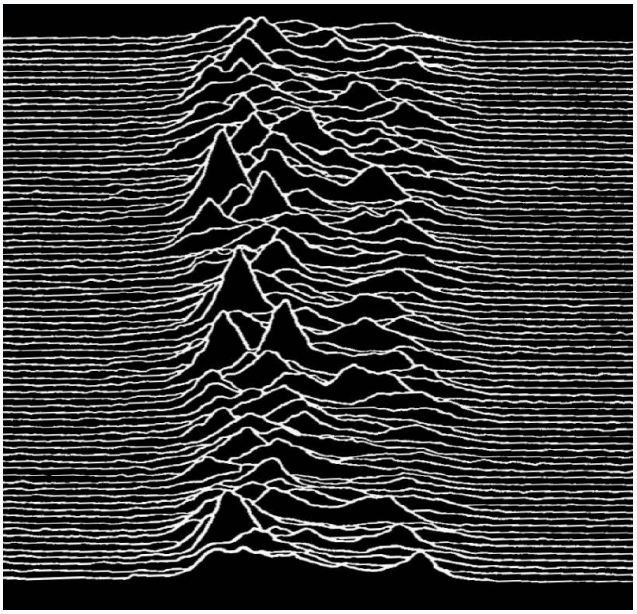
Techniques for Digital Image Forensics and Counter-Forensics



Andrea Costanzo Piccinnano

Ph.D Thesis in Information Engineering

University of Siena

UNIVERSITÀ DEGLI STUDI DI SIENA

Facoltà di Ingegneria

Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche

UNIVERSITÀ
DI SIENA
1240

Techniques for Digital Image Forensics and Counter-Forensics

Andrea Costanzo Piccinnano

Ph.D. Course on Information Engineering
XXV Cycle, 2009–2012

Supervisor

Prof. Mauro Barni

Examination Commitee

Prof. Sebastiano Battiato

Prof. Chang-Tsun Li

Prof. Marco Maggini

Thesis reviewers

Prof. Sebastiano Battiato

Prof. Chang-Tsun Li

Siena, February 20, 2014

Contents

List of Figures

List of Tables

You fling this book on the floor, you would hurl it out of the window, even out of the closed window, through the slats of the Venetian blinds; let them shred its incongruous quires, let sentences, words, morphemes, phonemes gush forth, beyond recomposition into discourse; through the panes, and if they are of unbreakable glass so much better, hurl the book and reduce it to photons, undulatory vibrations, polarised spectra; through the wall, let the book crumble into molecules and atoms passing between atom and atom of the reinforced concrete, breaking up into electrons, neutrons, neutrinos, elementary particles more and more minute: through the telephone wires, let it be reduced to electronic impulses, into flow of formation, shaken by redundancies and noises, and let it be degraded into a swirling entropy. You would like to throw it out of the house, out of the block, beyond the neighborhood, beyond the city limits, beyond the state confines, beyond the regional administration, beyond the national community, beyond the Common Market, beyond Western culture, beyond the continental shelf, beyond the atmosphere, the biosphere, the stratosphere, the field of gravity, the solar system, the galaxy, the cumulus of galaxies, to succeed in hurling it beyond the point the galaxies have reached in their expansion, where space-time has not yet arrived, where it would be received by nonbeing, or, rather, the not-being which has never been and will never be, to be lost in the most absolutely guaranteed undeniable negativity. Merely what it deserves, neither more nor less.

If on a winter's night a traveler
Italo Calvino

Chapter 1

Introduction

Begin at the beginning and go on
till you come to the end; then stop.

Alice in Wonderland
Lewis Carroll

AFTER A first decade of studies, Digital Image Forensic is not anymore just a promising field of research but rather a mature discipline that can count on rigorous formalisations and on a very large selection of algorithms to gather information on the *history* of an image, i.e. on its origin, the processing it has undergone and its authenticity. The need for such tools is the natural consequence of the widespread diffusion of digital content, which anyone can modify, manipulate and distribute almost effortlessly. It is not surprising, then, that restoring the credibility of digital content has become a task of paramount importance.

With this thesis we contribute to the above mission by addressing three open issues in Image Forensics. In the sequel we briefly introduce the motivations that pushed us to tackle with such issues, to each of which we dedicate a part of the thesis.

In the first part of the thesis, we devise a forensic tool to study the history of large groups of images sharing the same visual content, with the objective to determine their net of parent-child relationships. The idea underlying our study originates from simple web searches of popular images depicting, for example, famous paintings or events with a relevant social impact. Web engines usually find several similar or apparently identical copies of the queried image; even when the content is exactly the same, colors, brightness, contrast

or image dimensions are different. Clearly, not all the images are original: it is very unlikely that hundreds, if not thousands, of people photographed a lonely bear near the North Pole! More likely, the original image was copied or scanned, edited and redistributed multiple times. Our idea is to exploit forensic algorithms, so far used on single images, to uncover these links between images. Our studies brought to a rigorous formalisation of the problem (to the best of our knowledge, the only one when our works were published the first time) and its practical implementation, that we call Dependency Explorer Framework, which served as an inspiration for recent works of other research groups [Dias et al., 2012].

Several numerical results extensively reported in the thesis support the trustworthiness of our method. However, we value as much as such results a fact occurred during the writing of this thesis. Since the beginning of our research, we struggled with the unavoidable major drawback behind its practical application: because rarely image users report the source of the image they (eventually) modified and published, we do not know the real relationships linking the data we collect. More often, we know the author of the original image (a famous painter or photographer, for example) but we do not possess the archetype. As a consequence, it took us a lot of detective work to evaluate by visual inspection the plausibility of our findings. While writing this thesis, we extended the original results with an in-depth analysis of a large Web set depicting a young polar bear in distress. The original picture photographed by Arne Nævra played a key role in the awareness campaign against global warming, thus explaining the widespread diffusion of its near-duplicates. Because we ignored the real relationships, we had to make a lot of assumptions on the links between the images. Then, we tried to involve the photographer, who was really interested in our studies. From his words we understood the seriousness of problems such as copyright protection and near-duplicate tracing from the point of view of a professional photographer; last but not least, he provided us with the archetype. We included it in the set of images used for our tests and we ran again all the algorithms; the results confirmed that the majority of our conclusions were indeed correct and we were glad not to have to rewrite the entire section!

Somehow dually, in the second part of the thesis we show how to use multiple forensic tools at the same time to study a single image. The objective is to make possible to fuse the output of multiple tools by coping with incompatibilities and unreliability. The rationale behind our study is once again inspired by the real-world experience of any image user: to create forgeries of convincing quality, the counterfeiter must resort to any kind of processing tools provided by imaging software. If that is the case, why resort to a single forensic algorithm to detect a manipulation? We believe that detection accuracy could greatly benefit from the *fusion* of the information coming from multiple detectors. To some extent, forensic research has already attempted to integrate different detectors by concatenating the features they provide before feeding them to a classifier. However, we try something yet unexplored in the field of Image Forensics: we let each tool decide independently on the authenticity of an image and then we *reach a consensus* among them. While doing so, we have to tackle with the uncertainty afflicting the tools, which, similarly to any other process, are subject to error or noise. We devise a general framework based on Fuzzy Theory and we apply it to a typical image forensic scenario of splicing detection. Experimental results confirm that our framework, as well as its twin system developed by Fontani *et al.* [Fontani et al., 2011, 2013], outperforms more traditional approaches.

In the third part of the thesis we join the enemy. We dedicate ourselves to Counter-forensics, the art of misleading the forensic analysis by taking advantage of its weaknesses. Despite the relative youth of the discipline, literature already offers several interesting methods to remove relevant footprints like the artefacts introduced by lossy compression, CFA interpolation or resampling, thus making possible to impair source or forgery detection based on such traces. However, we noticed that no one so far attempted to counter a particular category of forensic algorithms, that is those based on robust salient point detectors. We then investigate the weaknesses of the methods based on the most popular of such detectors, the Scale Invariant Feature Transform (SIFT). Even though challenging SIFT-based applications proved to be no small feat, the results of our work are promising: apart from the thrill of assuming the role of the villain for a change, we were able to point out that it

is possible to remove SIFT features without significantly impacting the quality of the forged image. Such a conclusion is quite alarming, considering the popularity of SIFT-based application.

We have organised this part of the thesis like a duel of wits between a forensic analyst and an adversary of equal knowledge. The former wants to detect something (a forgery, the origin of an image, etc.) by relying on SIFT features, the latter wants to evade such detection. We let the adversary move first by devising a SIFT keypoint removal attack whose novelty resides on the hypotheses that there exist multiple classes of keypoints and that each class must be removed by means of specifically tailored attacks. Even though so far literature has assumed the opposite (e.g. [Hsu et al., 2009; Do et al., 2010a]), experimental results confirm the goodness of our intuition. As forensic analysts, we notice that removal attacks tend to leave peculiar footprints in the form of textured regions suspiciously deprived of SIFT keypoints. Based on such observation, we develop three novel keypoint removal detectors forcing the adversary to limit the amount of removed keypoints to preserve the undetectability of the forgery. We then provide the adversary with a new tool to inject *fake* keypoints, which were not in the authentic image but are regularly detected by SIFT following the manipulations. Such keypoints are meant to hide the preceding keypoint removal without interfering on the results it attained, thus preventing the detection by means of the removal detectors. It turns out that two detectors out of three are quite robust to this latter attack, thus proclaiming the (temporary?) victory of the analyst.

1.1 Overview

This thesis is subdivided in three parts. The first part focuses on the forensic analysis of the history of groups of near-duplicate images. Before that, in Chapter 2 we provide some introductory information on Digital Image Forensics; a reader who is already familiar with such concepts can skip this chapter and start reading from Chapter 3, where we define the problem of finding dependencies within a set of images with similar content. A practical implementation of such principles, which we call Dependency Explorer Framework, is introduced and experimentally validated in real Web scenarios in Chapter 4.

The second part investigates the problem of merging heterogenous data provided by forensic detectors while dealing with the incomplete, noisy or not fully reliable information they may provide. This is a typical decision fusion task, whose basic principles are briefly described in Chapter 5. In Chapter 6 we formalise our solution based on Fuzzy Theory, which we put in practice in Chapter 7, by fusing the outputs of five popular forensic algorithms revealing the presence of cut & paste forgeries based on JPEG artefacts.

The third part is dedicated to Counter-forensics. In particular, we propose new techniques to impair SIFT-based detection and then we develop possible countermeasures to restore the credibility of the forensic analysis. In Chapter 8 we introduce the counter-forensic problem. In Chapter 9 we analyse in detail the SIFT algorithm, in order to fully understand the working principles of the keypoint removal attack proposed in Chapter 10. In Chapter 11 we successfully apply the removal attack to impairing a state-of-the-art SIFT-based copy-move detector. In Chapter 12 we investigate the visibility of the removal attack; in particular, we individuate two peculiar footprints on which we base three novel detectors revealing keypoint removal. In Chapter 13 we study the dual problem of keypoint removal, i.e. the injection of fake keypoints with the goal to mask keypoint removal and we assess its impact on removal detection. We consider again the SIFT-based copy-move detection scenario and we apply all the tools developed so far, that is keypoint removal, keypoint injection and keypoint removal detection. Chapter 14 concludes the thesis by outlining some possible directions for future research.

1.2 Contributions

In the following we summarise the contributions of this thesis.

- A theoretical framework (Chapter 3) to analyse the relationships between large groups of near-duplicate images and a system putting the framework in practice (Chapter 4). These studies led to publication 2 in the list of Sec. 1.4.

- A theoretical framework for the exploitation of multiple forensic tools by means of the fusion of their decision scores (Chapter 6). Such framework

allows to overcome several problems such as heterogenous, mutually exclusive, incomplete or noisy outputs by relying on Fuzzy Theory. A practical implementation (Chapter 7) is applied to a cut & paste forgery detection scenario with results superior to those ensured by traditional methods. These studies led to publications 3–6 of the list of Sec. 1.4.

- A new method to remove SIFT keypoints based on the hypothesis that not all SIFT keypoints have the same properties (Chapter 10). Following the classification of its neighbourhood, each keypoint is removed by means of a procedure specifically tailored to the class. This solution outperforms state-of-the-art class-unaware keypoint removal attacks and provides a good trade-off between removal effectiveness and the impact on the counterfeited image's quality.

- A practical counter-forensic application of the removal attack whereby a state-of-the-art SIFT-based copy-move detector is successfully bypassed (Chapter 11).

- Three forensic algorithms revealing both global or local, class-aware or unaware keypoint removal attacks (Chapter 12). To the best of our knowledge, before the introduction of our detectors the only existing alternative was visual inspection, which is unreliable at best, given the typical sparsity and quantity of SIFT keypoints in natural images.

- A set of new counter-forensic methods to introduce fake keypoints into images (Chapter 13). This thesis represents the first systematic study on keypoint injection, which so far has been often considered more like a side effect of keypoint removal rather than a powerful tool to counter SIFT matching-based applications and removal detectors. By resorting to these attacks, it is possible to reduce the performance of the proposed keypoint removal detectors. The above four contributions led to publications 7–12 of the list of Sec. 1.4.

In order to ensure the reproducibility of our experimental results, we always provide detailed pointers to the software we used, the parameter settings and the image data sets; furthermore, we also provide the pseudo-code of our

algorithms wherever this feels necessary.

1.3 Activity within FET European projects

The activity of the thesis has been mainly carried out within the LivingKnowledge and REWIND European projects, funded under the Future and Emerging Technologies Open scheme.[1] The thesis represents part of the activity conducted by CNIT[2] (National Inter-university Consortium for Telecommunications), namely by the Research Unit of the University of Siena, led by Prof. Mauro Barni, and within the informal VIPP (Visual Information Processing and Protection) research group.[3]

The LivingKnowledge project[4] (2009-2012) revolved around the concepts of diversity and time, and their impact on the opinion-forming process on the Web. The project envisaged a future where search and navigation engines automatically classify opinions and bias (e.g. about social or political matters), to produce more insightful, better organised, easier-to-understand output. LivingKnowledge offered us the opportunity to address the image forensic problem from an unusual perspective in which the rigorous data provided by the detectors were supported by higher level semantic information. For example, the cooperation of splicing detectors and automatic text processing tools allowed to understand whether a forged image was used to convey a particular message. Moreover, contrast manipulation detectors were employed to understand whether Web images were brightened or darkened to suggest a particular emotion. In accordance with the LivingKnowledge's vision, the Dependency Explorer Framework presented in the first part of the thesis was originally imagined as a way to exploit the relationships between near-duplicates to understand the role of different websites (and the groups behind them) in the formation of opinions on the Web; by doing so, one could identify opinion leaders, common feelings about specific events and the preferred sources of information in a given temporal or geographical context. In

[1] Respectively under grants no. 231126 and no. 268478.

[2] http://www.cnit.it.

[3] http://clem.dii.unisi.it/~vipp/.

[4] http://livingknowledge.europarchive.org.

addition, knowing how a few source images have evolved into a large set of derived pictures, could allow to reconstruct how the usage of the information contained in the original images has evolved in time and space; by doing so, we can identify, for instance, how these images have been used by groups of people with different opinions and cultures.

The REWIND (REVerse engineering of audio-VIdeo coNtent Data) project[5] aims to develop a comprehensive set of new mathematical models and techniques that address the forensic footprint detection problem independently of the number of processing stages the data goes through and the media involved. To this end, three main goals are pursued: the definition of an universal footprints theory; the development and the composition of tools for reverse-engineering the history of multimedia objects (audio, image, video); and the study of attacker-aware detectors. This thesis contributes to the latter goal by exposing the weaknesses of SIFT-based forensic detectors in such a way to develop secure countermeasures that are aware of adversarial processing.

1.4 List of publications

The activity of the thesis resulted into the following publications.

2010:

1. M. Barni, **A. Costanzo**, and L. Sabatini, *Identification of cut & paste tampering by means of double-JPEG detection and image segmentation.* In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), pages 1687–1690, IEEE, 2010.

2. A. De Rosa, F. Uccheddu, **A. Costanzo**, A. Piva, and M Barni, *Exploring image dependencies: a new challenge in image forensics.* In IS&T/SPIE Electronic Imaging, pages 75410X–75410X, International Society for Optics and Photonics, 2010.

[5] http://www.rewindproject.eu.

2012:

3. M. Barni and **A. Costanzo**, *Dealing with uncertainty in image forensics: A fuzzy approach.* In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1753–1756, IEEE, 2012.

4. M. Fontani, **A. Costanzo**, M. Barni, T. Bianchi, A. De Rosa and A. Piva, *Two decision fusion frameworks for image forensics.* In Annual meeting of the Telecommunications and Information Technology Group (GTTI), 2012.

5. M. Barni and **A. Costanzo**, *A fuzzy approach to deal with uncertainty in image forensics.* In Signal Processing: Image Communication, Elsevier, 2012.

6. R. Caldelli, I. Amerini, L. Ballan, G. Serra, M. Barni and **A. Costanzo**, *On the effectiveness of local warping against SIFT-based copy-move detection.* In Proceedings of the International Symposium on Communications, Control and Signal Processing (ISCCSP), 2012.

2013:

7. I. Amerini, M. Barni, R. Caldelli, and **A. Costanzo**, *Counter-forensics of SIFT-based copy-move detection by means of keypoint classification.* In Journal on Image and Video Processing, 2013(1):18, EURASIP, 2013.

8. I. Amerini, M. Barni, R. Caldelli, and **A. Costanzo**, *SIFT keypoint removal and injection for countering matching-based Image Forensics.* In Proceedings of the 1st ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec), ACM, 2013.

9. I. Amerini, M. Barni, R. Caldelli, and **A. Costanzo**, *Removal and injection of keypoints for SIFT-based copy-move counter-forensics.* In Special Issue on Revised Selected Papers of ACM IH&MMS 2013, Journal on Information Security, 2013:8, EURASIP, 2013.

10. A. Melloni, P. Bestagini, **A. Costanzo**, M. Barni, M. Tagliasacchi, and S. Tubaro, *Attacking image classification based on Bag-of-Visual-Words.*

In Proceedings of the International Workshop on Information Forensics and Security (WIFS), 2013.

2014:

11. I. Amerini, F. Battisti, R. Caldelli, M. Carli and **A. Costanzo**, *Exploiting perceptual quality issues in countering SIFT-based forensic methods.* [Submitted to] IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014

12. **A. Costanzo**, I. Amerini, R. Caldelli and M. Barni, *Forensic analysis of SIFT keypoint removal and injection.* [Submitted to] IEEE Transactions on Information Forensics and Security (TIFS)

1.5 Acknowledgements

I would like to express my gratitude, first and foremost, to my advisor Prof. Mauro Barni for the continuous support of my research, for his motivation, enthusiasm and knowledge. His guidance helped me in all the time of research and writing of this thesis. Besides my advisor, I wish to thank my thesis reviewers and members of the committee, Prof. Sebastiano Battiato and Prof. Chang-Tsun Li, for their insightful comments and suggestions. Last but not least, a special acknowledgment goes to the LivingKnowledge and REWIND European projects for their financial support during the three years of the Ph.D. course and beyond, and for making possible the fruitful collaboration with very professional and friendly researchers. Without their contributions, this thesis and the works on which it is based would have not been possible.

