l'image, indexation et faux positif

Pour décrire une image, nous utilisons le descripteur SIFT proposé par Lowe [56, 63].

Pour effectuer une recherche rapide, les descripteurs SIFT sont indexés en utilisant le NVTree [48]. Deux approches de vote sont prises en compte : vote unique et vote multiple.

L'étape de filtrage des faux positifs repose sur l'estimation de la transformation affine entre la requête et chaque image candidate. Pour estimer cette transformation, les points d'intérêt des deux images sont d'abord mis en relation en utilisant le critère de Lowe [56]. La transformation affine sur les points mis en correspondance est estimée par une transformée généralisée de Hough (GHT) [56]. Ensuite, une approche moindres carrés [56] est adoptée pour éliminer les points incohérents par rapport au modèle. Le nombre de points restant, ceux en cohérence avec le modèle, est utilisé pour donner un score aux images, de facto éliminant de la tête du classement les faux positifs.

### 9.4.3 Dataset et les requêtes

La base de données des images est composée de 100 000 images téléchargées aléatoirement de Flickr. Cette collection donne 103 454 566 descripteurs SIFT indexés par le NV-Tree. Nous avons ensuite choisi au hasard 1000 de ces images que l'on considère dès lors comme des images protégées. C'est à partir de ces images que nous créons des modifications spéciales visant à prendre en défaut le système. Ainsi, les attaques dans les chapitres suivants sont effectuées sur ces 1000 images. Les images attaquées sont ensuite utilisées en tant que requêtes. Pour chaque requête, nous avons gardé une trace des scores des 100 images les plus ressemblantes. Le résultat final est la moyenne sur 1000 requêtes.

### 9.4.4 Trois manières d'attaquer les images et de créer des quasi-copies

Cette section présente trois idées clé pour la conception des attaques.

1. Attaquer les points d'intérêt détectés. Nous empruntons deux pistes ici. La première essaye de supprimer des points d'intérêt dans l'image. Cela rendra l'image difficile à être reconnues. La deuxième insère des points d'intérêts artificiels. Ces nouveaux points seront mis en correspondance avec des images de la base différentes de la copie originale, rendant les scores plus difficiles à trancher.
2. Attaquer les descripteurs. Le déplacement des descripteurs dans l'espace des caractéristiques (feature space) rend la mise en correspondance plus difficile, les plus proches voisins étant perturbés.
3. Attaquer la géométrie : le résultat des attaques doit être géométriquement cohérent avec les images de la base pour tenter de faire passer devant dans le classement final des images différentes de l'originale.

Chacun des trois angles d'attaque est le sujet de chaque chapitre ultérieur.

## 9.5 Chapitre 5 : attaque des points d'intérêts

Deux stratégies d'attaque des points d'intérêts pour leurrer la reconnaissance d'un système sont présentées. La première consiste à modifier l'image devant être dissimulée de façon à ce que son score avec l'image originale soit considérablement réduit. La deuxième consiste à attaquer l'image en y introduisant des éléments

visuels correspondant souvent à d'autres images de la base de données. L'image attaquée votant alors pour d'autres images de la base, l'image originale peut être classée loin dans la liste des résultats. Il est bien entendu possible de combiner ces deux stratégies.

### 9.5.1 Réduction du scores par suppression des points d'intérêts

Deux approches pour supprimer des points d'intérêts sont présentées, évaluées et discutées. La première, appelée "Removal with Minimum Local Distortion" (RMD), repose comme son nom l'indique, sur une minimisation de la distorsion locale. La seconde consiste en un lissage de l'image.

#### Suppression par minimisation de la distorsion locale (Removal with Minimum Local Distortion–RMD)

Pour le point d'intérêt  $\mathbf{x} = (x, y, \sigma)^T$ , cette approche détermine un patch  $\epsilon$  à appliquer sur la région support du point d'intérêt de telle sorte qu'il minimise la distorsion locale. À l'échelle  $\sigma$ , la différence de noyau gaussiens  $\Delta G_\sigma$  considérée pour la détection de points d'intérêt a un support spatial limité  $\mathcal{S}_\sigma$ . Il s'agit alors de déterminer le patch  $\epsilon$  défini sur  $(x, y) + \mathcal{S}_\sigma$  permettant de modifier la valeur de la différence de gaussienne (DOG) au point  $\mathbf{x}$ ,  $D(x, y, \sigma)$ , d'une quantité donnée  $\delta$ . En d'autres termes, pour  $(u, v)$  dans le voisinage de  $(x, y)$ , l'image est modifiée dans  $I'(u, v) = I(u, v) + \epsilon(u, v)$ , de sorte que  $D'(\mathbf{x}) = D(\mathbf{x}) + \delta$ . Le patch  $\epsilon$  doit être de norme Euclidienne minimale pour réduire la dégradation visuelle. Ce problème fait appel à une optimisation sous contrainte :

$$\epsilon = \arg \min_{\epsilon: D'(\mathbf{x})=D(\mathbf{x})+\delta} \frac{1}{2} \|\epsilon\|^2 \quad (9.1)$$

La contrainte étant affine et la fonction à minimiser convexe, une simple résolution de Lagrange donne le résultat suivant :

$$\epsilon = \frac{\delta}{\|\Delta G_\sigma\|^2} t_{(x,y)}(\Delta G_\sigma) \quad (9.2)$$

où  $t_{(x,y)}$  est l'opérateur de translation 2D d'un déplacement  $(x, y)$ . La première condition pour qu'un point d'intérêt  $\mathbf{x}$  soit détecté est que la valeur de sa DOG  $D(\mathbf{x})$  soit plus grande qu'un seuil de contraste  $C$ . La puissance de l'attaque par RMD est contrôlée en ciblant un nombre limité de points d'intérêt à supprimer. Ceci est réalisé en introduisant une valeur  $\delta^+ > 0$  qui définit le sous-ensemble  $\mathcal{E}_{\delta^+} = \{\mathbf{x} : C < |D(\mathbf{x})| < C + \delta^+\}$ . Supprimer les points d'intérêts de  $\mathcal{E}_{\delta^+}$  signifie que  $|D(\mathbf{x})|$  est diminuée d'un montant  $|\delta^+|$  tel que sa nouvelle valeur soit en-dessous du seuil  $C$ .