Part I

Digital Image Forensics of near-duplicate images

Digital Image Forensics is a relatively new research field aiming at gathering information on the history of an image in such a way that its authenticity can be evaluated. Image Forensics is based on the observation that any processing carried out during any stage of an image's life cycle leaves specific subtle traces, whose presence can be exploited to expose the corresponding manipulation. By doing so, it is possible to verify the history of an image blindly, i.e. without the help of the original image prior to the manipulations.

Though the current state of the art of Image Forensics permits to acquire very interesting information, all the instruments developed so far focus on the analysis of single images. In this part of the thesis we push the analysis forward by proposing a new approach considering groups of images instead of single images. The idea is to discover parent-child relationships among a group of images representing similar or equal contents. Given the strong effect that images posted on the Web have on opinions and bias in the networked age we live in, such an analysis could be extremely useful for understanding the role of pictures in the opinion forming process. Among more traditional applications, we mention copyright protection and image clustering. We formalise the concept of image dependency and we describe a system, named Dependencies Explorer Framework, putting it in practice.

Chapter 2
Introduction to Digital Image Forensics

"I'm not upset that you lied to me,
I'm upset that from now on I can't believe you"

Friedrich Nietzsche

IT'S ALL so easy with Photoshop. Nowadays, with imaging software so widely available, the manipulation of digital images is not a matter for experts only. It is a widespread belief that analog photos preceding the advent of the digital era were trustworthy. This, however, is not true as they were simply harder to counterfeit and required deep knowledge of the acquisition process and great expertise in the darkroom. In 1825, Nicéphore Niépce made an image that is recognised as the first surviving photograph of History. In 1822, a failed attempt of duplicating one of his previous experiments costed him the loss of an even older photograph and three years of work to reproduce it successfully. Not long after, in the decade 1860-1870, expert photographers already learned how to create some of the oldest examples of forgeries the world is aware of. The print of Fig. 2.1 (a) documents one of such convincing forgeries: only the face of General Grant posing in front of his encampment during the American Civil War is genuine; the man on the horse was another person and even the background did not depict the real encampment.

Since those pioneering times photographic techniques evolved side by side with the expertise in manipulating them. Recent History is full of similar cases of forgeries as every leader, influential politician or dictator systematically added or removed from official photos collaborators raising or falling into disgrace. Today, things are not different. Modern digital imaging allows to create extremely convincing fakes on-the-go, even with devices with relatively low computational power such as smartphones and tablets. Content

can be easily faked in diverse contexts: a fitter fashion model, a different hour of the day, a cheating companion, an army parading non-existent missiles, a politician interacting with the wrong people.

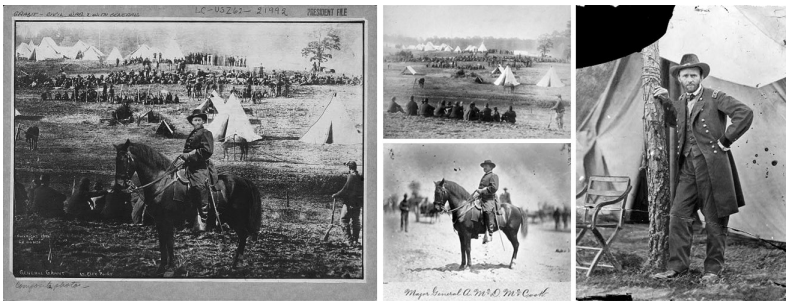
While there is little harm besides gossip in retouching an unwanted belly or an incipient baldness, the simplicity of counterfeiting is a serious issue when it is exploited to convey social, political or military messages. What could be the reaction of the opposite faction to a forged image exhibiting one missile too much, if the image is believed to be authentic? What if a jury has to decide on innocence or guilt of a defendant based on an image that could be counterfeited or whose origin is uncertain? There is more. Images are tampered with in academic papers to display better results to boost chances of publication or funding (Fig. 2.1 (c)). Similarly, medical images can be altered to hide or pretend a pathology for insurance purposes (Fig. 2.1 (b)).

To understand whether an image is authentic or not by carefully looking at it is no more a viable option, as sometimes legitimate photos may appear as fakes at first glance (and vice-versa).[1] As a consequence, more sophisticated methods to restore the credibility of digital images are needed.

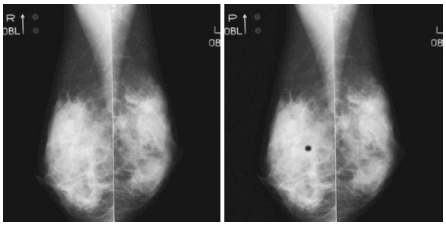
Digital Image Forensics [Redi et al., 2011; Piva, 2013] is a relatively new research field aiming at gathering information on the history of an image in such a way that its veracity can be evaluated. In particular, this task is fulfilled by providing an answer to questions like the following: was an image acquired by the device it is claimed to be captured with? Is the scene that is being depicted still the original one? One of the strengths of Digital Image Forensics consists in its blind approach these kinds of verification, in the sense that it is assumed that the original image prior to the manipulations is not available for comparison. As a matter of fact, this is the case in the majority of the real-world scenarios. Image Forensics is based on the observation that any processing carried out during any stage of an image's life cycle leaves specific subtle traces whose presence can be exploited by forensic analysts to expose the corresponding manipulation. It goes without saying that most of

[1] As a proof, consider the following challenges: Autodesk's *Fake or Foto*, available at http://area.autodesk.com/fakeorfoto/, on discriminating between natural and computer generated images; Nova's *Fake or Real*, available at http://www.pbs.org/wgbh/nova/sciencenow/0301/03-fakeorreal.html, on distinguishing between authentic images and photomontages.

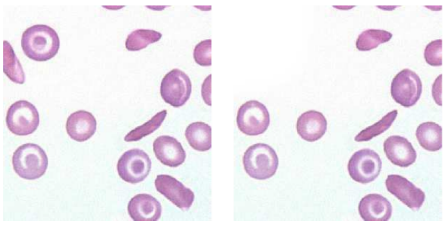
the times analysing the full history of an image is no small feat, especially if one considers that every processing carried out in the image tends to weaken the traces left by previous modifications.



(a)



(b)



(c)

(d)

(e)

(f)

Figure 2.1: *Examples of image tampering thoroughout History: (a) composite (1865 ca); (b) forged pathology (right); (c) forged academic results (right); (d)-(f) fake magazine covers (respectively: spliced face, cloned planes, spliced hood and gun). For more examples check* http://www.fourandsix.com/photo-tampering-history.

2.1 Digital Image Forensics in practice

Image Forensics inherits part of its challenges and methods from Digital Steganography and Digital Watermarking. Both these sciences conceal information in images in such a way to protect the rightful owner or to verify integrity and authenticity.

Steganography is the science of communicating securely in a completely undetectable manner [Chandramouli et al., 2004]. In practice, the message that needs to be conveyed is imperceptibly embedded into an image (referred to as carrier or cover) whose content is often not relevant. Digital watermarking [Barni and Bartolini, 2004] is the science of embedding a watermark (i.e. a kind of signature) revealing the owner or the integrity of the multimedia object. A watermark can be robust or fragile: in the former case the watermark is designed so that it survives to certain processing tools, while in the latter case the watermark is destroyed as soon as the image is altered. Typical applications are respectively copyright protection and integrity verification.

The main difference between Image Forensics and the two above sciences is that the former does not require the original object nor additional information about it or the acquisition device, thus carrying out a *blind* analysis. Furthermore, such an analysis is also *passive*, in the sense that contrarily to watermarking techniques no specific hardware (like trusted cameras) must be used to make the techniques practically viable.

2.1.1 The image generation process

Forensic techniques gather information on the history of an image by looking at peculiar footprints left by different processing. According to the categorisation in [Piva, 2013], there are various typologies of footprints, which are related to the stages defining the life cycle of every image as shown in Fig. 2.2.

When a real-world scene is photographed, a combination of concave and convex optical lenses conveys the light towards a set of optical filters allowing only the visible part of the spectrum to pass and reducing the aliasing effect. Upon filtering, the light is directed towards the imaging sensor, that is a matrix of photodiode elements (pixels). Each pixel converts the light entering it into a voltage that is proportional to the intensity of the light. At this point, the produced digital signal does not convey color information, because sensors react exclusively to brightness. Therefore, a filter named Color Filter Array (CFA) is placed in front of the sensor in order to capture the color. CFA filters are designed in such a way that only a particular color (red, green or blue) rather than all three is captured by each pixel. This design does not depend on technological limitations but rather on the necessity of reducing manufacturing costs. The effect of CFA is to create a single mosaic channel of red, green and blue pixels that needs to be converted to the three-channel output by estimating the missing pixel values based on their sensed neighbours (demosaicing). Once the RGB is generated, commercial cameras usually perform a number of processing to enhance its quality and to reduce its size for storage purposes, commonly by means of JPEG compression [Battiato et al., 2010]. The stored image can then undergo additional processing aimed at further improving its quality or at manipulating its semantic meaning.

By following the above process, the footprints that can be analysed with forensic techniques are divided into *acquisition*, *coding* and *processing*. Depending on the purpose of each technique, each category of features can be used to perform the two main tasks of Image Forensics, that is source device identification and forgery detection. Although a comprehensive survey on the state-of-the-art of forensic algorithms is not the aim of this chapter, some examples of each category are provided in the next sections without going into the details.

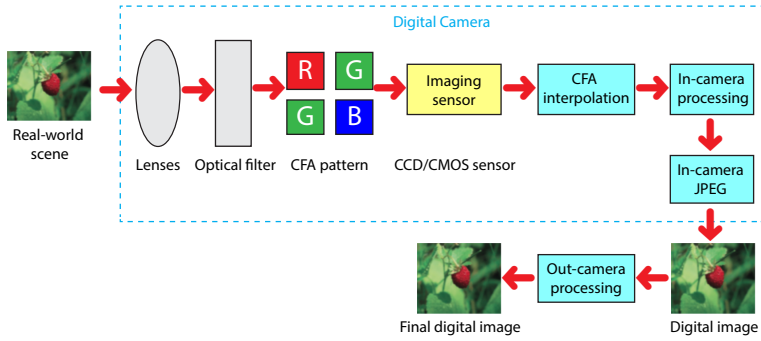


Figure 2.2: *Life cycle of a digital image as depicted in* [Piva, 2013].

2.1.2 Acquisition footprints

Every stage of the acquisition process leaves traces in a digital image. Even though the pipeline in Fig. 2.2 is common to the majority of cameras, the in-camera hardware and software generally vary from manufacturer to manufacturer. As a consequence, it is possible to discriminate between the kind of device (e.g. camera, mobile, tablet), the brand and the model by exploiting acquisition footprints. This task can be seen as a classification problem whereby an image is assigned to a certain class identified by the acquisition features. Moreover, it is also possible to detect forgeries by looking for inconsistencies between acquisition features in different regions within the same image. In this way one can reveal, for example, splicing of content originated by different camera models.

The first footprint introduced into an image by any camera is due to the aberration of lenses, i.e. a distortion depending on the striking light and on the geometry of lenses. There exist multiple aberrations, each of which has distinctive features; for example, chromatic aberration is responsible for colored edges along boundaries separating dark and bright parts of the image. Several techniques leverage on such artefacts for source identification [San Choi et al., 2006; Van et al., 2007; Dirik et al., 2008] and for tampering detection [Johnson and Farid, 2006; Yerushalmy and Hel-Or, 2011].

The subsequent stage in which the sensor captures the light and converts it into digital values also leaves peculiar cues, the most important of which are

related to the Photo Response Non Uniformity (PRNU) of the sensor and to CFA's pattern and interpolation. The PRNU is a noise pattern generated by the variation in pixel response over the CCD under illumination. Since PRNU is caused by the physical properties of the sensor itself, it is almost impossible to eliminate it completely and is usually considered to be a normal characteristic of the sensor. As a consequence, PRNU can be used to identify different cameras according to the technical imperfections of their sensors [Lukas et al., 2006; Chen et al., 2008; Li, 2010].

Both the CFA pattern and the interpolation algorithm converting the sensed matrix into the three-channel image are proprietary to the manufacturers and allow to gather information on the acquisition device [Bayram et al., 2005; Popescu and Farid, 2005; Celiktutan et al., 2006].

Before concluding this section, it is worth noting that it is also possible to detect two other sources for digital images: scanners and computer-graphics (CG). In the former case, even though the pipeline generating the image is obviously different from the one in Fig. 2.2, acquisition fingerprints are still present; for example, scanner's sensor noise can be exploited similarly to the PRNU [Gloe et al., 2007a; Gou et al., 2007; Khanna et al., 2007b].

CG images are not acquired but generated by processing functions, therefore there is no underlying noise and this fact can be exploited for investigation [Dehnie et al., 2006]. Moreover, the fact that computer graphics tools simplify the complexity of real-world scenes by approximating geometry, surfaces and lightning sources permits to identify features characterising CG images [Ng et al., 2005; Chen et al., 2007].

2.1.3 Coding footprints

Most of digital cameras compress the images with high quality JPEG for efficient storage purposes. Due to the widespread diffusion of the format, a big effort has been dedicated by the research community to the study of the compression history of an image. JPEG, in fact, leaves a number of traces including blocking artefacts caused by the underlying block-wise approach and quantisation artefacts, with which one can estimate compression parameters like quality factor [Fan and de Queiroz, 2000] or quantisation tables [Luo et al., 2010]. Since manufacturers and imaging software developers generally

set different compression parameters for their products, quantisation tables can be also used for source and forgery detection [Farid, 2008].

It is also possible to understand whether an image has been compressed multiple times. When an image is recompressed, the new 8×8 JPEG grid is superposed to the already existing one in such a way that they are either aligned or misaligned; forensic algorithms have then been devised for the aligned case [Farid, 2009a; Lin et al., 2009; Bianchi et al., 2011] and for the misaligned case [Luo et al., 2007; Bianchi and Piva, 2011]. These techniques can also be employed for forgery detection by searching for local inconsistencies in the JPEG grid of an image.

2.1.4 Editing footprints

Following the acquisition and the in-camera enhancement stages, an image can still undergo a number of diverse modifications. In general, an editing tool can be used in a *legitimate* way to enhance the quality of an image or in an *illegitimate* way to alter its semantic content. The tools serving the former purpose are discussed in this section, while those serving the latter are outlined in the next section. Unfortunately, the above distinction is not always neat, as processing often considered harmless can be used with malicious intents; for example, filtering or resampling can hide traces of a previous tampering, or color manipulations can convey a different message with respect to that of the authentic image, as in the famous case of a water puddle recolored so that it resembled blood following a terrorist attack in Luxor, Egypt.[2]

One of the most common editing operators is resampling, which is performed when geometric transformations like rotation and resizing are applied to an image. Following resampling, certain pixel values are a linear combination of their neighbours and the correlation between them in the resampled image manifests itself with periodical artefacts [Popescu and Farid, 2005; Kirchner, 2008; Mahdian and Saic, 2008].

Contrast enhancement is a very common manipulation to increase the perceived quality of images. Usually, such enhancement is carried out by means of histogram equalisation, whereby the intensity values of pixels in

[2] http://www.fourandsix.com/photo-tampering-history/after-58-tourists-were-killed-in-a-terrorist-attack-at-the.html.

the input image are remapped in such a way that the output image has an uniform distribution of intensities. Contrast enhancement can be also used to change the semantics of an image; for example, an image can be darkened to convey a sensation of menace or discomfort. Global contrast enhancement can be detected by means of the technique presented in [Stamm and Liu, 2008]; such algorithm is based on the observation that altering contrast causes the introduction into an image's histogram of artefacts (peaks and gaps) that are not found in unaltered images.

Median filtering has several applications in image enhancement, including denoising and smoothing but it can also be used to conceal traces of a precedent processing. Currently there are three algorithms detecting median filtering based on the following observations: the probability of two adjacent pixels being equal is greatly increased by median filtering [Kirchner and Fridrich, 2010]; the difference between two adjacent pixels is exactly zero [Cao et al., 2010b]; and median filtering's block-wise approach introduces correlation between blocks [Yuan, 2011].

Finally, JPEG re-compression following the above manipulations is a powerful instrument when it comes to hiding their traces, as quantisation may reduce the distinctiveness of certain features without raising suspicion and without significantly degrading the perceptual quality of the image.

2.1.5 Tampering detection

The two most common ways to manipulate the semantic content of an image are splicing (or *cut & paste*) and cloning (or *copy-move*) [Farid, 2009b]. The purpose of both such forgeries can either be that of hiding authentic content or introducing fake content into the image. The difference between cut & paste and copy-move is that the former requires a second image other than the one being tampered with, as described in the sequel.

Cut & paste forgery

A cut & paste forgery is carried out by taking a region from a source image and pasting it into a target image, thus producing a fake image. Such a procedure is very likely to introduce inconsistencies between the characteristics

of the pasted region and of the rest of the image. There exist multiple methods to reveal cut & paste, some of which have been hinted throughout the previous sections. Given the widespread diffusion of the format, JPEG-based techniques like those in Sec. 2.1.3 are very popular. If the tampered image is compressed with JPEG, then grid (mis)alignment between the region and the rest of the image can be exploited to detect the tampering [Barni et al., 2010]. Similarly, differences in the estimated parameters of lens aberration, PRNU or CFA interpolation can reveal portions of an image that have been acquired by means of a different device. Moreover, when a cut & paste is carried out most likely the pasted objects are framed at different scales with respect to the target image; hence resampling is necessary to render the forgery visually convincing. Therefore, traces of resampling within the suspicious regions can be used to reveal the forgery [Popescu and Farid, 2005].

The problem with all these detection techniques is the lack of robustness to common processing such as resampling or compression, which erase the traces that the forensic algorithms are looking for. To overcome these limitations, novel approaches based on the study of geometrical and physical inconsistencies have been recently proposed. All these approaches rely on the observation that it is very difficult to create a forgery whereby the geometry, lights and shadows of the fake content are coherent. Among the most popular algorithms we mention those in [Johnson and Farid, 2008; Conotter et al., 2010; Yao et al., 2012] for geometrical inconsistencies and those in [Johnson and Farid, 2005, 2007; Zhang et al., 2009] for physical inconsistencies.

Copy-move forgery

A copy-move forgery is obtained by copying and pasting a portion of an image once or more times elsewhere into the same image. The duplicated region can be manipulated with an arbitrary number of processing tools to seamlessly blend the forged and the original contents. Several forensic techniques revealing such kind of forgery by looking for very similar (or identical) portions of the image have been proposed: fifteen among the most popular are thoroughly discussed and compared in [Christlein et al., 2012]. Earlier techniques were based on an exhaustive block-wise analysis followed by a similarity ordering of the features extracted by each block and a final matching [Fridrich

et al., 2003a; Popescu and Farid, 2004]. The major drawback of block-wise approaches is computational complexity, which rapidly renders the problem intractable as the image size grows; additionally, such methods are usually not robust against geometric manipulations and compression, as observed in [Nguyen and Katzenbeisser, 2011]. For these reasons, most recent methods rely on the matching of robust local descriptors (e.g. SIFT, SURF), which are computationally light and robust (or invariant) to affine transformations [Pan and Lyu, 2010; Amerini et al., 2011]. For a detailed analysis of SIFT-based copy-move detection we refer to Chapter 11.