La suppression des points d'intérêts par la méthode RMD donne une distorsion minimale, cependant, un aspect indésirable de cette méthode est la création de nombreux nouveaux points d'intérêts. L'étude de ces nouveaux points d'intérêts montre qu'ils sont localisés à proximité des points d'intérêts d'origine et que leurs descripteurs sont donc potentiellement similaires.

### Suppression par lissage

Les extrema locaux de la DOG correspondent à des points d'intérêts localisés sur des discontinuités importantes de l'image. Une façon simple d'éviter leur détection consiste à lisser l'image. En n'introduisant pas de discontinuités fortes, contrairement à RMD, le lissage permet de réduire le nombre de points d'intérêts, tout en minimisant la création de nouveaux points d'intérêts. Trois mises en oeuvre de lissage sont proposées.

- (i) Lissage global (GS) : cette attaque effectue un lissage sur l'ensemble de l'image.
- (ii) Lissage local (LS) : cette attaque remplace la région de taille  $n \times n$  autour d'un point d'intérêt par sa version lissée avec un noyau gaussien dont la variance est égale à l'échelle du point d'intérêt, et vérifie si ce point d'intérêt est toujours détecté ou non. Ceci est en fait effectué itérativement à l'aide des régions de taille croissante ( $n = \{1, 3, 5, 7\}$ ) (attaques dénotées LS1, LS3, LS5, LS7), jusqu'à ce que le point d'intérêt ne soit plus détecté.
- (iii) Lissage basé sur la densité (DS) : il s'agit d'une variante de l'attaque globale. Au lieu de lisser l'ensemble de l'image, cette attaque ne lisse que les régions denses en points d'intérêts.

## 9.5.2 Création de nouveaux points d'intérêts

La stratégie que nous utilisons pour créer de nouveaux points d'intérêts avec une distorsion locale minimale est symétrique à l'attaque RMD. Dans ce cas, les extrema locaux modifiés sont dans le sous-ensemble  $\mathcal{F}_{\delta^-} = \{\mathbf{x} : C - \delta^- < |D(\mathbf{x})| < C\}$ , par l'ajout de patches qui renforcent le contraste dans le voisinage des points d'intérêts.

## 9.5.3 Évaluation des attaques

La combinaison de GS, LS et FMD peut tromper le CBIRS dans le cas d'un schéma de vote multiple : l'image originale n'est pas au sommet de la liste résultat. Elle est "cachée" derrière une autre image qui obtient un meilleur score. Le

PSNR moyen sur 1000 images entre les images attaquées et l'image originale lors de l'application d'une attaque GS +LS7 +FMD est de 28.23dB.

## 9.6 Chapitre 6 : attaque du descripteur

Le but de l'attaque proposée dans ce chapitre est de déplacer les descripteurs de l'image attaquée dans l'espace de description loin de ceux de la version originale. Ainsi, un descripteur attaqué aura moins de chance d'être mis en correspondance le descripteur correspondant de l'image originale au travers d'une recherche de plus proches voisins. Une façon de modifier le descripteur SIFT est de changer l'orientation principale des points d'intérêt, cela ayant un impact direct sur le calcul du descripteur.

L'attaque comprend deux étapes. À la première étape, un processus d'apprentissage basé sur des SVM est utilisé pour apprendre l'hyperplan séparant les régions support de points d'intérêts ayant des orientations différentes. Une fois l'hyperplan appris, il est possible, étant donné un point d'intérêt situé d'un côté de l'hyperplan, de calculer la déformation minimale à appliquer à sa région support de façon à la "pousser" vers l'autre côté de l'hyperplan.

### 9.6.1 Impact du changement de l'orientation sur les descripteurs

Pour voir la relation entre l'orientation de points d'intérêts et le descripteur SIFT résultant, nous avons modifié artificiellement les orientations des points d'intérêts détectés sur l'image Lena, en modifiant le code source de calcul des descripteurs utilisé (VLFeat [86]). Le résultat de ces simulations montre que la distance la plus grande entre descripteurs est atteinte lorsque l'orientation principale de la région support est modifiée de  $\pi/2$  et  $3\pi/2$ .

### 9.6.2 Utilisation d'un SVM pour modifier les orientations

Une collection de SVM différents nous servent à apprendre la déformation minimale  $\epsilon$  des régions support pour en changer l'orientation de  $\pi/2$ . Chaque SVM détermine l'hyperplan séparant les points d'intérêts ayant une orientation  $\theta_1$  des points d'intérêts ayant une orientation  $\theta_2 = \theta_1 + \pi/2$ . Pour faciliter l'apprentissage, réduire le bruit et être plus efficace, l'espace des orientations est quantifié en pas de longueur  $\pi/18$ . Soit  $\mathcal{X}_1 = (r_i, l_i)_i$  l'ensemble d'apprentissage composé de régions



support normalisées  $r_i$  ayant une orientation  $\theta_1$ , formant la classe étiquetée par  $l_i = +1$ .  $\mathcal{X}_2 = (r_j, l_j)_j$  est l'ensemble d'apprentissage composé de régions support normalisées  $\mathbf{r}_j$  dont l'orientation est de  $\theta_2 = \theta_1 + \pi/2$ , formant la classe duale marquée par  $l_j = -1$ . durant la phase d'apprentissage, le SVM en charge de  $\theta_1$  et  $\theta_2$  apprend les paramètres lié à l'hyperplan  $(\mathbf{w}, b)$  séparant  $\mathcal{X}_1$  et  $\mathcal{X}_2$  déterminé en trouvant la solution à :