Other forgeries

Seam carving is a recently proposed method to modify the size of an image in such a way that its content is preserved [Avidan and Shamir, 2007]. Because this goal is achieved by altering only the least noticeable portions of the image, the technique is often referred to as content-aware resizing. The algorithm relies on the computation of seams, that is paths of pixels traversing the image from left to right or from top to bottom. Each path is composed by one pixel per column (or row) chosen by minimising an energy functional in such a way that pixels belong to irrelevant content. Resizing can be obtained either by deleting a seam (downscaling) or by adding a seam (upscaling). Seam carving can also be used to remove a specific image region by iteratively deleting all the seams traversing it. Currently, two algorithms reveal this kind of forgery [Sarkar et al., 2009; Fillion and Sharma, 2010] by means of a seam/non seam classification based on peculiar traces introduced by seam carving.

Digital image inpainting is a technique allowing the reconstruction of lost or deteriorated parts of images (e.g. scratches in a scanned photograph or undesired text overlays) or removing unpleasant objects from a scene [Bertalmio et al., 2000; Shih and Chang, 2005]. Since inpainting is capable of removing large objects with acceptable quality degradation, it can be used to alter the semantics of an image. A detector localising regions inpainted with the method in [Criminisi et al., 2003] has been presented in [Wu et al., 2009]. Such algorithm leverages on the filling scheme of the inpainting method, that introduces similar blocks whose difference is very low or null in the suspicious areas. The algorithm has been applied in [Das et al., 2012] to detect inpainting

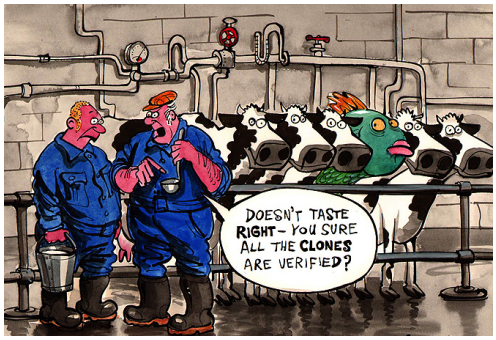
in videos treated as sequences of still images.

2.2 Digital Image Counter-forensics

So far, little or no importance has been given until recently to the study of countermeasures specifically devised to bypass the analysis of the forensic algorithms. Anyone with such a purpose is commonly referred to as an *adversary*, i.e. a party who has the same knowledge on signal processing as the forensic analyst and some reasons to mislead a specific image forensic investigation. A goal like this can be pursued, for example, by hiding, removing or falsifying traces of an illicit processing or manipulation, so that the altered image appears authentic. In other words, the adversary does not limit herself anymore to manipulate an image but also wants that manipulation to be *undetectable*. All the solutions in this sense fall into a discipline called *Counter-forensics* (alternatively anti-forensics) [Böhme and Kirchner, 2012]. Most of the times, similarly to the algorithms they aim to fool, counter-forensic techniques are not perfect and leave traces on their own, which can be exploited by the forensic analyst to detect the adversary's misdoing. Studying the interplay between these two parties can help exposing the limitations of current forensic tools and push towards devising improved or new tools. Nowadays, by relying on the available counter-forensic techniques, we can hide traces of JPEG compression [Stamm et al., 2010a], resampling [Kirchner and Bohme, 2008], filtering [Fontani and Barni, 2012] and histogram manipulations [Barni et al., 2012]. For a possible formalisation of the problem and a brief survey of the state-of-the-art the reader is referred to Chapter 8.

Chapter 3

The image dependency problem



THOUGH THE current state-of-the-art of Image Forensics permits to gather very interesting information about the *history* of an image, the majority of instruments developed so far focus on the analysis of single images. In several applications, however, the investigation of image dependencies, i.e. the relationships between a group of images, may be of similar or even greater importance. For instance, knowing how a set of images are related one to each other could allow the clustering of images originating from the same root; in this way, it could be possible to discover that several images regarding a particular event have been actually produced from a limited set of source images. Such an information could then be used to understand the role of different web sites (and the groups behind them) in the formation of opinions on the web, permitting to identify opinion leaders, common feelings about specific events, and the preferred sources of information in a given temporal or geographical context. In other situations, knowing how a few source images have evolved into a large set of derived pictures, could allow to reconstruct how the usage of the information contained in the original images has evolved in time and space, thus permitting to identify, for instance, how

these images have been used by groups of people with different opinions and cultures. Other applications include detection of copyright infringement (illegal copies derived from a copyrighted image) and image retrieval applications (improving clustering accuracy or reducing information redundancy).

In general, Web users do not interact with the original archetype of an image but rather with a version that has been derived from it following an unknown number of modifications carried out by other users. By starting from such image, users create new similar versions by means of a typically limited set of manipulations like resizing, cropping or recoloring. These images are then made available to other users who employ them as basis for newly manipulated versions, thus feeding the process leading to large sets of similar images. Iconic moments in recent History and facts of relevant social impact, such as those collected in Fig. 3.1, exemplify very well the above concepts. The three images have something in common: they have not been captured

Figure 3.1: *Iconic photos with large sets of similar images.*

by several different people. The polar bear was photographed by Arne Nævra in 2005 in Svalbard Island (East of Edgeøya, Norway); the Afghan woman was photographed by Steve McCurry of *The National Geographic* in 1984; finally, it is fair to assume that only a few selected members of the international press were invited to attend the meeting of the three leaders in Washington in 1993. How comes, then, that the Internet is literally flooded with duplicates of these images? Clearly they descend from a limited set of original photos or in some cases, given their age, from high quality scans.

Photographs of famous paintings are another example of the large diffusion of duplicated images. Paintings are accessible to millions of people, their

scenes do not evolve over time and they are practically always photographed approximately from the same distance and perspective and under the same lightning conditions. Despite this, the Web provides countless examples of painting photos whose content is the same but whose characteristics are surprisingly different (e.g. color, size, detail). Consider the most famous of all paintings: the *Mona Lisa*. Every day $26,600$ visitors enter the Louvre museum[1] and the majority of them photograph the *Mona Lisa*. Even though not all the visitors distribute their personal photos over the Internet, some of these pictures are uploaded to websites, blogs and social networks, downloaded by other users, possibly edited (and even tampered with) and uploaded again elsewhere. Other images could be produced by scanning art books. Moreover, as reported by OverstockArt.com,[2] a leading enterprise in handmade oil painting reproductions, the *Mona Lisa* was the most required reproduction of 2010 for both private and public uses (newsstands, commercials, television shows, motion pictures). If some of such imitations are photographed or printed and scanned, several offsprings may be generated, thus complicating the situation even more. In Fig. 3.2 we provide a glimpse of what we mean by displaying just a small subset of the pictures coming from a simple Web search.

It is interesting, then, to understand what kind of relationships exist within such large, ever expanding groups of images. The most challenging aspect of this topic, besides devising a sound formalisation for the problem underlying it, is that in most of the cases we do not known the *true* relationships within a set of images. Rarely authors share information on their photos or users acknowledge the source of their manipulations. Therefore, often we must combine data gathered automatically and information interpreted with human reasoning to understand whether certain relationships are *plausible*.

In this chapter we formalise the above scenario. Sec. 3.1 briefly reviews the state-of-the-art techniques for analysing a group of similar images; Sec. 3.2 defines the meaning of finding dependencies, so that a sound and rigorous framework can be devised; Sec. 3.3 completes the discussion by introducing some basic assumptions on the way images are commonly duplicated, so that the analysis of their relationships is viable in practice.

[1] Source: `www.louvre.fr` annual reports.
[2] Source: `http://tinyurl.com/yaxrjx5`.

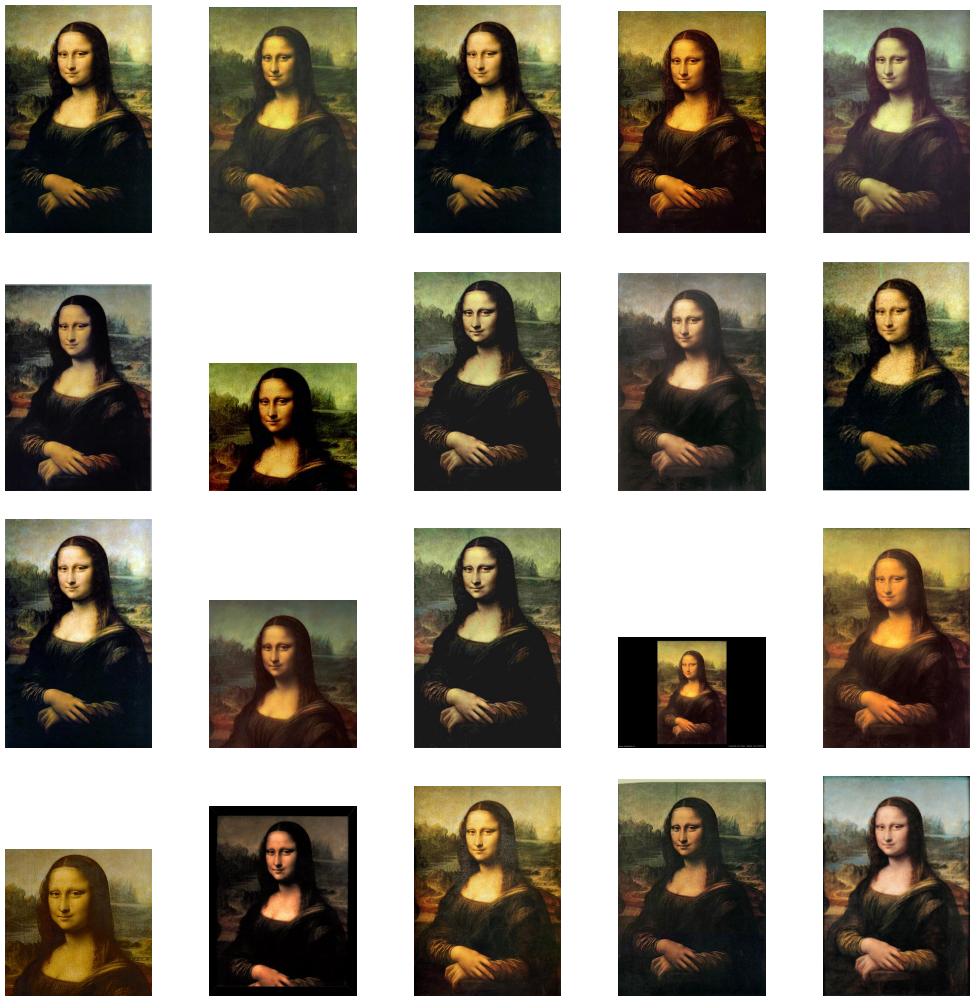


Figure 3.2: *Real-world example of dependencies:* 20 *pictures of Leonardo Da Vinci's "La Gioconda" (1503-1514 circa).*

3.1 Pointers to near-duplicates analysis

Literature on image retrieval provides several techniques for finding duplicated images. Researchers typically identify two instances of the problem, namely Image Exact Duplicate (IED) detection and Image Near Duplicate (IND) de-

tection: techniques belonging to the former category identify *exact copies* of a reference image, while those belonging to the latter identify *near-duplicates*, i.e. variants with similar content generated by means of diverse processing. According to [Jaimes, 2003], the modifications producing near-duplicates can be categorised as follows: i) *Scene*, e.g. change of background, occlusions, movement; ii) *Camera*, e.g. change of perspective, zoom, tilting; iii) *Photometric*, e.g. change of lightning conditions or exposure; and iv) *Digitisation*, e.g. compression, recoloring, resizing, crop.

Regardless of the various categorisations, IED and IND detection methods usually work with large data sets of images with tight time constraints and thus require efficient data management. For this reason, images are represented in a compact form by means of distinctive and robust descriptors such as those provided by the Scale Invariant Feature Transform [Lowe, 2004] (see Chapter 9), which has been employed for example in [Ke et al., 2004; Foo et al., 2007a; Zhu et al., 2008]. In some cases, the combination of descriptors coming from more than one method ensures a higher accuracy with respect to the traditional single-descriptor paradigm, as it has been recently showed in [Battiato et al., 2013]. Other solutions rely on color and texture features [Foo et al., 2007b], hashing [Chum et al., 2007] and stochastic models [Zhang and Chang, 2004].

Given a large collection of images, the above techniques allow to separate similar images into different clusters. However, they are not capable of extracting information on the relationships between the members of each cluster. A first attempt in this sense is the so called *image archaeology* in [Kennedy and Chang, 2008], which analyses the connections between a set of near-duplicates by means of two categories of binary detectors: context free (scaling and grayscale conversion) and context-dependent (crop, overlay and composition). Given a pair of images (I_A, I_B), each detector evaluates whether I_A could generate I_B or vice-versa according to simple rules (e.g. low resolution images cannot generate higher resolution images). If all detectors agree, then the pair is linked and their parent-child relationship is defined according to the direction of the link. The results are organised in a graph called *Visual Migration Map*. Despite the promising experimental results, the system in [Kennedy and Chang, 2008] suffers from some drawbacks such as:

the absence of a rigorous theoretical framework; the assumption to work with data sets not containing exact duplicates, which is not the case in real-world scenarios; the absence of a confidence score describing the plausibility of the relationships; the absence of an estimation of the parameters describing the parent-child relationship; the incapability of taking into account common processing like compression or rotation; the usage of several empirical thresholds for the consistency checks.

The method proposed in [De Rosa et al., 2010], which represents the main contribution of this part of the thesis, allows to overcome the above limitations. A framework for exploring image relationships, called here *dependencies*, is rigorously formalised and practically implemented. Similarly to the work of Kennedy *et al.*, such framework is based on the pairwise comparison of near-duplicates. More specifically, De Rosa *et al.* assume that an image can be divided into two independent contributions, the image content and a random "noise" which behaves like a fingerprint and bears information about the acquisition process. The authors measure the strength of the relationship between a pair of images (I_A, I_B) by correlating the corresponding random fingerprints. [De Rosa et al., 2010] introduced several novel aspects, besides the theoretical formalisation, with respect to the method in [Kennedy and Chang, 2008]. Among them we mention a significant extension of the set of possible geometric transformations and the capability of dealing with compressed images and with exact duplicates.

Following the study of De Rosa *et al.*, a similar system was proposed in [Dias et al., 2012]. In analogy with biological systems, the authors use the term *image phylogeny* to indicate the analysis of the relationships between near or exact duplicates. Dias *et al.* compute a dissimilarity measure between pairs of images by addressing the same subset of processing as [De Rosa et al., 2010]. The procedure, however, is carried out on the whole image rather than on the noise component. Concerning the construction of the dependency graph, called *Image Phylogeny Tree* by the authors, the main difference with the method of De Rosa *et al.* resides in the interpretation of the similarity scores. Dias *et al.*, in fact, favour a more theoretical approach based on minimum spanning trees, while the dependency graph of De Rosa *et al.* is the result of heuristic criteria ruling out implausible relationships.

3.2 Theoretical background

3.2.1 The rationale behind finding dependencies

Consider an event occurring in the real world. Even though an event could be temporally and spatially extended, suppose that the event occurs at a fixed time and it is seen from a particular viewpoint (for example, one of the events of Fig. 3.1). Let the event be called *the real scene*. Suppose that a set of images representing the same real scene is available. The problem is to find the dependencies among such images in order to construct a sort of graph helping to understand how these images have been generated and how the information about the real scene has evolved in time and space.

The first question requiring an answer is: what does *finding the dependencies between images* mean exactly? Among all the possible meanings that can be given to the term *image dependency*, here we aim at understanding whether a digital image has been produced by starting from another image representing the same real scene. Note that, since by definition the images correspond to the same content, the investigated relationship should not be related to the content itself, because from that point of view all the images representing the same scene would be judged as dependent. To better clarify this concept, consider the example of two artists working on two different paintings; if they are free to paint any possible subject, then the possibility that the two painters draw the same topic is extremely low. In this case, a similar content could be taken as an evidence that some form of communication (or some dependency) between the painters occurred. On the contrary, if the subject of the paintings was imposed to the artists beforehand, then the fact that their paintings represent the same scene could not be taken as a demonstration that the artists communicated between them or that one of them copied the work of the other. But if they painted the same content by using exactly the same colors and the same pictorial metaphors, then it could be concluded that the painters had some kind of contact or that one artist copied the other. This chapter focuses exactly on this situation by considering a form of dependency that does not rely on the semantic content of the images.

More precisely, we will assume that any image can be described as the composition of two parts: a part conveying the semantic information related

to the real scene and a content-independent part representing the peculiarities of the process that produced the images. We will consider two images as dependent if some form of similarity exists between their content-independent components. We now give a rigorous formalisation of the above concept.

3.2.2 Preliminary notions

Consider a set of color images $\mathcal{I}$, where an image $I \in \mathcal{I}$ is a $N \times M \times 3$ matrix whose entries are integer values in $[0, 255]$.

Definition 1. *The function ϕ_f characterised by a set of parameters $\wp_{\phi_f}$ is called* fundamental image processing function (f-IPF)*:*

$$\phi_f(\cdot) : \mathcal{I} \times \wp_{\phi_f} \to \mathcal{I}.$$

The domain $\mathcal{D}_{\phi_f}$ of ϕ_f is the set of input images on which the f-IPF can work, the codomain $\mathcal{C}_{\phi_f}$ is the set of output images:

$$\mathcal{C}_{\phi_f} = \{I \in \mathcal{I} \mid \exists I^* \in \mathcal{I},\ \exists p^* \in \wp_{\phi_f} \text{ s.t. } I = \phi_f(I^*, p^*)\}.$$

For example, if ϕ_f were the image resize function, $\wp_{\phi_f}$ would include the scale factors along width and height and the interpolation method. Note that even when no processing is carried out (e.g. when an image is duplicated), Definition 1 remains valid: in this case, the identity $\phi_\perp(\cdot) : \mathcal{I} \times \{\varnothing\} \to \mathcal{I}$ is applied. From now on the set of all f-IPFs will be indicated with Φ_f.

Two fundamental image processing functions can be composed with each other as follows.

Definition 2. *Let $\phi_1 : X \to Y$ and $\phi_2 : V \to Z$ be two f-IPFs and let $I \in \mathcal{I}$ be an image; the function $\phi_3 = \phi_2(\phi_1(I))$ is called* composite image processing function (c-IPF) *and is indicated with $\phi_3 = \phi_2 \circ \phi_1 : X \to Z$.*

In other words, ϕ_1 and ϕ_2 can be composed by firstly applying ϕ_1 to an argument I and then applying ϕ_2 to the result. Note that the codomain of ϕ_1 must be included in the domain of ϕ_2: $Y \subseteq V$. The above composition can be generalised to an arbitrary number r of f-IPFs as follows.

Definition 3. *Let $\phi_1, \phi_2, \ldots, \phi_r$ be r f-IPFs; then Φ_c is the set of all possible compositions of r f-IPFs $\in \Phi_f$, where r is the* order *of the composition.*

For example, the set Φ_c of order 2 defined by functions ϕ_1 = JPEG compression with parameters $\wp_{jpeg}$ and ϕ_2 = rotation with parameters $\wp_{rot}$ is
$\Phi_c = \{$ `JPEG(rotation(`$I, \wp_{\phi_{rot}}$`),`$\wp_{\phi_{jpeg}}$`)` , `rotation(JPEG(`$I, \wp_{\phi_{jpeg}}$`),`$\wp_{\phi_{rot}}$`)` $\}$.

In general, according to the application scenario the analysis can be limited to a possibly small subset of image processing functions, namely $\Phi \subset \Phi_f \cup \Phi_c$. To conclude, it is also useful to introduce the concept of compatibility between a given image and the subset Φ.

Definition 4. *Given an image $I \in \mathcal{I}$ and an image processing function $\phi \in \Phi$, I is* compatible *with ϕ if $I \in \mathcal{C}_\phi$.*

3.2.3 Dependency test

Consider a set of images $\mathcal{I}$ representing the same scene; the main interest is to investigate the pairwise dependency between such images. To do so, we hypothesise that any image $I \in \mathcal{I}$ can be univocally described as the composition of two separable and independent parts, i.e. $[I]_\mathcal{C}$ describing the content of the real scene and $[I]_\mathcal{R}$ representing the content-independent characteristics of the image, a sort of random part of the image:

$$I \leftrightarrow \left[\, [I]_\mathcal{C}, [I]_\mathcal{R} \,\right] \qquad \forall I \in \mathcal{I}. \tag{3.1}$$

In the sequel we will refer to $[\cdot]_\mathcal{R}$ as *randomness*. We can verify the dependency between two images I_A and I_B, hereafter considered as two random information sources, by means of their mutual information

$$\mathrm{I}(I_A; I_B) = H(I_A) - H(I_A|I_B), \tag{3.2}$$

where $H(I_A)$ is the entropy of the source I_A and $H(I_A|I_B)$ is the conditional entropy of the source I_A conditioned to I_B. By representing the images through

the independent components introduced before, we rewrite Eq. (3.2) as:

$$\begin{aligned}
\mathrm{I}(I_A; I_B) &= H\big([[I_A]_{\mathcal{C}}, [I_A]_{\mathcal{R}}]\big) - H\big([[I_A]_{\mathcal{C}}, [I_A]_{\mathcal{R}}] \mid [[I_B]_{\mathcal{C}}, [I_B]_{\mathcal{R}}]\big) && (3.3)\\
&= H\left([I_A]_{\mathcal{C}}\right) + H\left([I_A]_{\mathcal{R}}\right) - H\big([I_A]_{\mathcal{C}} \mid [I_B]_{\mathcal{C}}, [I_B]_{\mathcal{R}}\big)\\
&\quad - H\big([I_A]_{\mathcal{R}} \mid [I_B]_{\mathcal{C}}, [I_B]_{\mathcal{R}}, [I_A]_{\mathcal{C}}\big) && (3.4)\\
&= H\big([I_A]_{\mathcal{C}}\big) + H\big([I_A]_{\mathcal{R}}\big) - H\big([I_A]_{\mathcal{C}} \mid [I_B]_{\mathcal{C}}\big) - H\big([I_A]_{\mathcal{R}} \mid [I_B]_{\mathcal{R}}\big) && (3.5)\\
&= \mathrm{I}\big([I_A]_{\mathcal{C}}; [I_B]_{\mathcal{C}}\big) + \mathrm{I}\big([I_A]_{\mathcal{R}}; [I_B]_{\mathcal{R}}\big). && (3.6)
\end{aligned}$$

The first term of Eq. (3.4) is obtained by exploiting the chain rule[3] and the independence[4] between $[I_A]_{\mathcal{C}}$ and $[I_A]_{\mathcal{R}}$; the second term is obtained by exploiting again the chain rule.[5] Eq. (3.5) is obtained by exploiting the independence between $[I_A]_{\mathcal{C}}$ and $[I_B]_{\mathcal{R}}$ and between $[I_A]_{\mathcal{R}}$, $[I_A]_{\mathcal{C}}$ and $[I_B]_{\mathcal{C}}$. The mutual information between the images can thus be expressed as the sum of the mutual information between the $\mathcal{C}$ components and the $\mathcal{R}$ components. The analysis can be limited to the second term (the content-independent one), since the first term will never be null, due to the intrinsic dependency between the $\mathcal{C}$ components (since they refer to the same real scene).

We can now cast the problem of determining the dependency between I_A and I_B as a hypothesis testing problem as follows:

$$\begin{aligned}
\mathcal{H}_0 &= \{\ [I_A]_{\mathcal{R}} \text{ and } [I_B]_{\mathcal{R}} \text{ are independent } \}\\
\mathcal{H}_1 &= \{\ [I_A]_{\mathcal{R}} \text{ and } [I_B]_{\mathcal{R}} \text{ are dependent } \}.
\end{aligned} \tag{3.7}$$

Equivalently, we can verify whether $\mathrm{I}\left([I_A]_{\mathcal{R}}; [I_B]_{\mathcal{R}}\right) = 0$. The design of an optimal criterion for such a test would require the availability of a good statistical model to describe $[I_A]_{\mathcal{R}}$ and $[I_B]_{\mathcal{R}}$ and their possible relationship through the functions contained in Φ. Modeling such a relationship is very complicated, hence we will adopt a simplified strategy.

More precisely, by considering a set of image processing functions Φ, we make the following assumption: if there is some relationship between two

[3] Let X, Y be two random variables; then $H(X,Y) = H(X) + H(Y|X)$ (see [Cover and Thomas, 2006], Chapter 2).

[4] In general $H(X|Y) \leqslant H(X)$. If X, Y are independent, then $H(X|Y) = H(X)$.

[5] According to the following corollary of the chain rule: $H(X,Y|Z) = H(X|Z) + H(Y|X,Z)$.

images $I_A, I_B \in \mathcal{I}$, then one of the two images can be obtained at least approximately by applying a $\phi_j \in \Phi$ to the other. A possibility, then, would be to compute the correlation coefficient ρ_j between $[I_B]_{\mathcal{R}}$ and $[\phi_j(I_A)]_{\mathcal{R}}$, for each $\phi_j \in \Phi$, and use as decision statistic the maximum in this set of correlations:

$$\rho_{max} = \max_{\phi_j \in \Phi} \rho([I_B]_{\mathcal{R}}, [\phi_j(I_A)]_{\mathcal{R}}). \tag{3.8}$$

In Eq. (3.8) the maximisation over the parameters in $\wp_{\phi_j}$ has been omitted for simplicity. Note that the previous statistic is voluntarily asymmetric, i.e. it tests the dependency of I_B on I_A and not the other way round. Looking for all possible $\phi \in \Phi$ requires a huge computational effort, all the more that for each function all the parameters in $\wp_\phi$ should be considered. In addition, the probability of detecting a false dependency would increase with the number of functions and with the size of the parameter space. In order to overcome the above problems, we devise a different strategy, i.e. we try to "guess" the function $\phi^* \in \Phi$ that has been used to pass from I_A to I_B by relying on the content part of the images. Suppose, for instance, that the set Φ contains only the rotation f-FIP, i.e. a function that rotates the input image by a certain angle. Instead of computing the correlation coefficient between the random part of I_B and the random part of all the rotated versions of I_A, we estimate the most likely rotation angle by relying on $[I_A]_{\mathcal{C}}$ and $[I_B]_{\mathcal{C}}$. Then, we compute the correlation coefficient between the random part of I_B and the random part of I_A rotated by the estimated angle and we use such coefficient as decision statistic. Assuming the existence of an efficient way to estimate the rotation angle, this approach is much faster than searching exhaustively for ρ_{max} as in Eq. (3.8). To finally accept or reject the hypothesis of independence, we compare the correlation coefficient ρ^* between $[I_B]_{\mathcal{R}}$ and $[\phi^*(I_A)]_{\mathcal{R}}$ with a suitable threshold T_ρ:

$$\begin{cases} \rho^* < T_\rho & I_B \text{ does not depend on } I_A \\ \rho^* \geqslant T_\rho & I_B \text{ depends on } I_A. \end{cases} \tag{3.9}$$

T_ρ should be set rigorously by studying the statistical characteristics of ρ^* and by fixing a value for the false positive probability. Alternatively an empirical analysis may be carried out and T_ρ determined experimentally.

3.2.4 Dependency graph

The final aim of the analysis consists in representing the collection of $|\mathcal{I}|$ images by means the so called *dependency graph*, i.e. an oriented weighted graph $G = (V, E)$ whereby the nodes $V = \{1, \ldots, |\mathcal{I}|\}$ correspond to the images in $\mathcal{I}$ and $E = \{e(i, j) : i, j \in V\}$ correspond to the relationships between pairs (i, j) of images. The orientation of each edge describes the parent-child relationship between the images and the weight $w_{(i,j)}$ quantifies the strength of their relationship. Edges are also labelled with the composite image processing function $\phi \in \Phi$ that was used to pass from i to j.

To construct the actual graph, we collect the dependency test values for each pair of images in $\mathcal{I}$, either by computing ρ_{max} or ρ^*, and we build a first version of the dependency graph by keeping only those oriented links for which the correlation is above the threshold T_ρ. By giving the graph a semantic nature, we can infer other relationships between images by means of additional sets of rules that we call *ontology*. Such relationships can enrich the information provided by Eq. (3.9) and modify the graph to resolve ambiguous situations that could not be disambiguated by a pairwise analysis. We can imagine several examples of ontology rules, including the following.

- When two nodes are connected by two edges oriented in opposite directions, the weakest link is removed. A similar strategy can be applied to avoid the presence of loops in the graph.
- Assuming that no image splicing is considered (i.e. only one parent for each image), if there are multiple links leading to the same node, then the one with the highest correlation is kept.
- If it is possible to understand whether an image of the pair is a crop of the other, then the link connecting the former to the latter is suppressed.
- If the two files storing the images under analysis are exact copies, e.g. according to checksum, then the two nodes can either be fused together or file information (for example the date of creation) can be used to establish the correct link.
- Assuming that inpainting is less probable than text overlay, if two images have exactly the same content and the second contains text overlays

while the first does not (techniques such as those in [Chen et al., 2004] and [Wu et al., 2009] can reveal these processing), the link from the latter to the former is suppressed.

Furthermore, information provided by image forensic tools could be also exploited. For example, source detection algorithms could help pruning links between images generated by incompatible categories of devices. Similar benefits could be brought by forgery detection, which would also allow to abandon the assumption that an image can not have more than one parent.

3.3 Definition of Φ and parameter estimation

As explained in the previous section, given a set of image processing functions Φ, if there is some relationship between two images $I_A, I_B \in \mathcal{I}$, then we can obtain one of the two images by applying a certain number of $\phi_j \in \Phi$ to the other. The computational effort required by looking for all possible $\phi \in \Phi$ over all parameters $\wp_{\phi_j}$ prevents the use of an exhaustive approach to the problem. Therefore, the idea is to choose a subset Φ^* whose functions allow to estimate the best transformation between I_A and I_B. Then, I_A is transformed according to such functions and the similarity between the resulting image I'_B and I_B (or more precisely, between their random components) is evaluated.

We choose the functions in Φ^* according to the following rationale. Excluding forgeries for the sake of simplicity, users typically want to improve the quality of an image. To do so, a limited set of tools is usually employed: commonly colors are changed if they look wrong or unpleasant; similarly, contrast and brightness can be increased or reduced; the image can be adjusted if the original framing was not oriented correctly, or can be cropped, zoomed and so on; finally, most of the times the image is stored or redistributed in a compressed format (e.g. JPEG).

By following this rationale, we assume that the best transformation ϕ^* leading from an image I_A to an image I_B can be obtained as a composition of the following three functions:

$$\Phi = \{ \phi_{color}, \phi_{geometry}, \phi_{compression} \}. \tag{3.10}$$

In case one of these operations is not carried out, then the respective image processing function will coincide with the identity $\phi_{\perp}$.

3.3.1 Color transformation

We find the optimal transformation ϕ^*_{color} mapping the colors of an image I_A to another image I_B by means of the so called *color transfer* [Reinhard et al., 2001; Welsh et al., 2002; Tai et al., 2005]. As the name suggests, this is the process of borrowing the color characteristics from a source image and passing it to the target image. Following the transfer, the colors of the target image are similar to those of the source image. In Fig. 3.3 an example obtained with the method in [Reinhard et al., 2001] is provided. Following color transfer, the approximation I'_B of I_B generated by starting from I_A corresponds to:

$$I'_B = \phi^*_{color}(I_A, \wp_{\phi_{color}}). \tag{3.11}$$

Figure 3.3: *Example of color transfer with the technique of* [Reinhard et al., 2001]. *Left: source; middle: target; right: result.*

3.3.2 Geometrical transformation

The most common geometric transformations belong to a rather small subset: translation, rotation, resampling, shearing, zoom, sometimes warping. All these processing can be effectively estimated in a single step by means of *image registration*, i.e. the process allowing to find corresponding points

between two images and to spatially align them in such a way that a certain distance measure between them is minimised. Registration has been used in several computer vision applications including medical imagining, image fusion, object detection and recognition, motion analysis and change detection. Although several different approaches to registration have been proposed [Zitová and Flusser, 2003], their working principle is similar: firstly, robust and highly descriptive features are extracted from both the source image and the target image; secondly, the features are matched across the two images; finally, the mapping functions and their parameters are estimated and applied to align the target and the source. The parameters can be applied to the image globally (rigid and affine transformations) or locally (elastic transformation). An example of the registration procedure is shown in Fig. 3.4.

For the dependency problem, let I_A and I_B be respectively the source and the target image; then, the approximation I'_B of I_B for the geometry estimation stage corresponds to:

$$I'_B = \phi^*_{geometry}(I_A, \wp_{\phi_{geometry}}). \tag{3.12}$$

3.3.3 Compression transformation

From now on we will assume that the images whose dependencies are under analysis are JPEG compressed. This hypothesis does not undermine the generality of the framework mainly for three reasons: i) although not limited to it, the application scenario is the Internet, where around 70% of the total images on the top 10 million websites are in JPEG format,[6] as shown in Fig. 3.5; ii) the conclusions drawn here naturally adapt to other kinds of compression; iii) if the images are not compressed, then $\phi_{compression} = \phi_\perp$.

In practice, if I_A and I_B are two JPEG images, the best transformation $\phi^*_{compression}$ leading from I_A to I_B should be obtained by computing all the possible versions of compressed I_A over the set of parameters $\wp_\phi$ and compare each version with I_B. Even though this task would not represent a huge computational burden, we adopted a different strategy. It is possible to gather

[6] Source: http://w3techs.com/technologies/overview/image_format/all. Statistics are computed on the top 10 million websites according to the Amazon.com company (July 2013).

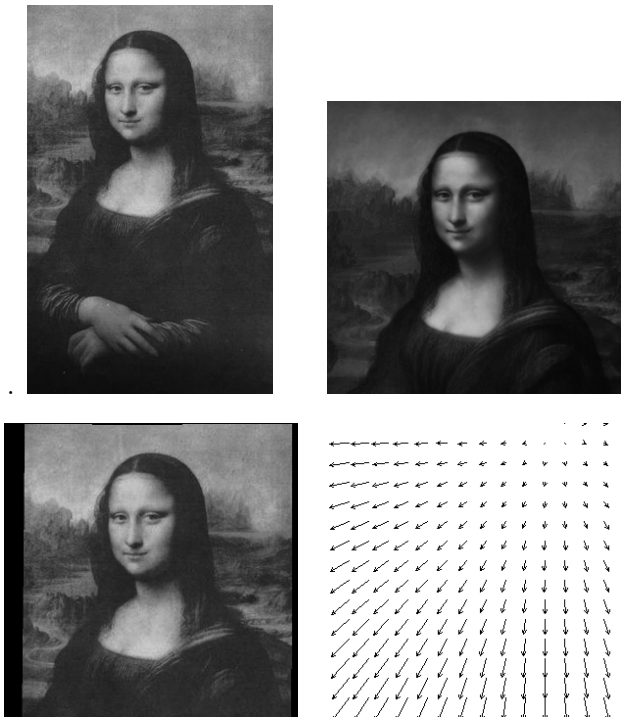


Figure 3.4: *Example of registration with the technique of* [Arganda-Carreras et al., 2006]. *From top left to bottom right: source image, target image, registered image and deformation field mapping the transformation.*

information about DCT coefficients, quantisation tables, Huffman coding tables and color space directly from the JPEG file. From these data we can estimate the compression quality factors QF_A and QF_B. Consequently, from the perspective of the sole JPEG compression, the approximation I'_B of I_B corresponds to:

$$I'_B = \phi^*_{compression}(I_A, QF_B). \tag{3.13}$$

3.3.4 Composition of ϕ_{color}, $\phi_{geometry}$ and $\phi_{compression}$

According to Sec. 3.2.3, we should carry out the dependency test not only over all the parameters $\wp_\phi$ of each $\phi \in \Phi^*$ but also on all possible combinations of

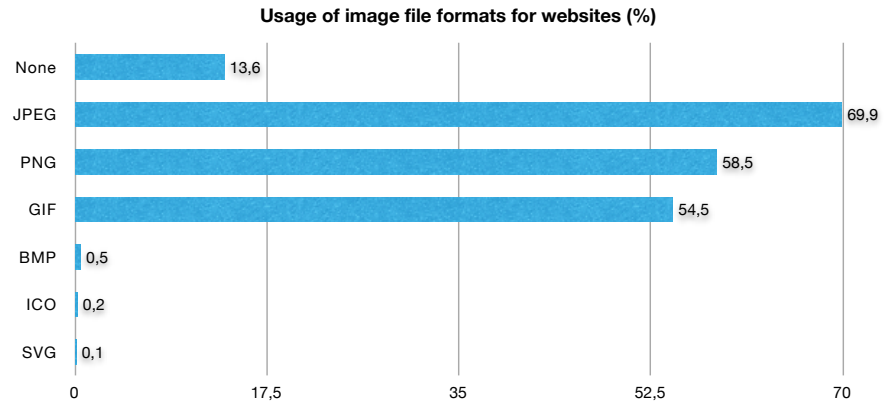


Figure 3.5: *Usage of image formats in websites (updated July 2013). Note that a website may use more than one format.*

the functions. The current case of a limited number of functions should not be too computationally taxing and nothing forbids to maximise ρ over all the combinations. However, here we choose to replicate the typical behaviour of an image user by chaining color, geometry and compression of Eqs. (3.11)–(3.13):

$$I'_B = \phi^*_{compression}(\ \phi^*_{geometry}(\ \phi^*_{color}(I_A, \wp_{\phi_{color}}), \wp_{\phi_{geometry}}), \wp_{\phi_{compression}}\ ). \quad (3.14)$$

One observation is in order. While it is safe to assume that ϕ^*_{color} and $\phi^*_{geometry}$ are approximatively commutative due to the fact that color transfer does not take into account geometry and vice-versa, the same is not true for compression. As a consequence, $\phi^*_{compression}$ should always close the chain of processing.[7] We verify the hypothesis of dependence by computing ρ^* between $[I_B]_{\mathcal{R}}$ and $[I'_B]_{\mathcal{R}}$.

3.4 Concluding remarks

In this chapter we investigated the problem of finding relationships within a set of near-duplicate images. Although there exist a number of forensic algorithms

[7] The possible intermediate cases in which partial results are stored in JPEG format are not considered here because when multiple compressions occur, only the quality factor of the last one can be estimated accurately.

permitting to gather information about the history of an image, the majority of instruments developed so far focus on the analysis of single images. In several applications, however, the investigation of the relationships between a group of images, may be of similar importance. This chapter discussed a possible formalisation of the problem, which will be validated in a practical scenario in the next chapter.

Chapter 4

The Dependency Explorer Framework

WE NOW put in practice the framework formalised in Chapter 3 in realistic scenarios of growing complexity. The chapter proceeds according to the following outline. We describe the system going by the name of *Dependency Explorer Framework* in Sec. 4.1, where we detail its building blocks and the tools implementing them. Sec. 4.3 introduces the case studies consisting of simulated sets of near-duplicates whose relationships are known and of real-world sets whose images have been gathered from the Web. Secs. 4.4–4.6 assess the performance of the proposed system in reconstructing the dependency graph. Sec. 4.7 outlines some possible directions for future research.