$$\begin{aligned} \ell_k \cdot (\langle \mathbf{w}, \Phi(\mathbf{r}_k) \rangle + b) &\geq 1 \quad \forall \mathbf{r}_k \in \{\mathcal{X}_1, \mathcal{X}_2\}, \\ \text{with } \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle &= \sum_{k:\alpha_k > 0} \alpha_k \ell_k K(\mathbf{x}, \mathbf{r}_k), \end{aligned}$$

où  $\Phi$  projette  $\mathbf{x}$  vers un espace de dimension supérieure,  $\alpha_k$  sont les multiplicateurs de Lagrange, et  $K$  est une fonction noyau radiale (RBF):

$$K(\mathbf{x}, \mathbf{r}_k) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{r}_k) \rangle = \exp\left(-\frac{\|\mathbf{x} - \mathbf{r}_k\|^2}{2\sigma^2}\right). \quad (9.3)$$

le SVM déterminé ainsi est utilisé pour calculer la déformation  $\boldsymbol{\epsilon}$  de norme minimale devant être ajouté à une région  $\mathbf{r} \in \mathcal{X}_1$ , de telle sorte que  $\mathbf{r} + \boldsymbol{\epsilon} \in \mathcal{X}_2$ .

La trouver demande de résoudre l'optimisation suivante:

$$\min \frac{1}{2} \|\boldsymbol{\epsilon}\|^2 \quad (9.4)$$

$$\text{s.t. } \sum_{k:\alpha_k > 0} \alpha_k \ell_k K(\mathbf{r} + \boldsymbol{\epsilon}, \mathbf{r}_k) + b = -\Delta d, \quad (9.5)$$

$$\text{and } 0 \leq r_i + \epsilon_i \leq 1, \quad \forall i \in \{1, \dots, L\} \quad (9.6)$$

où  $\Delta d > 0$  est la distance entre  $\mathbf{r} + \boldsymbol{\epsilon}$  et l'hyperplan  $(\mathbf{w}, d)$ . La valeur de  $\Delta d$  est proportionnelle à la probabilité  $r + \boldsymbol{\epsilon} \in X_2$ .

Eq. (9.6) assure que la région modifiée demeure dans la fourchette  $[0, 1]$ . En réécrivant  $a_k = \alpha_k \ell_k K(\mathbf{r}, \mathbf{r}_k)$ , et  $\mathbf{c}_k = 2(\mathbf{r} - \mathbf{r}_k)$ , Eq. (9.5) devient alors :

$$\sum_{k:\alpha_k > 0} a_k \exp\left(-\frac{\mathbf{c}_k^\top \boldsymbol{\epsilon} + \|\boldsymbol{\epsilon}\|^2}{2\sigma^2}\right) + b + \Delta d = 0. \quad (9.7)$$

en dérivant (9.7) :

$$\sum_{k:\alpha_k > 0} a_k (\mathbf{c}_k + 2\boldsymbol{\epsilon}) \frac{-1}{2\sigma^2} \exp\left(-\frac{\mathbf{c}_k^\top \boldsymbol{\epsilon} + \|\boldsymbol{\epsilon}\|^2}{2\sigma^2}\right) \quad (9.8)$$

Cette minimisation de problème sous contraintes est résolue en utilisant une approche interior point method [7, 8], donnant l' $\boldsymbol{\epsilon}$  à appliquer.

### 9.6.3 Évaluation

L'attaque d'orientation arrive à leurrer les CBIRS utilisant système de vote multiple. L'image d'origine n'est pas au sommet de la liste résultat. Le PSNR entre les images originales et leurs quasi-copies attaquées selon une approche orientation comme décrit dans ce chapitre est de 28.39dB, en moyenne sur 1000 images.

## 9.7 Chapitre 7 : prendre en compte la cohérence géométrique

Ce chapitre propose une méthode de modifications de quasi-copie qui prenne en défaut des CBIRS employant un filtrage des faux positifs grâce à une prise en compte de la cohérence géométrique entre la requête et les images candidates du résultat. Cette méthode est de type image-dans-image (Picture in Picture–PiP, voir TRECVID [84, 70]).

Cependant, contrairement à ce qui est fait en général avec les distorsions de type PiP qui sont généralistes et testent la robustesse des systèmes, nous créons ici des quasi-copies où l'image insérée l'est avec un but sécurité. Autrement dit, on veut produire des attaques qui non seulement prennent la technique de description en défaut (SIFT) mais aussi tirent parti de la connaissance du fonctionnement de la phase de cohérence géométrique.

Pour résumer notre technique, nous insérons dans une image une petite imagerie déterminée de manière à ce qu'elle attire le plus possible de mises en correspondance avec une ou plusieurs images de la base, assurant ainsi une parfaite cohérence géométrique. Nos objectifs sont donc de (i) faire en sorte que l'imagerie insérée domine la reconnaissance, (ii) qu'il soit complexe de séparer cette imagerie insérée de l'image dans laquelle elle est incrustée.

La méthode proposée ici est en deux étapes. D'abord nous "lavons" une copie d'une image protégée comme nous l'avons expliqué dans les chapitres précédents, cela afin de réduire autant que possible le nombre de descripteurs pouvant être mis en correspondance. Ensuite nous déterminons les caractéristiques de l'imagerie à incruster avant de l'insérer, formant ainsi la quasi-copie. Nous montrons que cette approche réussit à empêcher l'identification des images originales à partir de telles quasi-copies.