4.1 Architecture of the system

To be compliant with the simplified formalisation of Sec. 3.3, the instantiation of the set Φ includes the concatenation of 3 elementary functions: color transformation or histogram equalisation ϕ^*_{color}, JPEG compression $\phi^*_{compression}$ and geometric transformation $\phi^*_{geometry}$. For the sake of brevity, we will refer to such functions as ϕ_c, ϕ_j and ϕ_g; moreover, we will omit their parameter sets $\wp_{\phi_c}$, $\wp_{\phi_j}$ and $\wp_{\phi_g}$ whenever they are not necessary.

Furthermore, we assume that all the images have been processed by the consecutive application of ϕ_c, ϕ_g and ϕ_j. We handle the case in which one or more of these functions have not been applied by adjusting the parameters of the missing function(s), so that its (their) effect is null.

We work only with JPEG images, that is after any processing step the manipulated image is JPEG compressed with arbitrary quality factor.

4.1.1 Overview of the Dependency Explorer Framework

The Dependency Explorer Framework is based on a pairwise comparison of all the images belonging to the set under analysis. For each pair $(I_A, I_B) \in \mathcal{I}$, the goal is to understand whether I_B could have been generated from I_A.

We have already observed that such an analysis should be carried out on the random component of the images. Therefore, we decompose each image as in Eq. (3.1) by means of image denoising, whereby the denoised image corresponds to $[I]_{\mathcal{C}}$ and the noise to $[I]_{\mathcal{R}}$.

Fig. 4.1 sketches the architecture of the implemented system. For every pair (I_A, I_B) the system works as explained in the sequel.

1. **Color Matching Block**. The image I_A is modified so that its colors correspond to the colors of I_B. According to our notation, the output of color matching is indicated with $\phi_c(I_A)$.

2. **JPEG Matching Block**. The quality factor of I_B is estimated and used to compress the color-matched version of I_A, thus producing the image $\phi_j(\phi_c(I_A))$, whose content $[\phi_j(\phi_c(I_A))]_{\mathcal{C}}$ and randomness $[\phi_j(\phi_c(I_A))]_{\mathcal{R}}$ are separated by means of image denoising.

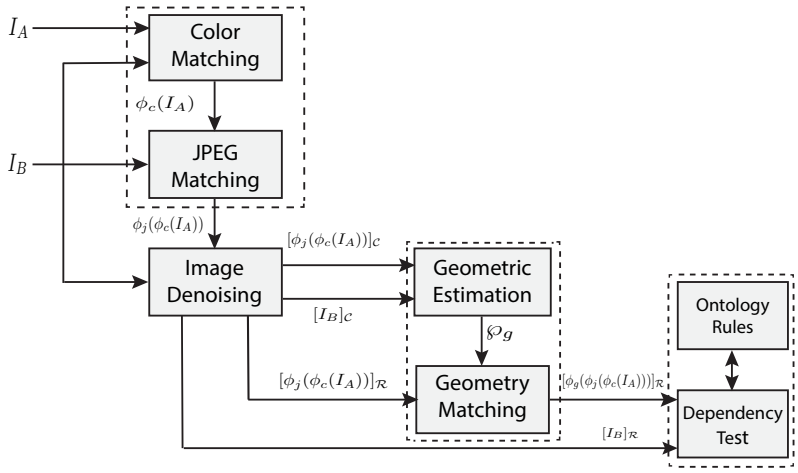


Figure 4.1: *Scheme of the Dependency Explorer Framework.*

3. **Geometrical Matching Block**. The parameters $\wp_{\phi_g}$ of the geometric transformation leading from I_A to I_B are estimated by relying only on the content of the two images for two reasons: i) the information brought by visually significant content guarantees more accurate results with respect to using randomnesses; and ii) to avoid that the correlation between the random parts is artificially increased as a consequence of the registration.[1] The parameters in $\wp_{\phi_g}$ are then used to align the randomness produced by the previous stage, i.e. $[\phi_j(\phi_c(I_A))]_{\mathcal{R}}$ to $[I_B]_{\mathcal{R}}$, thus leading to the registered randomness $[\phi_g(\phi_j(\phi_c(I_A)), \wp_{\phi_g})]_{\mathcal{R}}$.

4. **Dependency test**. The correlation coefficient ρ^* between randomnesses $[\phi_g(\phi_j(\phi_c(I_A)), \wp_{\phi_g})]_{\mathcal{R}}$ and $[I_B]_{\mathcal{R}}$ is computed. To accept or reject the hypothesis of dependence, ρ^* is compared with a threshold T_ρ.

In practice, we repeat the above procedure for all the pairs $(I_A, I_B) \in \mathcal{I}$ and we store the un-thresholded values of ρ^* into the $|\mathcal{I}| \times |\mathcal{I}|$ *correlation matrix*

[1] Registration maximises the similarity between the source and the target. The registered image is then highly correlated to the target image. This is good for the content, since the transformation can be estimated more accurately; the original values of the source randomness, however, may be altered to be more similar to those of the target, thus leading to false relationships.

C, where $|\mathcal{I}|$ is the number of images in the set. Obviously, we do not carry out the comparison between an image and itself. By relying on C we compute a preliminary version of the dependency graph and we further refine it with the following ontology rules:

- **No pairwise loops**. If two nodes are connected by two links with opposite orientations, then only the one with highest ρ^* is kept.
- **Only one parent**. If a node has more than one predecessor, i.e. an image with more than one parent, then only the one with by the highest ρ^* is kept.

4.1.2 Employed tools

(A) Image denoising

The decomposition of the images into the content $[I]_{\mathcal{C}}$ and the randomness $[I]_{\mathcal{R}}$ is carried out by means of denoising. The chosen algorithm is the one in [Mihçak et al., 1999], which relies on a spatially adaptive statistical model for the Discrete Wavelet Transform (DWT) image coefficients. The method assumes that noisy coefficients $Y(k)$ can be modeled as the addition of the image without noise $X(k)$ (i.i.d. with zero mean) and a white Gaussian noise $n(k)$ with known variance σ_n^2. The denoised image is predicted in the Wavelet domain by means of the Minimum Mean Squared error (MMSE) estimation as follows:

$$\widehat{X}(k) = \frac{\sigma_x^2(k)}{\sigma_x^2(k) + \sigma_n^2} G(k), \tag{4.1}$$

where $\sigma_x^2(k)$ and σ_n^2 are respectively the variance of $X(k)$ and $n(k)$. Since the true signal and thus $\sigma_x^2(k)$ are unknown, the estimation $\widehat{\sigma}_x^2$ achieved by means of a MAP (Maximum A-posteriori Probability) approach on noisy wavelet coefficients is used instead.

We perform an additional step before leaving the DWT domain: to better discriminate the noise from the high frequency scene content surviving Mihçak *et al.*'s filtering, a second filter called *PRNU enhancer* is borrowed from [Caldelli et al., 2010]. This function exalts weak components of n in the DWT

domain by weighting them more than the others:

$$[I]_{\mathcal{R}} = \begin{cases} 0 & \text{if } n(k) < -\gamma \\ -\cos\left(\frac{n(k)\pi}{2\gamma}\right) & \text{if } -\gamma \leqslant n(k) < 0 \\ +\cos\left(\frac{n(k)\pi}{2\gamma}\right) & \text{if } 0 \leqslant n(k) < \gamma \\ 0 & \text{if } n(k) \geqslant \gamma. \end{cases} \tag{4.2}$$

The parameter γ can be used to control the cut-off value between the noise components and the scene details.

(B) Color matching

The chosen method for the color matching stage relies on the algorithm described in [Reinhard et al., 2001]. In a nutshell, it works as follows. First of all, the image is converted into the $L\alpha\beta$ color space to avoid the very strong correlation existing between RGB channels. In this space, mean and standard deviation of the three axes of the target's color distribution are passed to the source image as in Eq. (4.3):

$$\begin{aligned} L^* &= L_s - \overline{L}_s \quad , & \hat{L} &= (\sigma_t^L/\sigma_s^L) \cdot L^* + \overline{L}_t \\ \alpha^* &= \alpha_s - \overline{\alpha}_s \quad , & \hat{\alpha} &= (\sigma_t^\alpha/\sigma_s^\alpha) \cdot \alpha^* + \overline{\alpha}_t \\ \beta^* &= \beta_s - \overline{\beta}_s \quad , & \hat{\beta} &= (\sigma_t^\beta/\sigma_s^\beta) \cdot \beta^* + \overline{\beta}_t. \end{aligned} \tag{4.3}$$

In practice, Eq. (4.3) summarises three steps: i) the mean is subtracted from each channel of the source image; ii) the source channels are scaled by factors corresponding to the ratio of target and source standard deviations; iii) mean values of the target image channels are added to the scaled source. Finally, the resulting color matched image is converted to RGB.

The source image I_A and the color-transformed image $\phi_c(I_A)$ can be compared by means of the *color difference* ΔE, i.e. a metric based on Human Visual System describing the perceptual distance between the channels of the two images in a convenient color space [Wyszecki and Stiles, 1982]. The higher is ΔE, the more distinguishable are the colors of the two images. If (I_A, I_B) are dependent, then ΔE can be effectively used as an indication of the intensity of the original color transformation. In this thesis we compute the color difference by means of the method described in [Rajeev Ramanath et al., 2002],

which is based on the observation that the human eye is more sensitive to differences along edges than in uniform regions. Consequently, edge information is incorporated into the distance metric as follows:

$$\Delta E^2 = \left[(1+\alpha)\Delta L^2 + (1+\beta)\Delta C^2 + (1+\gamma)\Delta H^2\right]. \tag{4.4}$$

ΔL, ΔC and ΔH are the channel by channel differences of the two images under analysis computed in the CIELAB color space.[2] The parameter α weights the luminance pixels sitting on edges as follows:

$$\alpha = 1 - e^{-\nabla^2 L/\eta_L}, \tag{4.5}$$

where $\nabla^2 L$ is the image gradient and η_L a constant normalising the exponent in $[0,1]$ fixed according to the maximum value of the gradient image. Parameters β and γ are computed in the same way for the remaining channels. The final metric $\overline{\Delta E}$ is obtained as the average of ΔE. It has been observed experimentally that the value $\overline{\Delta E} = 2.3$ acts as the just noticeable threshold above which color changes can be clearly perceived by a human observer [Mahy et al., 1994].

(C) Compression matching

The information that is exploited to estimate the quality factor of the images is obtained by means of the JPEG Toolbox by Phil Sallee.[3]

(D) Image registration

For the geometrical matching stage we have chosen the registration technique in [Sorzano et al., 2005]. This algorithm for elastic registration tries to estimate a function, called deformation field and modeled by means of a linear combination of weighted and shifted B-splines [De Boor, 1978], transforming the coordinates of the source image into the coordinates of the target image.

[2] In the original formula used in [Rajeev Ramanath et al., 2002], ΔE^2 is multiplied by a constant ζ describing the impact of large field of views on perceived color. Similarly to Rajeev *et al.*, we set $\zeta = 1$.

[3] The software is freely available for download at `http://dde.binghamton.edu/download/jpeg_toolbox.zip`.

The weights of such function are estimated through a minimisation problem including three terms: the energy F_{img} of the pixel-wise difference between the intensities of source and target; the error F_{μ} in mapping the automatic or manual landmarks (if provided) anchoring the deformation at some specific locations; a regularisation term ensuring the smoothness of the deformation by controlling the divergence and the curl of the field (see [Sorzano et al., 2005]). The minimised energy function F is a linear combination of the above terms:

$$F = w_i \cdot F_{img} + w_{\mu} \cdot F_{\mu} + (w_d \cdot F_{div} + w_r \cdot F_{rot}), \tag{4.6}$$

where each component is controlled by a specific weight w, whose value will be discussed in the experimental validation of Sec. 4.5.

4.2 Threshold determination

In Sec. 3.2.3 we explained that to accept or reject the hypothesis of independence among two images A and B we must compare the correlation coefficient ρ^* between $[I_B]_{\mathcal{R}}$ and $[\phi^*(I_A)]_{\mathcal{R}}$ with a suitable threshold T_{ρ}. To determine such a value, we relied on 20 independent images, to which we will refer to as the *archetypes*. Each archetype has 45 children that have been generated by means of a combination of cropping, color transfer and JPEG compression. More precisely, we varied the JPEG quality factor in $\{80, 85, 90, 95, 100\}$ and the number of cropped columns in $\{0, 40, 80\}$ and we used 3 external images for color transfer.

The goal of the experiment is to create two large sets: one of real father-child examples and one of images having same or similar content but no direct dependency. The examples belong to the following categories of correlations.

- **Correlation among archetypes and their children**. For each of the archetypes A_i, $i = 1, \ldots, 20$, we calculated the correlation with all its children $\rho^*(A_i, C_{i,j})$, $j = 1, \ldots, 45$. By doing so, we simulated father-child relationships. This procedure led to 900 scores corresponding to positive (or true) examples of dependency.

- **Autocorrelation of the archetypes**. For each of the archetypes A_i, $i = 1, \ldots, 20$, we calculated the autocorrelation $\rho^*(A_i, A_i)$. By doing

so, we simulated exact duplicates. This procedure led to 20 scores also corresponding to positive (or true) examples of dependency.

- **Correlation among children of the same archetype**. For each archetype A_i, $i = 1, \ldots, 20$, we calculated the correlation $\rho^*(C_{i,j}, C_{i,k})$, $j, k = 1, \ldots, 45$, $j \neq k$. By doing so, we simulated correlations among siblings that are not linked by a father-child relationship. We randomly selected 900 scores (45 from each archetype) corresponding to negative (or false) examples of dependency.

- **Correlation among archetypes**. For each of the archetypes A_i, we calculated the correlation $\rho^*(A_i, A_k)$, $i, k = 1, \ldots, 20, i \neq k$. By doing so, we simulated correlations among images that are independent and do not share the same content. We randomly selected 90 scores also corresponding to negative (or false) examples of dependency.

We visualised all the above scores in the scattergram of Fig. 4.2. The magenta triangles in the upper left corner correspond to the autocorrelations of archetypes; the green diamonds sitting on the x-axis correspond to the correlations among independent archetypes. The red circles and the blue squares, corresponding respectively to correlations among siblings and correlations among fathers and children, form two distinguishable clusters, which are close to each other because the majority of the images share exactly the same visual content.

To understand whether the clusters can be effectively separated, we varied the threshold T_ρ in $[0, 1]$ to obtain the Receiver Operating Characteristics (ROC) curve in Fig. 4.3 (left). We also computed the F_1-score as a function of T_ρ in Fig. 4.3 (right), that is the harmonic mean of precision p and recall r:

$$p = \frac{Tp}{Tp + Fp}, \quad r = \frac{Tp}{Tp + Fn}, \quad F_1 = 2\,\frac{p \cdot r}{p + r}. \tag{4.7}$$

Tp is the number of true dependencies that are correctly identified; Fp is the number of false dependencies erroneously identified as true; Fn is the number of true dependencies not identified correctly. Hence, precision is the fraction of true dependencies among all those claimed as true and recall is the probability of identifying true dependencies. The closer is the F_1-score to 1, the more accurately we can separate the two classes of dependencies.

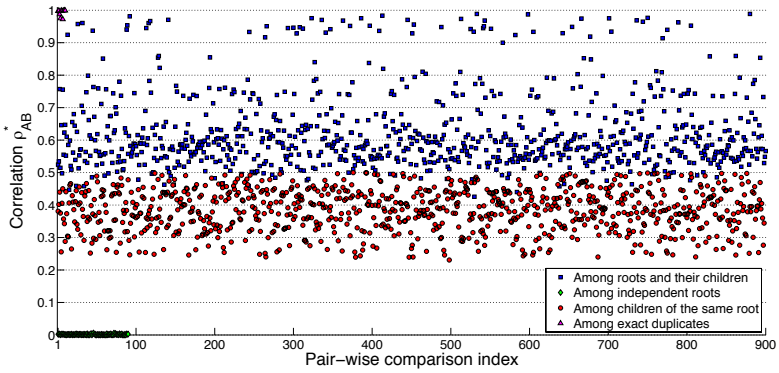


Figure 4.2: *Scattergram for the correlation computed among: archetypes and their children (squares); independent archetypes (diamonds); exact duplicates (triangles) and children of the same archetype (circles).*

Despite their proximity, the clusters can be separated with satisfactory accuracy. Based on the two curves of Fig. 4.3, we let $T_\rho = 0.5$; for such a value, in fact, the F_1-score reaches a maximum of 0.974 and the corresponding point in the ROC ensures true positive rate 0.967 and false positive rate 0.03.

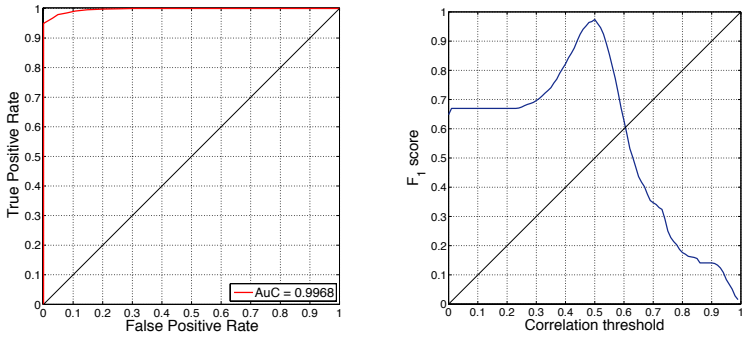


Figure 4.3: *Dependency threshold determination. Left: ROC obtained by varying the dependency threshold; and right: F_1-score as a function of T_ρ.*

4.3 Construction of case studies

We considered four case studies corresponding to different image data sets: two synthetic ones simulating a set of near-duplicates and two realistic ones consisting on images actually collected from the Web. The various cases are characterised by a growing difficulty as follows.

For the synthetic case studies the difficulty is related to the change of perspective of the two independent images used to create the graph: in the former case, a slight change makes the analysis somewhat *easier* because the difference of content increases the discriminative power of the system; conversely, the latter case is *harder* since the two root images share exactly the same visual content.

The difficulty of the real-world case studies derives from the size of the data sets (20 and 55 images respectively) and from the fact that ground truth for dependencies is not available. Very rarely, in fact, the source of an image is referenced in websites and consequently the validity of the results needs to be verified by visual inspecting the graph. Moreover, the subject of the images also contributes to the complexity of the study: the former case tests the system on pictures of a painting, which is available every day to many tourists potentially producing several independent images with few relationships between each other. In the latter case only one person photographed the *real scene* and made it available. Therefore, it is more probable that several dependent images have been generated and more complex links between the nodes of the dependency graph should be expected. The experimental results of the next section prove the validity of this assumption.

(A) Synthetic case studies

In the first two case studies the set $\mathcal{I}$ consists of 10 color images of size 2048×1536 pixels. The data sets are created in the same way for both cases by starting from two independent natural images, namely I_1 and I_2, taken by a digital camera with its native JPEG compression not considered as a processing. Fig. 4.4 shows I_1 and I_2 for the two case studies.

The remaining 8 dependent images (I_3 to I_{10}) have been obtained by post-processing I_1 and I_2 through the operators reported in Tab. 4.1. The original

Figure 4.4: *Archetypes of the synthetic case studies. First two images:* I_1 *and* I_2 *for first case study; third and fourth:* I_1 *and* I_2 *for second case study.*

dependency graph (the same for both cases) is shown in Fig. 4.5.

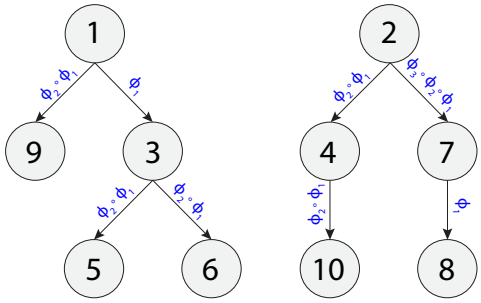


Figure 4.5: *Dependency graph for the synthetic case studies (ground truth).*

		Processing and parameters					
Image	**Parent**	ϕ_1	$\wp_{\phi_1}$	ϕ_2	$\wp_{\phi_2}$	ϕ_3	$\wp_{\phi_3}$
I_3	I_1	JPEG	50	–	–	–	–
I_4	I_2	histogram stretch	–	JPEG	70	–	–
I_5	I_3	rotation	+3	JPEG	100	–	–
I_6	I_3	scaling	1.1	JPEG	100	–	–
I_7	I_2	histogram stretch	–	scaling	1.2	JPEG	90
I_8	I_7	JPEG	60	–	–	–	–
I_9	I_3	rotation	−2	JPEG	100	–	–
I_{10}	I_4	scaling	0.9	JPEG	100	–	–

Table 4.1: *Fundamental-IPFs and their parameters for synthetic case studies.*

(B) Real-world case studies

The first data set, called `Girl`, consists of the 20 near-duplicates of Fig. 4.6 representing the famous painting *The Girl with a Pearl Earring* by Johannes Vermeer (1665 circa). Note that the set also includes some outliers, namely an amateur reproduction of the painting and 3 photos of Scarlet Johansson, the actress who played the role of the Girl in the movie (2003) inspired by the events behind the creation of the painting. All the images are compressed with JPEG quality factors roughly varying from to 70 to 100. Image dimensions vary in the range from 300×400 to 800×1000 pixels.

The second data set, called `Bear`, consists of 55 images representing one of the iconic symbols of the awareness raising campaign on global warming. Some of these images are gathered in Fig. 4.7. The estimated JPEG quality factors vary from 75 to 100, while image dimensions vary from 300×300 to 500×700 pixels. In this very particular case it was possible to obtain the original high resolution picture from the photographer[4] who took it in 2005.

4.4 Experimental setup

We implemented the core of the system in Matlab environment with the registration task delegated to the Java-based image processing software *ImageJ*.[5] This software is designed with an open architecture that is extensible via custom plugins: the one implementing the method in [Arganda-Carreras et al., 2006] is called *bUnwarpJ*.[6] The registration is supported by a plugin computing SIFT features[7] [Lowe, 2004], which ensure higher accuracy.

Although the system features a large number of configuration parameters, there is not a strong dependence upon most of them, with the exception of a small subset that drastically modifies the behaviour of the system. Here, the values of such parameters are simply listed; for a more in-depth analysis we

[4]The author would like to thank Arne Nævra for providing the original version.

[5]The software is freely available for download at `http://rsbweb.nih.gov/ij/`

[6]The software is freely available for download at `http://biocomp.cnb.csic.es/~iarganda/bUnwarpJ/`.

[7]The software is freely available for download at `http://fiji.sc/wiki/index.php/Feature_Extraction`.

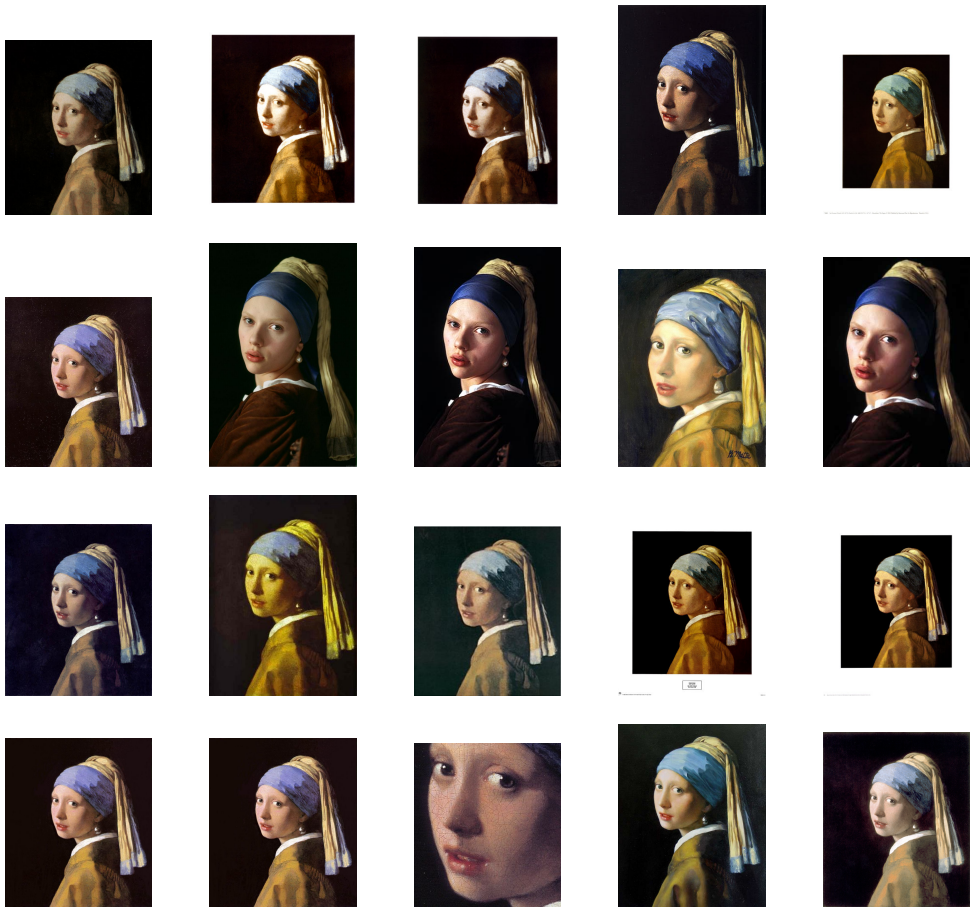


Figure 4.6: *Images composing the* `Girl` *case study.*

refer to the original papers [Reinhard et al., 2001; Arganda-Carreras et al., 2006; Lowe, 2004].

We performed color transfer in the $L\alpha\beta$ space with two differences with respect to [Reinhard et al., 2001]: i) we elaborated linear rather than logarithmic values; and ii) we kept the luminance channel L out of the processing.

The registration tool is the one that required the heaviest testing efforts, because it takes into account not only rigid body operations but also affine transformations and local warping. Given that the application scenario mainly

Figure 4.7: *Selection of images composing the* `Bear` *case study.*

deals with rigid deformations, the parameters of the algorithm have been tuned so to prefer rigid transformations. The used software allows also to perform bi-directional registration to evaluate the consistency between direct and inverse transformations: such feature is not used in this implementation. Tab. 4.2 gathers the values assigned to the various parameters.

Weight	Parameter explaination	Value
w_{div}	Controls the divergence of the deformation field	30
w_{curl}	Controls the curl of the deformation field	30
w_{land}	Controls the error related to SIFT features	1.0
w_{img}	Controls the MSE between image pixels	0.4
w_{cons}	Controls how strictly one deformation is the inverse of the other	0

Table 4.2: *Parameter settings of the image registration tool.*

Finally, the parameters `initial deformation` and `final deformation` determine the level of detail of deformations and refer to the type of grid

used by the algorithm. The choice of their values depends on how misaligned are the images to be registered. In the present implementation we adopted the medium misalignment configuration (`very coarse` and `fine`). Finally, we kept the parameters of SIFT supporting registration to their default values in accordance with [Lowe, 2004].

4.5 Results and discussion

For the synthetic case studies we set the correlation threshold to $T_\rho = 0.5$ as explained in Sec. 4.2. Unfortunately, this value did not seem to lead to equally accurate results for the real-world case studies. The scattergrams corresponding to these cases, in general, show lower values caused by the fact that the processing undergone by Web images is arguably more intense than that we simulated. Being the ground truth unknown, we can not approach the problem as in Sec. 4.2; consequently, we empirically lower T_ρ to 0.4 based on the visual investigation of the results.

4.5.1 Synthetic case studies

The dependency graph of Fig. 4.8 is the same for both the synthetic case studies. Even though the system found all the original relationships between the sets of images, one of them is not correct (dashed edge in Fig. 4.8): while I_7 is declared parent of I_2, it is actually the opposite that is true. Nevertheless, this result is not totally wrong because the system was still able to find that the two images are directly linked to each other. Indeed after applying the threshold, both the links from I_2 to I_7 and from I_7 to I_2 survive, however the latter (wrong) link is stronger than the true one, thus resulting in the reported error. The cause could be the intense processing that I_7 has undergone, since it is the only image originated by the combination of three operators.

4.5.2 The `Girl` case study

It is worth remembering that the ground truth for the dependency graph of real-world case studies is generally unknown. As a consequence, we must evaluate the plausibility of the results by visual inspection. The 20 images

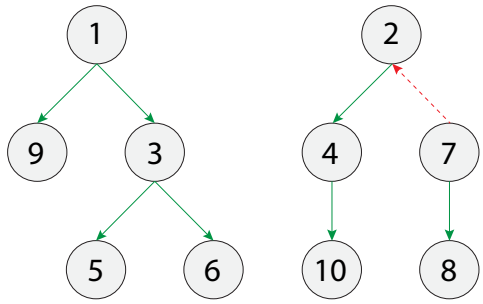


Figure 4.8: *Output dependency graph for synthetic case studies: solid edge: correct link; dashed edge: incorrect link.*

cluster into different groups corresponding to disjoint subgraphs. In Figs. 4.9–4.10, the clusters composed by at least two images are shown.

Consider the most complex cluster, i.e. cluster (A). From a visual inspection we can not say whether I_{12} could have generated I_1 and I_3. However, it is evident that I_{12}'s colors are rather different than those of the original painting, hence manipulations to correct them are highly probable, thus making the link found by the system plausible. Moreover, I_2 appears to be the result of a brightness correction of I_3. Finally, images I_5, I_{14} and I_{15} indeed form a cluster (though the link with I_1 is a bit doubtful): these three images are the only that show a white frame around the painting and some illegible text.

In cluster (B) of Fig. 4.10 (left) there are 3 images, i.e. I_7, I_8 and I_{10}, that are not pictures of the original painting but photos of the movie actress; the system clustered correctly these images and found out that I_7 is the "original": as a matter of fact, I_7's resolution is the highest of the cluster.

As for cluster (C) of Fig. 4.10 (right), it is plausible that I_{17} generated I_{16}: they show exactly the same content and colors but I_{17} has a higher resolution. Finally, it is worth noting that outliers such as the amateur oil reproduction of I_9 (see Fig. 4.6) have not been linked to any image.

4.5.3 The `Bear` case study

We now thoroughly examine the second real-world case study. As we already observed, this data set also contains the archetype (i.e. I_{50}) shown in Fig. 4.12,

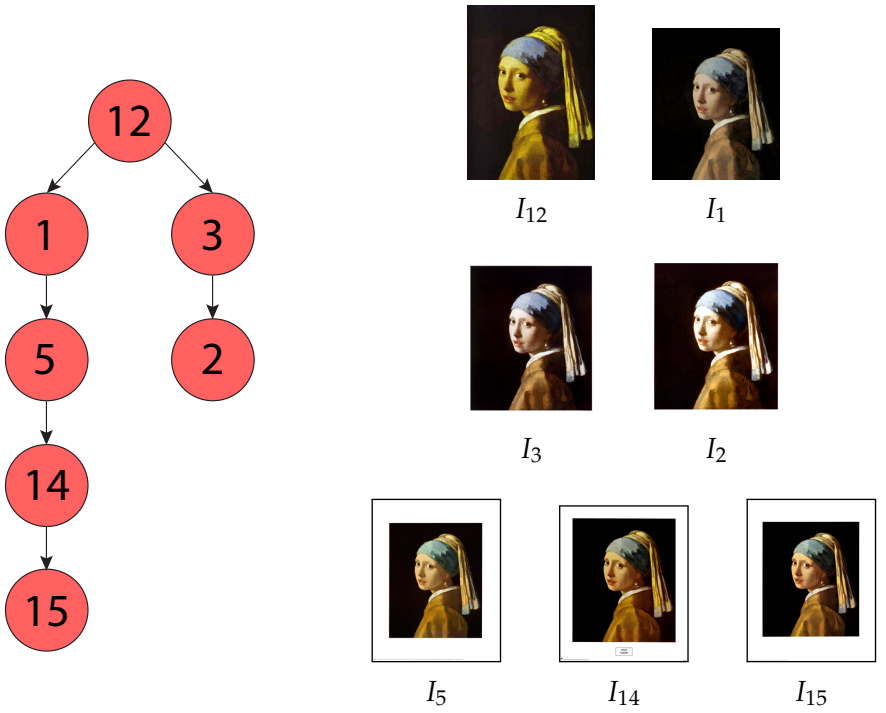


Figure 4.9: *Cluster* (A) *of the* `Girl` *case study.*

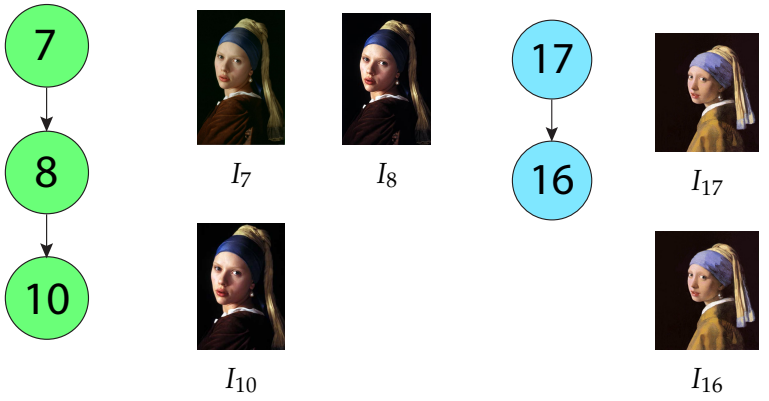


Figure 4.10: *Clusters* (B) *and* (C) *of the* `Girl` *case study.*

provided directly by the photographer who took it. This represents an unique case allowing to draw very interesting conclusions. As a matter of fact, even though we can not use the archetype to assess the veracity of all the links within the data set, it represents the reference for information such as the true image resolution, compression and colors.

Figure 4.11: *The original* `Bear` *image (courtesy of Arne Nœvra).*

We already observed that for case studies like the one we are examining the ground truth is unknown. Therefore, we came up with the following solution to support the soundness of our results: we created five images (I_{51} to I_{55}) from the archetype as described in Tab. 4.3 and we included them into the set. The idea is to use this cluster of images as an indicator (hence the name "*Litmus paper cluster*") of the overall performance of the Dependency Explorer Framework. Although this does not imply that all the links found by the system are valid, if the ground truth images are correctly clustered by the system, then we trust it more willingly. The five images are shown in Fig. 4.13.

The Dependency Explorer Framework found relationships between 44 out of 55 images, defining the 8 clusters of at least two members that are shown

		Processing and parameters					
Image	**Parent**	ϕ_1	$\wp_{\phi_1}$	ϕ_2	$\wp_{\phi_2}$	ϕ_3	$\wp_{\phi_3}$
I_{51}	I_{50}	crop	–	JPEG	85	–	–
I_{52}	I_{50}	scaling	0.7	JPEG	65	–	–
I_{53}	I_{51}	crop	–	JPEG	90	–	–
I_{54}	I_{51}	rotation	20	JPEG	90	–	–
I_{55}	I_{52}	color transfer	–	JPEG	100	–	–

Table 4.3: *IPFs and parameters for the* `Bear`*'s litmus paper cluster.*

in Fig. 4.12 with different colors and labels. The discussion proceeds as follows: firstly, we study the accuracy of the system on the Litmus paper cluster; secondly, we examine the relationships of all the images with the archetype to highlight the closest descendants; then, we examine in depth the remaining clusters labelled from (A) to (G). Even though the majority of the analysed data can be extracted and processed automatically, we also exploit some information that the system is not yet able to retrieve, like presence of text overlays or tampering, to assess the soundness of the results.

Verification of veracity: the *"Litmus paper"* cluster

The system correctly identified all the relationships between the five images composing the *Litmus paper cluster* (Fig. 4.13).

Relationships between the archetype and the rest of the data set

The following analysis does not take into account the "Litmus paper" images as they were artificially created from the archetype for performance evaluation. According to the dependency graph of Fig. 4.12, there are no relationships between the archetype I_{50} and the rest of the images when $T_\rho = 0.4$. Given that it is impossible to collect all possible versions of the image on the Web, the reason could be twofold: the links between the archetype have been severed by an excessive amount of processing; or some images representing the missing

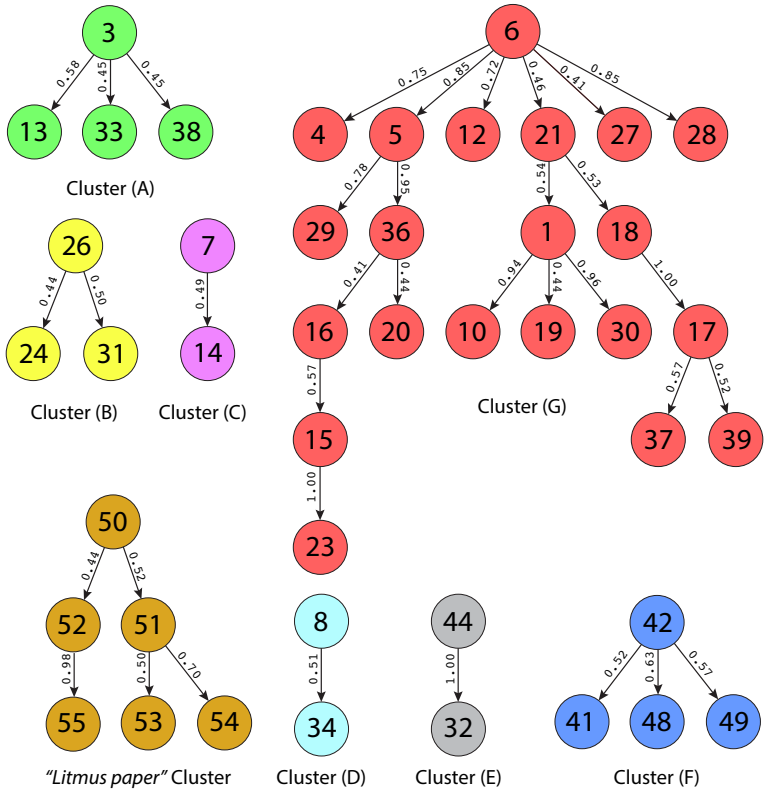


Figure 4.12: *Dependency graph for the* `Bear` *case study.*

links are not present in the data set. This result, however, is not surprising since it is very likely that Web images are scans which should not be linked to the archetype. Nevertheless, it is still interesting to find out which images are closest to the archetype. In Fig. 4.14, the correlation coefficient ρ^* between I_{50} and the rest of the data set is plotted; by lowering T_ρ to 0.3, it comes out that I_3, I_{26}, and I_{38} are the three closest images to the archetype. In particular, $\rho^*(I_{50}, I_{38}) = 0.3967$ is only slightly below the threshold of 0.4. The above images are shown in Fig. 4.15.