### 9.7.1 Lavage d'image

L'objectif de cette étape est de réduire le nombre de correspondances entre la quasi-copie de l'image originale et l'image originale, ce qui rend plus difficile de reconnaître le contenu protégé. Nous appliquons une combinaison de DS (voir Section 9.5.1), de changement forcé d'orientation (voir Section 9.6) puis l'attaque LS (voir Section 9.5.1) pour réduire le nombre de correspondances satisfaisant le critère de Lowe (voir 9.4.2) entre la quasi-copie et sa version originale.

### 9.7.2 Création de l'attaque PiP

L'objectif de cette étape est de créer une imagerie qui sera à incruste et qui dominera la reconnaissance. Pour cela, nous créons d'abord un dictionnaire d'images puis déterminons où incruste.

**Création du dictionnaire d'images :** nous analysons la base d'images si cela est possible ou non téléchargeons un grand nombre d'images aléatoires, de Flickr par exemple. Ces images sont analysées et on en extrait des régions carrées de tailles variables très denses en points d'intérêt. Ces régions forment le dictionnaire d'images.

**Insertion de l'image :** étant donnée une image à insérer, quatre stratégies sont proposées pour déterminer où celle-ci est à incruste dans la quasi-copie lavée. L'idée générale est d'insérer l'image  $I_p$  dans l'image lavée  $I_w$  en ajustant son illumination à chaque position  $I_w(x, y, p)$ . Cet ajustement s'opère ainsi :  $\tilde{I}_p = I_p - \bar{I}_p$ . Déterminer cet ajustement exige de trouver les paramètres  $a$  et  $b$  de  $a\tilde{I}_p + b$  tels que:

$$\min_{a,b} \|a\tilde{I}_p + b - I_w(x, y, p)\|^2, \text{ subject to: } a \geq a_{min}. \quad (9.9)$$

$a \geq a_{min}$  évite l'uniformisation de  $\tilde{I}_p$  par de trop petites valeurs pour  $a$ , ce qui supprimerait la plupart (sinon tous) des points d'intérêts. Résoudre (9.9) donne  $a$  et  $b$ :

$$\begin{aligned} b &= \bar{I}_w(x, y, p), \\ a &= \max \left( a_{min}, \frac{\tilde{I}_p^T I_w(x, y, p)}{\|\tilde{I}_p\|^2} \right). \end{aligned} \quad (9.10)$$

Les quatre stratégies sont :

- **Stratégie #1 : orientée PSNR** On trouve la place dans l'image lavée où l'image ajustée peut être insérée en réduisant le PSNR aussi peu que possible.

- **Stratégie #2 : orientée densité** Cette politique détermine l'endroit où insérer l'imagette ajustée en fonction du nombre de correspondances qui existent entre l'image lavée et sa version d'origine. L'imagette est mise à la position ayant le nombre maximum de correspondances.
- **Stratégie #3 : orientée attention visuelle & PSNR** Cette politique détermine les régions saillantes de l'image, puis applique la stratégie #1. L'imagette ne peut se mettre dans les régions saillantes.
- **Stratégie #4 : orientée attention visuelle & densité** Une fois que les régions saillantes ont été déterminées, la stratégie #2 est appliquée. L'imagette ne peut se mettre dans les régions saillantes.

### 9.7.3 Frontières floues

L'objectif de cette étape est de rendre la séparation des imagettes du contenu lavé très difficile. Ceci est obtenu en brouillant les frontières de l'imagette par une gaussienne.

### 9.7.4 Évaluation de la méthode

Le tableau 9.1 présente les résultats d'efficacité quand une imagette est insérée dans une quasi-copie lavée. La deuxième colonne donne le PSNR moyen entre les images originales et leur version attaquée, pour les quatre stratégies. La troisième colonne donne le nombre de fois que l'attaque PIP est en échec car le système retrouve tout de même l'image originale à partir de la quasi-copie contenant une imagette.

stratégie	PSNR	rang=1	rang=2	rang>100
#1	29.55	25	923	47
#2	28.14	17	921	60
#3	29.52	20	928	46
#4	28.35	13	918	63

Table 9.1: Les rangs des images identifiées en moyenne sur 1000 images, 1 imagette insérée.

Le taux d'échec de l'attaque est très faible  $((25+17+20+13)/4000 = 1.88\%$ . La quatrième colonne donne le nombre de fois où l'image originale n'est pas trouvée dans les 100 meilleures images les plus similaires. Il y a  $(47+60+46+63)/4000 = 5.4\%$  d'images originales ayant un rang supérieur à 100. Cela signifie que ces images sont perdues. Il est impossible de reconnaître la version forgée de ces images.

## 9.8 Chapitre 8: Conclusions et perspectives

Dans cette thèse, nous avons présenté quelques techniques pour attaquer un système CBIR complet. Les sections suivantes résument nos contributions et posent quelques extensions potentielles.

### 9.8.1 Conclusions

Si les pirates connaissent les briques techniques qui composent le système CBIR, ils créent d'abord des attaques spécifiques mettant à mal chaque composante, puis les combinent pour mettre à terre le système. Plus fine est leur connaissance, plus ils ont de chance d'atteindre leur but. Le chapitre 5 présente deux techniques pour enlever des points d'intérêts dans l'image. Elles sont appelées RMD (Removal with Minimum Local Distortion) et GS, DS ou LS pour le filtrage passe-bas. Le chapitre 5 présente une technique de création artificielle de nouveaux points d'intérêt avec une distorsion locale minimale (FMD). Une méthode d'attaque des descripteurs est détaillée dans le chapitre 6. La méthode tente de changer l'orientation principale des régions supports pour perturber la description. Le chapitre 7 a proposé une attaque contre un système CBIR complet où une vérification géométrique raffine la liste de résultats. L'attaque comporte quelques étapes complexes pour déterminer les "distracteurs" visuels et les insérer en estompant leurs frontières.