The first interesting fact is that image I_3 has a text overlay acknowledging the bear's photographer. By checking the website from which I_3 has been downloaded, it comes out that it belongs to the Natural History Museum of

I_{51} I_{52} I_{53} I_{54} I_{55}

Figure 4.13: *Litmus paper cluster of the* `Bear` *case study.*

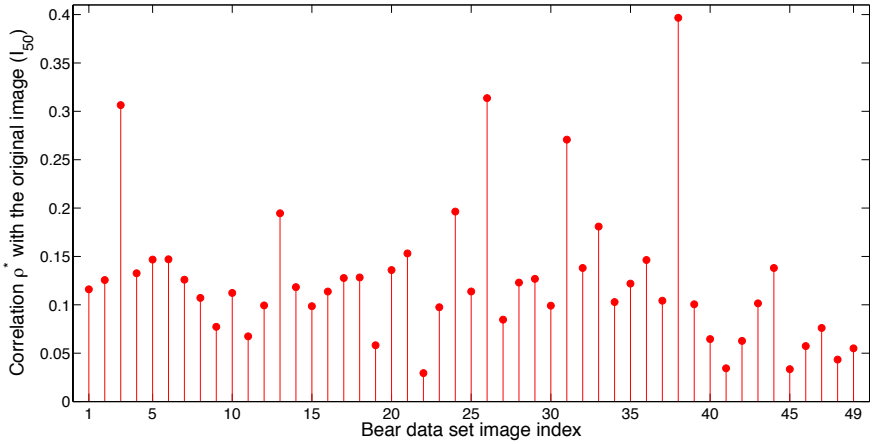


Figure 4.14: *Correlation between I_{50} and the rest of the data set.*

London, which is also selling high quality print-outs. Therefore, it is indeed plausible that I_3 is very close to the original picture. The link with I_{38} is also plausible, as the image appears to be a crop originated from the archetype. Conversely, it is not possible to verify the link with I_{26} by visual inspection, considering the slight difference in colors.

Finally, to highlight the proliferation of several different variants of the original colors, we show the color difference $\overline{\Delta E}$ of Eqs. (4.4)–(4.5) in Fig. 4.16. We will discuss this phenomenon case by case in the following.

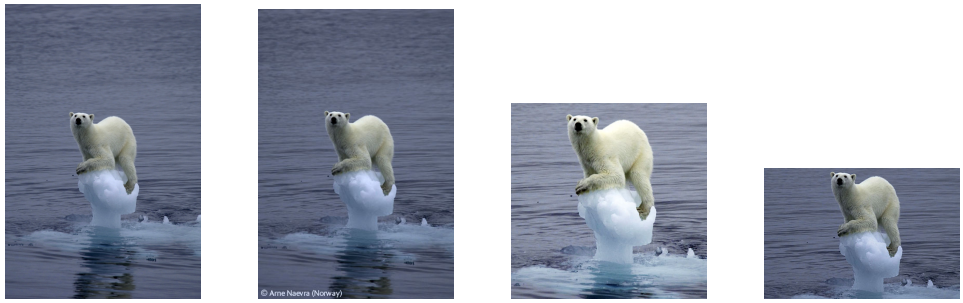


Figure 4.15: `Bear` *data set images closest to the original (leftmost picture).*

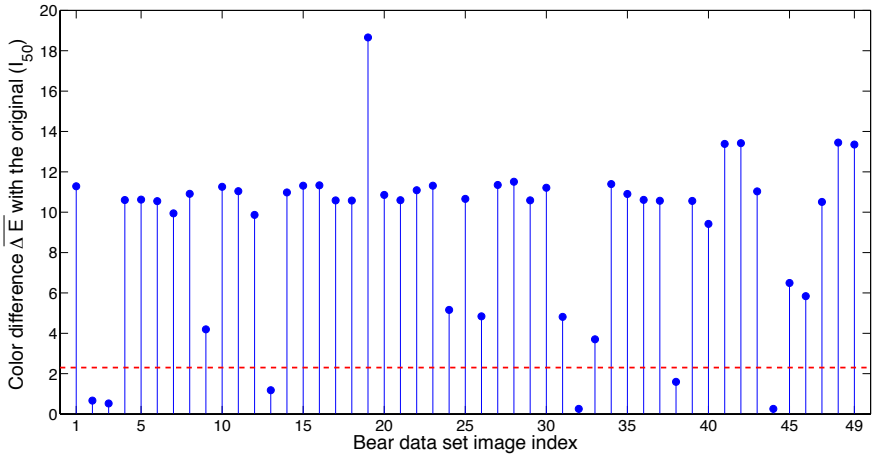
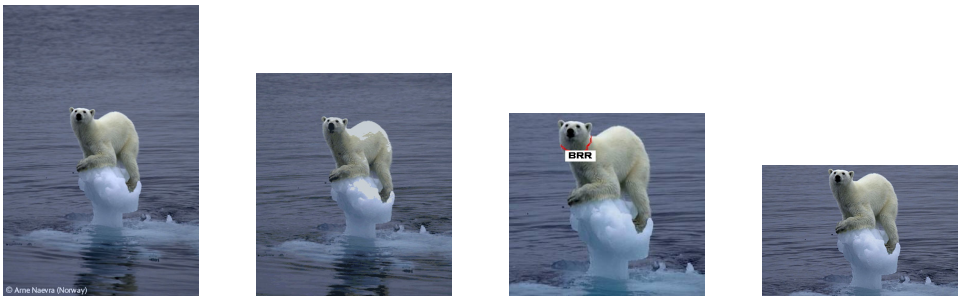


Figure 4.16: *Color difference between the original image I_{50} and the rest of the data set. The red dashed line represents the just-noticeable threshold.*

Analysis of cluster (A)

The root of cluster (A) in Fig. 4.17 is image I_3, which has already been observed to be very close to the archetype. Images I_{13} and I_{33} have the same colors but have been cropped to remove copyright information and to capture only the relevant content into an almost squared region around the bear. This last processing is common also for many images of other subgraphs. Moreover, I_{33} has also been tampered with (see the fake necklace wore by the bear). The link

with I_{38}, on the other hand, is dubious, considering that $\rho^*(I_{50}, I_{38}) = 0.397$ and $\rho^*(I_{50}, I_3) = 0.310$. Since it is not possible to generate I_3 from I_{38} because the latter is a crop, the two images are probably siblings.



I_3 I_{13} I_{33} I_{38}

Figure 4.17: *Cluster* (A) *of the* `Bear` *case study.*

Analysis of cluster (B)

The three images composing cluster (B) in Fig. 4.18 are characterised by brighter colors with respect to cluster (A). I_{24} is a cropped version of I_{26} where the melting ice below the bear has been removed and has the same JPEG quality factor of its parent (85); I_{31} is generated by scaling I_{26} by 0.9 and by recompressing it with a higher quality factor (90).

I_{26} I_{24} I_{31}

Figure 4.18: *Cluster* (B) *of the* `Bear` *case study.*

Analysis of clusters (C) and (D)

Both clusters (C) and (D) in Fig. 4.19 are characterised by darker colors than those of the archetype: the average color difference for each cluster from I_{50} is respectively 9.2 and 10.2. Concerning cluster (C), I_7 contains two text overlays, i.e. an ironic phrase and the website hosting the image, while its child I_{14} has been cropped enough to remove both. Following the crop, I_{14} has been stored with a higher quality factor (85 instead of 75). Concerning cluster (D), I_{34} is another almost squared crop, in this case of I_8, recompressed with higher quality factor (95 instead of 75).

I_7 I_{14} I_8 I_{34}

Figure 4.19: *First and second image: cluster* (C) *of the* `Bear` *case study; third and fourth image: clusters* (D).

Analysis of cluster (E)

The two images in cluster (E) in Fig. 4.20 are actually the same image. The most interesting aspect of this cluster is that although I_{44} and I_{32} descend without any doubt from I_3 (by means of scaling by 0.71 and recompression), the system was not able to find any connections. The reason could be the quality of the two members of cluster (E), which is rather poor with respect to their alleged predecessor, as a closer inspection of the JPEG ringing artefacts around the text can confirm (see the rightmost image of Fig. 4.20). The colors of the cluster are consistent with those of the archetype, as the average color difference with I_{50} is equal to 0.82.

I_{44} I_{32} **text overlay**

Figure 4.20: *Cluster* (E) *of the* `Bear` *case study. Rightmost image: comparison between text overlay of* I_{44} *(top),* I_{32} *(middle) and* I_3 *(bottom).*

Analysis of cluster (F)

The four images composing cluster (F) in Fig. 4.21 are characterised by yet another set of colors that appear to be the result of an intense processing including brightness correction; the average color difference between such images and the archetype is equal to 12.1. The quality factor of the root image I_{42} is rather low (75) and the blocking artefacts are visible in its descendants I_{41}, I_{48} and I_{49} despite their higher factors (respectively 100, 95 and 85).

I_{41} I_{42} I_{48} I_{49}

Figure 4.21: *Cluster* (F) *of the* `Bear` *case study.*

Analysis of cluster (G)

Cluster (G) is surprisingly complex and consists of 21 images linked by several edges with high correlation values. The first thing that comes out is that according to the system all images descend from I_6. As a matter of fact, I_6

is the largest image of the cluster (445×640 pixels) and has JPEG quality factor 100. Among all of I_6's descendants, I_4, I_5, I_{12}, I_{28}, I_{29} and I_{36} have the same size of the root image but quality factors varying between 60 and 95. Furthermore, all the images have similar colors corresponding to the darker version of the picture already observed for clusters (C) and (D).

For the sake of clarity, the graph is subdivided in three parts discussed separately. The first subgraph is depicted in Fig. 4.22. Images I_4 and I_5 have the same size of I_6 but are both compressed with a lower quality factor (60 and 85 respectively); I_{29} is basically I_5 with a text overlay; given the high correlation between them, probably I_5 and I_{36} are the same image; from I_{36} to I_{16} there is the usual crop of the sea content above the bear; I_{20} can be obtained by scaling I_{36}'s size by 0.95 and recompressing it with factor 90; similarly, I_{15} derives from I_{16} following scaling by 0.85 and compression with same quality factor; I_{15} and I_{23} are exactly the same image (same dimensions, quality factor and file size).

According to the second subgraph of Fig. 4.23, I_{21} can be obtained by scaling I_6 by 0.67 and image I_1 derives from I_{21} following the usual crop. Moreover, I_1 and its children I_{10} and I_{30} have the same size of 316×293 pixels. The correlation of I_{10} and I_{30} with I_1 is very high but not maximum because they have been recompressed with lower quality factors (85 and 75 with respect to 100 of the parent). Finally, I_{19} has been generated from I_1 by adding a yellow frame, which caused the peak in color difference with the archetype in the plot of Fig. 4.16.

The third and last subgraph is shown in Fig. 4.24. As already said, I_{21} is a downscaled copy of I_6; I_{28} is the recompression of I_6 with quality factor 85; I_{12} and I_{27} are both crops of I_6; I_{18} is obtained by scaling I_{21} by 0.93; I_{18} and I_{17} are exactly the same image; I_{37} and I_{39} can be obtained by scaling I_{17} by 0.8 and 0.75 respectively.

Tab. 4.4 concludes the analysis by summarising the processing chain that led to each image. For each image the parent is reported, followed by the parameters of the f-IPFs ϕ_j, ϕ_g and ϕ_c. In particular, when scaling or rotation occurred, a number is reported in the column ϕ_g (followed by the symbol $^\circ$ in the second case). Furthermore, color difference $\overline{\Delta E}$ is considered as a

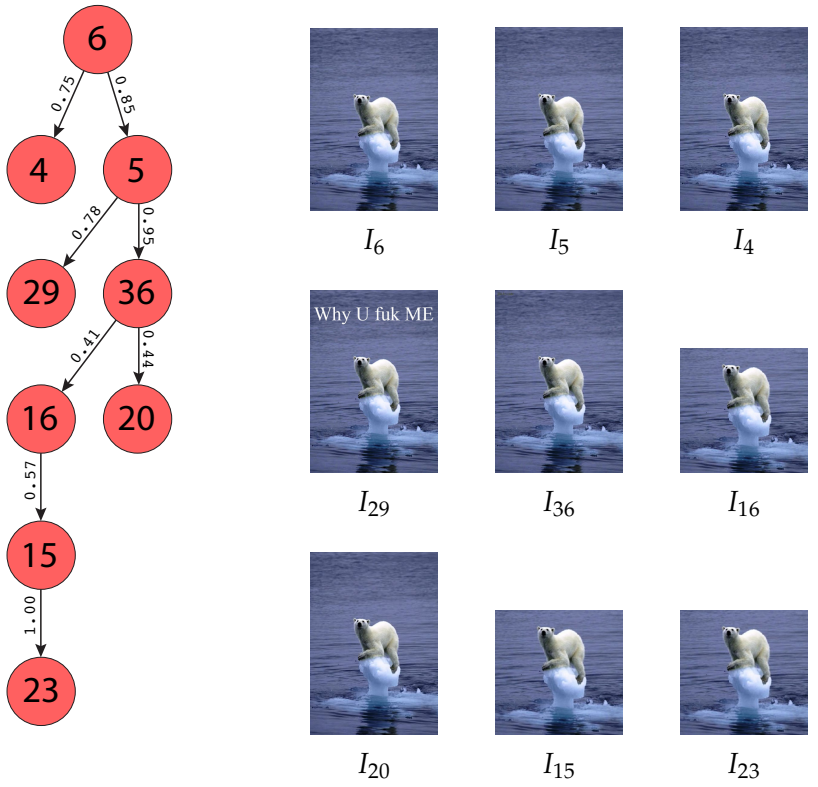


Figure 4.22: *First subgraph of cluster* (G) *of the* `Bear` *case study.*

parameter of ϕ_c hinting the cost of the transformation: very low but not null values, which are not noticeable by the human eye, are probably due to approximations in color conversions rather than to color modifications. Independent images are identified by the symbol $-$, while any transformation that did not occur (e.g. unaltered geometry from parent to child) is identified by the symbol $\perp$. The data in bold font correspond to the ground truth images artificially created to assess the performance of the system.

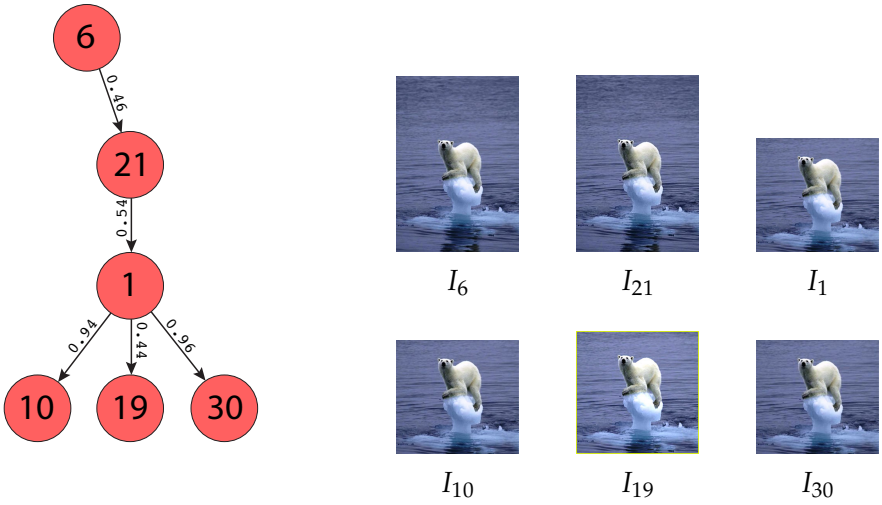


Figure 4.23: *Second subgraph of cluster* (G) *of the* `Bear` *case study.*

4.6 Observations on time complexity

The pairwise checks required by the Dependency Explorer Framework to build the dependency graph grow quadratically with the number of images. Each pairwise check consists of a number of stages, the most important of which are color, geometry and compression matching, whose complexity primarily depends on the image size. Therefore, to give a rough idea about the time complexity of the system, we created three sets of 10 images each by applying the same processing used for the synthetic data sets of Sec. 4.3 (see Tab. 4.1). All the images of each set have a size that is respectively in the order of 1600×1200, 800×600 and 400×300.

We ran the 100 comparisons of each set and we clocked the average time spent on each comparison, which clearly depends on the image size as shown in Fig. 4.25 (left). A comparison between fairly large images (topmost bar) requires about 60 seconds, that is a quite manageable time even when the number of images grows, although the real-time application of the framework becomes rapidly unfeasible. However, even though this is not always the case, according to our experience Web case studies tend be composed of smaller images, thus allowing to dramatically reduce the overall time complexity.

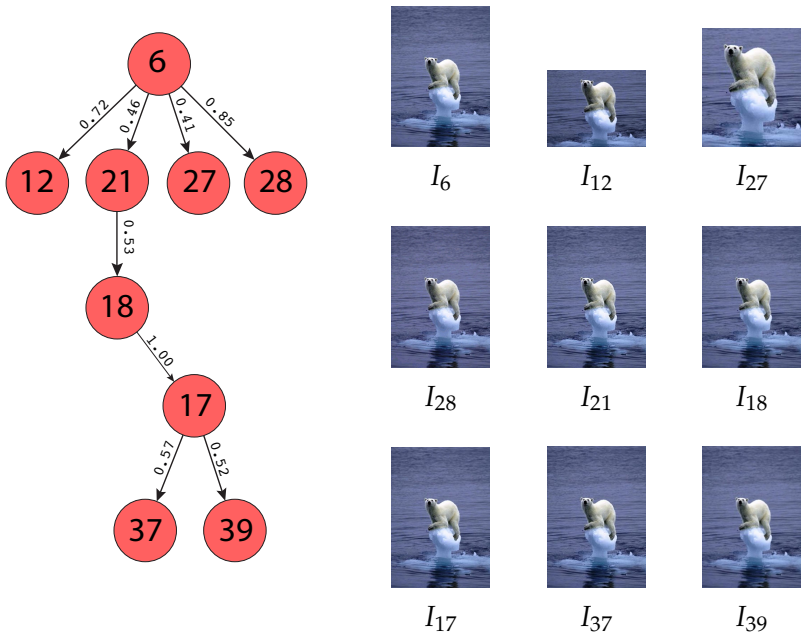


Figure 4.24: *Third subgraph of cluster* (G) *of the* `Bear` *case study.*

Finally, Fig. 4.25 (right) sheds a light on how the execution time is subdivided among the system's various tasks. Expectedly, the process of geometrical matching is very taxing due to the underlying image registration and requires 92% of the total time. More precisely, 43% of such time is spent on estimating the parameters of the geometric transformation leading from source to target image and 49% on actually applying the transformation to the randomnesses of the three image channels. The impact of the remaining tasks including color and compression matching is negligible and amounts to 8% of the total time.