Les systèmes à vote multiple sont plus facile à attaquer que les systèmes à vote simple. L'usage du critère de Lowe dans la vérification géométrique (chapitre 7) améliore la sécurité du système. L'usage de la vérification géométrique accroît non seulement la robustesse, mais aussi la sécurité du système.

### 9.8.2 Perspectives

**Attaquer la base de données.** Dans le chapitre 7, le distracteur inséré dans l'image provient d'images riches en points d'intérêt. Cependant, si les distracteurs

sont originaires des images de la base de données, il aura beaucoup plus de correspondances entre l'image attaquée et l'image dont les distracteurs extraits. Cela donnera une plus grande chance de succès.

**Améliorer la qualité visuelle.** La qualité des images attaquées par les méthodes des chapitres 5, 6, et 7 peut être renforcée par des stratégies de type “inpainting”.

**Attaquer d'autres schémas d'indexation.** Il y a de nombreux schémas d'indexation dans l'état de l'art. Plusieurs d'entre eux utilisent notamment le modèle “sac de mot” (Bag Of Feature—BoF) [76, 36, 38]. Il serait intéressant de tester les attaques proposées et également de concocter d'autres attaques sur ces techniques d'indexation.

**Attaquer la vidéo.** Il y a beaucoup de systèmes de détection de copies basés sur le contenu vidéo (VCBCD) utilisant une description locale [53, 54, 69, 22]. Le succès de l'attaque sur CBIRS dans cette thèse lève aussi une alarme pour ces systèmes VCBCD.

**Contre attaquer.** Connaissant mieux les vulnérabilités du système, nous devrions nous concentrer sur le problème de la contre-attaque. Une des approches possibles est d'intégrer des techniques de cryptographie pour CBIRS.



# Bibliography

- [1] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra. Geometric tampering estimation by means of a SIFT-based forensic analysis. In *Acoustics Speech and Signal Processing (ICASSP)*, 2010 IEEE International Conference on, pages 1702–1705, march 2010.
- [2] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra. A SIFT-based forensic method for copy & move attack detection and transformation recovery. *Information Forensics and Security, IEEE Transactions on*, 6(3):1099–1110, sept 2011.
- [3] D. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.
- [4] J. M. Barrios and B. Bustos. Content-based video copy detection: PRISMA at TRECVID 2010. In *TRECVID*, 2010.
- [5] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision, ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin / Heidelberg, 2006. 10.1007/11744023-32.
- [6] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002.
- [7] R. H. Byrd, J. C. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.
- [8] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM J. on Optimization*, 9(4):877–900, apr 1999.



- [9] F. Cayre, C. Fontaine, and T. Furon. Watermarking security: theory and practice. *Signal Processing, IEEE Transactions on*, 53(10):3976 – 3987, oct. 2005.
- [10] J. Chen and J. Jiang. University of Bradford at TRECVID 2008: Content based copy detection task. In *TRECVID*, 2008.
- [11] M. E. Choubassi and P. Moulin. On the fundamental tradeoff between watermark detection performance and robustness against sensitivity analysis attacks. In E. J. Delp III and P. W. Wong, editors, *Proc of SPIE IS&T Electronic Imaging*, volume 6072, page 60721I. SPIE, 2006.
- [12] O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. In *Proceedings of the 6th ACM international conference on Image and video retrieval, CIVR '07*, pages 549–556, New York, NY, USA, 2007. ACM.
- [13] I. Cox, J. Bloom, and M. Miller. *Digital Watermarking: Principles and Practice*. Morgan Kaufmann, 2001.
- [14] I. Cox, M. Miller, T. Minka, T. Papatomas, and P. Yianilos. The Bayesian image retrieval system, PicHunter: theory, implementation, and psychophysical experiments. *Image Processing, IEEE Transactions on*, 9(1):20 –37, jan 2000.
- [15] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry, SCG '04*, pages 253–262, New York, NY, USA, 2004. ACM.
- [16] T.-T. Do, L. Amsaleg, E. Kijak, and T. Furon. Security-oriented picture-in-picture visual modification. In *ACM International Conference on Multimedia Retrieval, ICMR'12, Hong-Kong, June 2012*.
- [17] T.-T. Do, E. Kijak, L. Amsaleg, and T. Furon. Enlarging hacker’s toolbox: deluding image recognition by attacking keypoint orientations. In *IEEE International Conference on Acoustics Speech and Signal Processing, Kyoto, Japan, mar 2012*. IEEE.
- [18] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Challenging the security of content-based image retrieval systems. In *Multimedia Signal Processing (MMSP), 2010 IEEE International Workshop on*, pages 52 –57, oct. 2010.

- [19] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Deluding image recognition in SIFT-based CBIR systems. In Proceedings of the 2nd ACM workshop on Multimedia in forensics, security and intelligence, MiFor '10, pages 7–12, New York, NY, USA, 2010. ACM.
- [20] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Understanding the security and robustness of SIFT. In Proceedings of the international conference on Multimedia, MM '10, pages 1195–1198, New York, NY, USA, 2010. ACM.
- [21] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In Proceedings of the ACM International Conference on Image and Video Retrieval, CIVR '09, pages 19:1–19:8, New York, NY, USA, 2009. ACM.
- [22] M. Douze, H. Jegou, and C. Schmid. An image-based approach to video copy detection with spatio-temporal post-filtering. *Multimedia, IEEE Transactions on*, 12(4):257–266, june 2010.
- [23] J. W. Earl. Tangential sensitivity analysis of watermarks using prior information. In E. J. Delp III and P. W. Wong, editors, Proc of SPIE IS&T Electronic Imaging, volume 6505, page 650519. SPIE, 2007.
- [24] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
- [25] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):209–226, Sept. 1977.
- [26] N. Gengembre and S.-A. Berrani. The Orange Labs real time video copy detection system - TRECVID 2008 results. In TRECVID, 2008.
- [27] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In Proceedings of the 25th International Conference on Very Large Data Bases, VLDB '99, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.
- [28] A. Guttman. R-trees: A dynamic index structure for spatial searching. In B. Yorlmark, editor, SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, pages 47–57. ACM Press, 1984.
- [29] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems*, pages 545–552, 2007.

- [30] C. Harris and M. Stephens. A combined corner and edge detection. In Proceedings of The Fourth Alvey Vision Conference, pages 147–151, 1988.
- [31] J. He, Z. Lin, L. Wang, and X. Tang. Detecting doctored JPEG images via DCT coefficient analysis. In Proceedings of the 9th European conference on Computer Vision - Volume Part III, ECCV'06, pages 423–435, Berlin, Heidelberg, 2006. Springer-Verlag.
- [32] M. Hill, G. Hua, B. Huang, M. Merler, A. Natsev, J. R. Smith, L. Xie, H. Ouyang, and M. Zhou. IBM research TRECVID-2010 video copy detection and multimedia event detection system. In TRECVID, 2010.
- [33] C.-Y. Hsu, C.-S. Lu, and S.-C. Pei. Secure and robust SIFT. In Proceedings of the 17th ACM international conference on Multimedia, MM '09, pages 637–640, New York, NY, USA, 2009. ACM.
- [34] L. Huston, R. Sukthankar, and Y. Ke. Evaluating keypoint methods for content-based copyright protection of digital images. In Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on, page 4 pp., july 2005.
- [35] H. Jégou, M. Douze, G. Gravier, C. Schmid, and P. Gros. INRIA LEAR-TEXMEX: Video copy detection task. In TRECVID, 2010.
- [36] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In Proceedings of the 10th European Conference on Computer Vision: Part I, ECCV '08, pages 304–317, Berlin, Heidelberg, 2008. Springer-Verlag.
- [37] H. Jégou, M. Douze, and C. Schmid. Packing bag-of-features. In 12th International Conference on Computer Vision (ICCV '09), Kyoto, Japan, 2009. IEEE Computer society. RAFFUT, QUAERO.
- [38] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. International Journal of Computer Vision, 87(3):316–336, feb 2010.
- [39] A. Jepson. utvistoolbox. <http://www.cs.toronto.edu/~jepson/csc2503/index09.html>.
- [40] M. Johnson and K. Ramchandran. Dither-based secure image hashing using distributed coding. In Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on, volume 2, pages II – 751–4 vol.3, sept. 2003.

- [41] Y. Ke and R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II-506 – II-513 Vol.2, june-2 july 2004.
- [42] Y. Ke, R. Sukthankar, and L. Huston. An efficient parts-based near-duplicate and sub-image retrieval system. In *Proceedings of the 12th annual ACM international conference on Multimedia, MULTIMEDIA '04*, pages 869–876, New York, NY, USA, 2004. ACM.
- [43] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, 9:5–38, jan 1883.
- [44] M. Kokare, P. Biswas, and B. Chatterji. Texture image retrieval using rotated wavelet filters. *Pattern Recognition Letters*, 28(10):1240 – 1249, jul 2007.
- [45] S. Kozat, R. Venkatesan, and M. Mihcak. Robust perceptual image hashing via matrix invariants. In *Image Processing, 2004. IICIP '04. 2004 International Conference on*, volume 5, pages 3443 – 3446 Vol. 5, oct. 2004.
- [46] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *Proceedings of the 6th ACM international conference on Image and video retrieval, CIVR '07*, pages 371–378, New York, NY, USA, 2007. ACM.
- [47] S. Lazebnik, C. Schmid, and J. Ponce. Sparse texture representation using affine-invariant neighborhoods. In *International Conference on Computer Vision & Pattern Recognition*, pages 319–324, 2003.
- [48] H. Lejsek, F. Asmundsson, B. Jonsson, and L. Amsaleg. NV-Tree: An efficient disk-based index for approximate search in very large high-dimensional collections. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):869 –883, may 2009.
- [49] H. Lejsek, F. H. Ásmundsson, B. T. Jónsson, and L. Amsaleg. Scalability of local image descriptors: a comparative study. In *Proceedings of the 14th annual ACM international conference on Multimedia, MULTIMEDIA '06*, pages 589–598, New York, NY, USA, 2006. ACM.
- [50] Y. Liang, B. Cao, J. Li, C. Zhu, Y. Zhang, C. Tan, G. Chen, C. Sun, J. Yuan, M. Xu, and B. Zhang. Athu-ing at trecvid 2009. In *TRECVID, 2009*.

- [51] K. Liu, C. Giannella, and H. Kargupta. An attacker's view of distance preserving maps for privacy preserving data mining. In J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, editors, *Knowledge Discovery in Databases: PKDD 2006*, volume 4213 of *Lecture Notes in Computer Science*, pages 297–308. Springer Berlin / Heidelberg, 2006. 10.1007/1187163730.
- [52] Z. Liu, T. Liu, D. C. Gibbon, and B. Shahraray. Effective and scalable video copy detection. In *Proceedings of the international conference on Multimedia information retrieval, MIR '10*, pages 119–128, New York, NY, USA, 2010. ACM.
- [53] Z. Liu, T. Liu, and B. Shahraray. AT&T Research at TRECVID 2009 content-based copy detection. In *TRECVID, 2009*.
- [54] Z. Liu, E. Zavesky, N. Sawant, and B. Shahraray. AT&T research at trecvid 2010. In *TRECVID, 2010*.
- [55] D. Lowe. Demo software: Sift keypoint detector. <http://www.cs.ubc.ca/~lowe/keypoints/>, 2005.
- [56] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, nov 2004.
- [57] C.-S. Lu, S.-W. Sun, C.-Y. Hsu, and P.-C. Chang. Media hash-dependent image watermarking resilient against both geometric attacks and estimation attacks based on false positive-oriented detection. *Multimedia, IEEE Transactions on*, 8(4):668–685, aug. 2006.
- [58] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.
- [59] M. Malkin and R. Venkatesan. The randlet transform: Applications to universal perceptual hashing and image identification. In *Proc of the Annual Allerton Conference on Communications, Control, and Computing*, Monticello, IL, USA, 2004.
- [60] Y. Mao and M. Wu. Unicity distance of robust image hashing. *Information Forensics and Security, IEEE Transactions on*, 2(3):462–467, sept. 2007.
- [61] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision-Part I, ECCV '02*, pages 128–142, London, UK, UK, 2002. Springer-Verlag.

- [62] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, oct 2004.
- [63] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, oct. 2005.
- [64] R. Mukai, T. Kurozumi, K. Hiramatsu, T. Kawanishi, H. Nagano, and K. Kashino. NTT communication science laboratories at TRECVID 2010 content-based copy detection. In *TRECVID, 2010*.
- [65] P. C. na, L. Pérez-Freire, and F. Pérez-González. Blind newton sensitivity attack. *IEE Proceedings on Information Security*, 153(3):115–125, sep 2006.
- [66] C. W. Ngo, S. A. Zhu, H. K. Tan, W. L. Zhao, and X. Y. Wei. VIREO at TRECVID 2010: Semantic indexing, known-item search, and content-based copy detection. In *TRECVID, 2010*.
- [67] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, may 2001.
- [68] S. R. M. Oliveira and O. R. Zaïane. Privacy preserving clustering by data transformation. In A. H. F. Laender, editor, *SBBD*, pages 304–318. UFAM, 2003.
- [69] O. Orhan, J. Hochreiter, J. Pooch, Q. Chen, A. Chabra, and M. Shah. University of central florida at TRECVID 2008 content based copy detection and surveillance event detection. In *TRECVID, 2008*.
- [70] P. Over, G. Awad, M. Michel, J. Fiscus, W. Kraaij, and A. F. Smeaton. TRECVID 2011 - an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *TRECVID, 2011*.
- [71] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn. Attacks on copyright marking systems. In *Proceedings of the Second International Workshop on Information Hiding*, pages 218–238, London, UK, UK, 1998. Springer-Verlag.
- [72] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, june 2007.

- [73] S. Poullot, M. Crucianu, and O. Buisson. Scalable mining of large video databases using copy detection. In Proceedings of the 16th ACM international conference on Multimedia, MM '08, pages 61–70, New York, NY, USA, 2008. ACM.
- [74] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In Proceedings of the 9th European conference on Computer Vision - Volume Part I, ECCV'06, pages 430–443, Berlin, Heidelberg, 2006. Springer-Verlag.
- [75] S. Roy and Q. Sun. Robust hash for detecting and localizing image tampering. In Image Processing, 2007. IICIP 2007. IEEE International Conference on, volume 6, pages VI –117 –VI –120, 16 2007-oct. 19 2007.
- [76] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2, ICCV '03, pages 1470–, Washington, DC, USA, 2003. IEEE Computer Society.
- [77] M. Stricker and M. Swain. The capacity and the sensitivity of color histogram indexing. Technical report, Communications Technology Lab, 1994.
- [78] C. Sun, J. Li, B. Zhang, and Q. Zhang. ATHU-IMG at TRECVID 2010. In TRECVID, 2010.
- [79] M. J. Swain and D. H. Ballard. Color indexing. International Journal of Computer Vision, 7(1):11–32, nov 1991.
- [80] A. Swaminathan, Y. Mao, and M. Wu. Robust and secure image hashing. Information Forensics and Security, IEEE Transactions on, 1(2):215 – 230, june 2006.
- [81] Q. H. Thu and M. Ghanbari. Scope of validity of PSNR in image/video quality assessment. Electronics Letters, pages 800–801, 2008.
- [82] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pages 1 –8, june 2008.
- [83] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 32(5):815 –830, may 2010.
- [84] TRECVID. Content-based copy detection. <http://www-nlpir.nist.gov/projects/tv2008/#4.5>, 2008.

- [85] Y. Uchida, S. Sakazawa, M. Agrawal, and M. Akbacak. Kddi labs and sri international at trecvid 2010: Content-based copy detection. In TRECVID, 2010.
- [86] A. Vedaldi and B. Fulkerson. VLfeat: an open and portable library of computer vision algorithms. In Proceedings of the international conference on Multimedia, MM '10, pages 1469–1472, New York, NY, USA, 2010. ACM.
- [87] J.-E. Vila-Forcen, S. Voloshynovskiy, O. J. Koval, F. Perez-Gonzalez, and T. Pun. Worst-case additive attack against quantization-based data-hiding methods. In E. J. Delp III and P. W. Wong, editors, Proc of SPIE IS&T Electronic Imaging, volume 5681, pages 136–146. SPIE, 2005.
- [88] P. Weinzaepfel, H. Jegou, and P. Perez. Reconstructing an image from its local descriptors. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pages 337 –344, june 2011.
- [89] W. L. Zhao, D. Borth, and T. M. Breuel. Participation at TRECVID 2011 semantic indexing & content-based copy detection tasks. In TRECVID, 2011.





# List of Figures

1.1	Visual rendering of the attacked Lena . . . . .	7
3.1	General overview of a CBIRS . . . . .	18
3.2	StirMark attack on Lena image . . . . .	21
3.3	An example of tampering attack . . . . .	21
3.4	Hsu’s attack on DoG coefficients . . . . .	23
4.1	Illustration of the octaves of the scale space . . . . .	30
4.2	Neighborhood of a DoG coefficient . . . . .	31
5.1	Impact of the RMD patch of the DoG coefficients . . . . .	39
5.2	Visual distortions caused by RMD . . . . .	40
5.3	Visual distortions caused by GS+LS7 . . . . .	42
5.4	Density-based Smoothing on Lena image . . . . .	44
5.5	Visual distortions caused by FMD . . . . .	45
5.6	Euclidean descriptors distance vs. keypoints distance . . . . .	48
5.7	Euclidean descriptors distance vs. scale shift . . . . .	49
5.8	Average score vs. PSNR. JPEG-, GS+LS and RMD . . . . .	50
5.9	Image scores in realistic settings . . . . .	51
5.10	Images of the database often ranked #1 when the attack succeeds.	52
5.11	Image scores with single voting mechanism . . . . .	53
6.1	The role of the orientation in the description of the keypoint . . . . .	56
6.2	Euclidean descriptors distance vs. orientation distance . . . . .	57
6.3	The way we find $\epsilon$ . . . . .	60
6.4	Orientation changes analysis for $\Delta d = 2$ and $\Delta d = 3$ . . . . .	62
6.5	Orientation changes comparison for $\Delta d = 2$ . . . . .	64
6.6	Illustration for orientation attack of keypoints . . . . .	67
6.7	Scores of the images under realistic settings . . . . .	68
6.8	Visual rendering along the successive attacks . . . . .	70

7.1	Two examples from TRECVID. . . . .	77
7.2	How to detect a PiP attack . . . . .	78
7.3	Matching between a distractor and its original image . . . . .	83
7.4	Illustration for illumination adjustment . . . . .	85
7.5	Matching density-oriented distractor insertion . . . . .	86
7.6	Saliency map. . . . .	87
7.7	Blurring the boundaries . . . . .	88
7.8	Illustrating the four distractor insertion policies. . . . .	89
7.9	When the attack fails . . . . .	90
7.10	Analysis of the distractors' size for Policy#1 . . . . .	91

## List of Tables

5.1	Number of deleted and created keypoints by RMD . . . . .	43
5.2	Properties of the new keypoints. Lena image . . . . .	47
5.3	Properties of the new keypoints. 1000 images . . . . .	47
6.1	Illustration of patches in 36 orientation bins used for SVMs . . . . .	59
6.2	Visual examples of patches $\epsilon$ . . . . .	61
6.3	Number of keypoints whose orientation has changed . . . . .	63
6.4	Number of keypoints whose orientation has changed . . . . .	65
6.5	Average PSNR vs $t_{PSNR}$ . . . . .	65
6.6	Number of keypoints for each category in attacked images . . . . .	66
6.7	$t_{PSNR}$ in dB for the single attacks of the combination. . . . .	69
6.8	Number of keypoints on average over 1,000 images. Attacks are applied in sequel. . . . .	70
6.9	Scores of the original images and of the three other best matches .	71
6.10	The average scores over the 1,000 queries . . . . .	72
7.1	Number of keypoints and matches between washed and original .	80
7.2	Dictionary of keypoint-dense visual distractors. . . . .	84
7.3	The ranks of the identified images with one distractor . . . . .	89
7.4	The ranks of the identified images with two distractors . . . . .	92
9.1	Les rangs des images identifiées en moyenne sur 1000 images, 1 imagerie insérée. . . . .	111





## Résumé

Les systèmes de recherche d'images par le contenu (Content-Based Image Retrieval System – CBIRS) sont maintenant couramment utilisés comme mécanismes de filtrage contre le piratage des contenus multimédias. Ces systèmes utilisent souvent le schéma de description d'images SIFT pour sa bonne robustesse face à un large spectre de distorsions visuelles. Mais aucun de ces systèmes n'a encore abordé le problème du piratage à partir d'un point de vue "sécurité". Cette thèse a comme objectif d'analyser les CBIRS de ce point de vue sécurité. Il s'agit de comprendre si un pirate peut produire des distorsions visuelles perturbant les capacités de reconnaissances d'un système en créant ces distorsions en fonction des techniques que ce système utilise. Tout d'abord, nous présentons les failles de sécurité des composantes typiques d'un CBIRS : composantes description d'image, indexation et filtrage des faux positifs. Ensuite, nous présentons des attaques ciblant le schéma de description SIFT. Les attaques sont effectuées durant l'étape de détection de points d'intérêt et de calculs des descripteurs. Nous présentons également une attaque ciblant la mise en correspondance des images sur un critère de cohérence géométrique. Les expériences menées avec 100 000 images réelles confirment l'efficacité des attaques proposées.

## Abstract

Content-Based Image Retrieval Systems (CBIRS) are now commonly used as a filtering mechanism against the piracy of multimedia contents. These systems often use the SIFT local-feature description scheme as its robustness against a large spectrum of image distortions has been assessed. But none of these systems have addressed the piracy problem from a "security" perspective. This thesis checks whether CBIRS are secure: Can pirates mount violent attacks against CBIRS by carefully studying the technology they use? First, we present the security flaws of the typical technology blocks used in state-of-the-art CBIRS. Then, we present very SIFT-specific attacks. The attacks are performed either during the keypoint detection step or during the keypoint description step. We also present a security-oriented Picture in Picture attack deluding CBIRS using post filtering geometric verifications. Experiments with a database made of 100,000 real world images confirm the effectiveness of proposed attacks.