4.7 Concluding remarks

The use of image forensic tools to discover the relationships between groups of images with the same or similar content may find interesting applications in diverse fields, including tracing the illegal distribution of copyrighted images on the Web and understanding how images contribute to the formation and

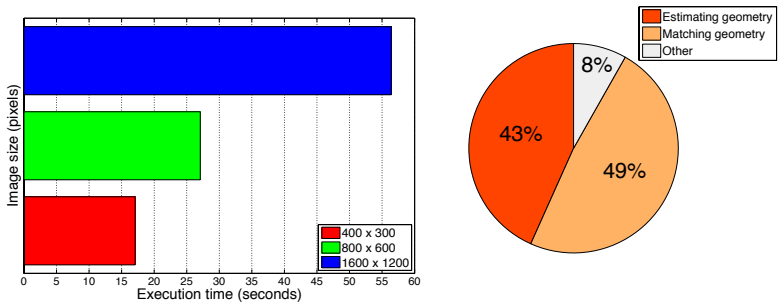


Figure 4.25: *Time complexity of the Dependency Explorer Framework for a single pairwise comparison. Left: average time depending on image size; right: percentage of execution time depending on task.*

evolution of opinions over the Internet. In particular, the latter application opens a new frontier in image forensic research for the difficult challenges it poses both from an image processing and a cognitive point of view. This chapter and the previous one focused on the image processing aspects of the problem by proposing a rigorous formalisation of the problem and by validating it in a simple but realistic scenario. It goes without saying that many difficult challenges are still ahead, including: the development of a theoretically sound formulation of the hypothesis testing problem lying at the heart of the system, going beyond the heuristic aspects of the proposed approach; the definition of an ontology capable of inferring higher order relationships starting from a pairwise analysis; the integration of forensic tools for source and forgery detection to further explore the relationships between images.

		f-IPFs					f-IPFs		
Child	**Parent**	$\wp_{\phi_j}$	$\wp_{\phi_g}$	$\wp_{\phi_c}$	**Child**	**Parent**	$\wp_{\phi_j}$	$\wp_{\phi_g}$	$\wp_{\phi_c}$
$\mathbf{I_1}$	I_{21}	100	crop	1.44	$\mathbf{I_{29}}$	I_5	80	$\perp$	0.97
$\mathbf{I_2}$	–	–	–	–	$\mathbf{I_{30}}$	I_1	75	$\perp$	0.75
$\mathbf{I_3}$	–	–	–	–	$\mathbf{I_{31}}$	I_{26}	90	0.89	0.88
$\mathbf{I_4}$	I_6	60	$\perp$	0.74	$\mathbf{I_{32}}$	I_{44}	$\perp$	$\perp$	$\perp$
$\mathbf{I_5}$	I_6	85	$\perp$	0.75	$\mathbf{I_{33}}$	I_3	95	crop	0.52
$\mathbf{I_6}$	–	–	–	–	$\mathbf{I_{34}}$	I_8	95	crop	0.97
$\mathbf{I_7}$	–	–	–	–	$\mathbf{I_{35}}$	–	–	–	–
$\mathbf{I_8}$	–	–	–	–	$\mathbf{I_{36}}$	I_5	$\perp$	$\perp$	$\perp$
$\mathbf{I_9}$	–	–	–	–	$\mathbf{I_{37}}$	I_{17}	95	0.80	0.69
$\mathbf{I_{10}}$	I_1	85	$\perp$	0.73	$\mathbf{I_{38}}$	I_3	$\perp$	crop	1.73
$\mathbf{I_{11}}$	–	–	–	–	$\mathbf{I_{39}}$	I_{17}	90	0.75	0.70
$\mathbf{I_{12}}$	I_6	95	crop	2.02	$\mathbf{I_{40}}$	–	–	–	–
$\mathbf{I_{13}}$	I_3	90	crop	0.58	$\mathbf{I_{41}}$	I_{42}	75	0.96	0.80
$\mathbf{I_{14}}$	I_7	85	crop	1.57	$\mathbf{I_{42}}$	–	–	–	–
$\mathbf{I_{15}}$	I_{16}	$\perp$	0.84	0.66	$\mathbf{I_{43}}$	–	–	–	–
$\mathbf{I_{16}}$	I_{36}	90	crop	1.41	$\mathbf{I_{44}}$	–	–	–	–
$\mathbf{I_{17}}$	I_{18}	$\perp$	$\perp$	$\perp$	$\mathbf{I_{45}}$	–	–	–	–
$\mathbf{I_{18}}$	I_{21}	$\perp$	crop	0.74	$\mathbf{I_{46}}$	–	–	–	–
$\mathbf{I_{19}}$	I_1	85	crop	2.84	$\mathbf{I_{47}}$	–	–	–	–
$\mathbf{I_{20}}$	I_{36}	90	$\perp$	0.61	$\mathbf{I_{48}}$	I_{42}	85	1.30	0.73
$\mathbf{I_{21}}$	I_6	85	0.67	0.66	$\mathbf{I_{49}}$	I_{42}	95	1.05	0.77
$\mathbf{I_{22}}$	–	–	–	–	$\mathbf{I_{50}}$	–	–	–	–
$\mathbf{I_{23}}$	I_{15}	$\perp$	$\perp$	$\perp$	$\mathbf{I_{51}}$	$\mathbf{I_{50}}$	**85**	**crop**	$\perp$
$\mathbf{I_{24}}$	I_{26}	$\perp$	crop	1.66	$\mathbf{I_{52}}$	$\mathbf{I_{50}}$	**65**	**0.7**	$\perp$
$\mathbf{I_{25}}$	–	–	–	–	$\mathbf{I_{53}}$	$\mathbf{I_{51}}$	**90**	**crop**	$\perp$
$\mathbf{I_{26}}$	–	–	–	–	$\mathbf{I_{54}}$	$\mathbf{I_{51}}$	**90**	**20°**	$\perp$
$\mathbf{I_{27}}$	I_6	90	crop	2.98	$\mathbf{I_{55}}$	$\mathbf{I_{52}}$	**100**	–	**23.1**
$\mathbf{I_{28}}$	I_6	85	$\perp$	1.52					

Table 4.4: *Estimated f-IPFs for* `Bear` *case study: – denotes independent images, $\perp$ absence of processing; when a number is reported for $\wp_{\phi_g}$, the transformation corresponds to scaling or rotation (if degrees are specified, e.g.* 20°*).*

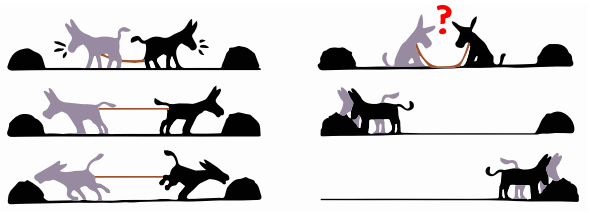
Part II

Decision fusion of Digital Image Forensics detectors

Image forensic research has mainly focused on the detection of artefacts introduced by a single processing tool. In tamper detection applications, however, the kind of artefacts the forensic analyst should look for is not known beforehand, hence making it necessary that several tools developed for different scenarios are applied. Two problems arise in such a scenario: i) devising a sound strategy to elaborate the information provided by the different tools into a single output, and ii) dealing with the uncertainty introduced by error-prone tools. This is a typical task of information fusion, i.e. the process of integrating multiple sources of data representing the same real-world object into a consistent, accurate and useful representation. In this part of the thesis we propose a possible solution to both these problems by introducing a fusion framework based on Fuzzy Theory. Fuzzy systems, in fact, proved to be useful in those applications where reasoning needs to be robust against noise, approximation or imprecise inputs. We describe a practical implementation of the proposed framework putting the theoretical principles in practice. To validate our method, we carried out some experiments addressing a simple realistic scenario in which five forensic tools exploit JPEG artefacts to detect cut & paste tampering within a specified region of an image. The results are encouraging, especially when compared with those obtained with traditional approaches.

Chapter 5

Decision fusion in Digital Image Forensics



INFORMATION FUSION, that is the process of integrating multiple sources of data and knowledge representing the same real-world object into a consistent, accurate and useful representation, has been widely used in several scientific fields and Image Forensics is no exception. Generally, each forensic technique deals with the detection of a typical footprint left by a single processing tool under specific settings. Forensic techniques, however, like any other realistic process or system, are never perfect and their measurements are usually affected by uncertainty, ambiguity or impreciseness. A noisy or unreliable response may have many causes, such as: wrong tool settings; particular characteristics of the analysed images (e.g. color space or type of compression); partial presence (or absence) of the feature(s) the tool is looking for; deviation from the working assumptions of the applied technique.

Another obstacle to overcome when judging the integrity of a given image is that most of the times a tampered image is not the result of the application of a single processing tool. On the contrary, even unexperienced users can create very convincing forgeries by resorting to several tools provided by any imaging software. Since rarely we know beforehand the kind of manipulation the image has undergone, the application of a single footprint detection technique may not be enough, thus requiring the parallel use of more than one technique. A problem with the use of several tools looking for different footprints is

that each tool provides an output describing the degree of presence of the specific footprint it is looking for. Even when using more than one tool, we are interested in obtaining a single global answer allowing to decide whether the image under analysis is authentic or not. Obtaining such a global answer is not a trivial task, as outputs may not only be inaccurate but also heterogeneous; for example, one tool may provide a binary output, another tool a scalar value to be compared with a threshold, while a third tool may output the probability that the image has undergone a certain processing. Moreover, depending on the input image, forensic tools may have technical limitations, be prone to errors or be in disagreement with each other, thus introducing another form of uncertainty. In all these cases traditional techniques such as simple majority voting (an image is tampered if the majority of tools say that the image is tampered) or binary OR (an image is considered tampered if at least one tool says so) may not lead to satisfactory results. This is usually the typical problem one would tackle with by means of Machine Learning approaches like Neural Networks (NN) and Support Vector Machines (SVM), which may provide satisfactory results with two caveats: the complexity of the learning process grows with the number of tools and training needs to be repeated whenever a new tool is added to the set. Since for these methods it is but a short step from computationally challenging to practically unfeasible, it becomes necessary to develop new efficient solutions to keep the uncertainty of different outputs under control while merging them into a single final decision.

So far, very few techniques have been devised in this sense and the fusion framework based on Fuzzy Theory that will be introduced later in this chapter represents one of the first contributions. The chapter is organised as follows. Sec. 5.1 formally introduces the problem of information fusion and reviews a sound categorisation of related approaches that is also valid for Image Forensics. Sec. 5.2 focuses on the techniques developed to address image forensic scenarios. Among such solutions there is the fuzzy-based fusion framework that will be analysed in depth in Chapters 6 and 7. For this reason, the chapter is concluded by the short detour of Sec. 5.3 into the basics of Fuzzy Theory and Logic.

5.1 General pointers to information fusion

In this section we only provide general pointers to the topic of information fusion and we refer to [Kuncheva, 2007] for a detailed discussion. Information fusion (or decision fusion, knowledge integration, expert conciliation, decision combination) is the process of integrating multiple sources of data and knowledge representing the same real-world object into a consistent, accurate and useful representation. This process has been employed in several engineering fields dealing with multiple diverse measurements coming from different instruments. Among these fields there are remote sensing, satellite imaging, biometry and voice/speaker recognition. Consider, for example, biometric applications: to improve authentication accuracy, several data such as fingerprints, iris, gait or voice are normally fused with each other. The interest on information fusion stemmed from the necessity to improve the accuracy in classification problems by merging the results of different classifiers. A great deal of studies were carried out in this direction [Kuncheva, 2007]. Inspired by the work in [Xu et al., 1992], a general categorisation of possible fusion approaches was defined in [Jain et al., 2005] for biometry. Such schematisation is valid also for Image Steganalysis [Kharrazi et al., 2006] and Image Forensics [Barni and Costanzo, 2012b; Fontani et al., 2013]. The classification in [Jain et al., 2005] is sketched in Fig. 5.1, where the dotted rectangles identify those approaches that are not applicable to Image Forensics and that we will not discuss in the following.

Each tool used to gather information on a certain real-world object usually proceeds as follows: firstly, it extracts some descriptive features from the object; secondly, it uses such features to compute a score according to some criteria; and finally, depending on the score, it assigns the object to a certain class. The information coming from multiple tools working on the same object can be fused during either of the previous stages. Therefore, in [Jain et al., 2005], fusion approaches are categorised into *feature level*, *measurement level* and *abstract level*. Note that a wider, higher level categorisation can be made according to whether the fusion is carried out before or after classification. Each category is briefly discussed in the sequel.

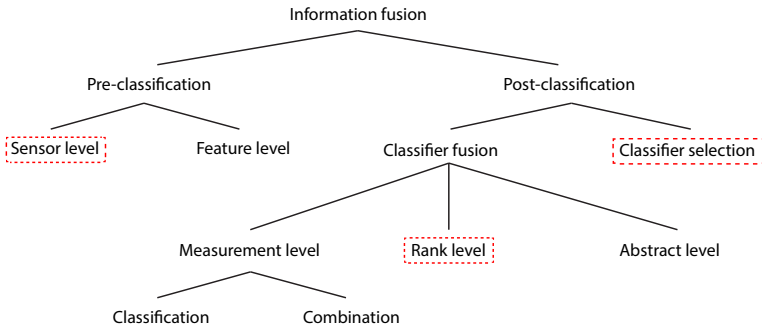


Figure 5.1: *Categorisation of information fusion techniques according to* [Jain et al., 2005]. *Dotted rectangles denote approaches not applicable to Image Forensics.*

(A) Fusion at feature level

This is potentially the more meaningful integration, since it directly deals with richer, highly distinctive and unprocessed information. The method consists in aggregating the features provided by the various tools before feeding them to a single classifier. The idea underlying it is that a higher discriminative power can be achieved by joining different features. More specifically, consider a set of K tools, each of which extracts a feature vector f_k. If all such vectors are homogeneous and their length is the same, their fusion can be carried out by simply combining them, e.g. with the weighted summation $f^* = \sum_{k=1}^{K} w_k f_k$, where $\sum_{k=1}^{K} w_k = 1$. Conversely, if the various vectors are not consistent with each other, they can be concatenated, i.e. $f^* = [f_1, f_2, \dots f_K]$. In both cases the classification is carried out on f^*. Typical problems that may arise at this level include conflictual, redundant or high dimensional features requiring an additional and computationally heavy stage of feature selection, whereby the less discriminative elements of f^* are discarded before the vector is fed to the classifier.

(B) Fusion at measurement level

Fusion at measurement level consists in combining the scores computed independently by each tool by relying only on its own features. It can be formalised

as follows. Given a feature vector $\mathbf{x}$, each classifier $\mathcal{D}_i$ produces a c-dimensional vector $[d_{i,1} \dots d_{i,c}]$, where $d_{i,j}$ represents the support for the hypothesis that $\mathbf{x}$ belongs to class j. Fusion at measurement level obtains a single final score by relying on all the partial vectors.[1]

This kind of fusion is generally less demanding than operating at the feature level, due to the smaller amount of data to deal with but, on the other hand and for the same reason, it is more sensitive to noise and uncertainty. Methods belonging to this category proceed either by means of *classification* or *combination*. Classification techniques build a vector with partial scores and feed it to a binary classifier (e.g. accept or reject, stego or cover image, authentic or forged image). Among such methods there are SVM,[2] neural networks and decision trees. The fact that information does not need any processing before being fed to the classifier represents the main advantage of these approaches. On the other hand, they are affected by the so called "curse of dimensionality", i.e. the rapidly increasing complexity of the learning process occurring when the number of tools grows. The above problem afflicts even more the fusion at the feature level.

Combination techniques aggregate the partial scores into a single score based on which a final decision is made usually by means of thresholding. Scores can be aggregated with linear combinations [Jain et al., 2005], min, max, mean, median or product rules [Kittler et al., 1998], Bayesian models [Domingos and Pazzani, 1997] and non-traditional approaches such as Fuzzy Theory [Chatzis et al., 1999] and Dempster-Schafer Theory of Evidence [Lu, 1996]. Regardless of the chosen approach, the problem with this kind of fusion is that all the partial scores must be represented in a consistent format, thus requiring a preliminary stage of transformation into a common domain.

[1] For instance, in an image forensic scenario each classifier i could produce vectors $[d_{i,0}, d_{i,1}]$, where classes $\mathcal{C}_0$ and $\mathcal{C}_1$ indicate authentic and forged images. In this case all the vectors would be merged into a final score $[d_0^*, d_1^*]$.

[2] In the case of SVM an additional, nested fusion can be carried out, namely *kernel fusion*: the score vectors are classified by means of different kernels, the best of which are used to create a hybrid kernel for the final classification.

(C) Fusion at abstract level

Fusion at abstract level is carried out by first letting each classifier $\mathcal{D}_i$ independently assign a class label ℓ_i to the feature vector $\mathbf{x}$ and then by taking a decision on the final class based on all the assigned labels. Since by definition any classifier is capable of assigning a label, contrarily to other categories the fusion at abstract level is universal. However, the information available at this point is limited and does not reflect the uncertainty on the predicted labels. Usually the task is carried out by thresholding the single decision values (or any measure acting as them) and then by aggregating the so obtained binary outputs by means of majority voting (i.e. giving power to the classifiers agreeing with each other), weighted voting (i.e. giving power to the most competent classifiers) [Lam and Suen, 1997] or AND/OR rules.

5.2 Information fusion in Image Forensics

We observed in Chapter 2 that image forensic tasks like source identification or forgery detection can be cast as classification problems. As a consequence, the outputs provided by the detectors can be fused either at feature, measurement or abstract level.

(A) Fusion at feature level

Without any doubt, the most common fusion is the one at feature level, which is generally used to improve the accuracy of source detection. A few techniques are listed in the following [Sencar and Memon, 2013], even though it is not the aim of this section to explain their working principles.

- **Camera model identification**. In [Celiktutan et al., 2008] similarity measures, image quality metrics and Wavelet coefficients statistics are fused to identify mobile phone camera models. Similarly, the three techniques in [Xu et al., 2009], [Kharrazi et al., 2004] and [Gloe et al., 2009] discriminate between different camera brands and models. The first technique merges 390 statistical moments characterising the image spatial representation, JPEG representation, and pixel co-occurrences;

both the second and third techniques fuse features including color information, quality metrics and wavelet statistics.

- **Scanner model identification**. In [Gou et al., 2007], 60 noise statistics features coming from different denoising algorithms are used to discriminate between different scanner brands and models; the same task is fulfilled in [Khanna and Delp, 2010] on images containing text by fusing gray-level co-occurrences and difference histograms.

- **Device class identification**. In [McKay et al., 2008] the fusion of noise statistics and color interpolation coefficients allows to discriminate between various image sources; different noise statistics are also fused in [Fang et al., 2009] and [Khanna et al., 2007a] with the same goal.

- **Forgery detection**. [Chetty and Singh, 2010] investigates the problem of detecting copy-move forgeries carried out on frames of compressed, low quality video sequences streamed from the Web. In practice, the technique fuses the features computed by two copy-move detectors in the hypothesis of double JPEG quantisation altering the natural correlation existing between forged image pixels. Such features correspond to noise [Hsu et al., 2008] and quantisation residues [Fridrich et al., 2003a]. Firstly, all the features are independently extracted from 32×32 pixel blocks belonging to each intra-frame of the analysed video; then, only those features whose significance is relevant are selected by means of various methods, including Independent Component Analysis (ICA) and normalised. At this point, the two feature vectors are fused by means of the Sugeno's fuzzy integral [Sugeno, 1974], which produces a crisp measure qualifying the frame under analysis as authentic or copy-moved. In their experiments, the authors demonstrate the superiority of the fusion of the two detectors with respect to the separate usage of each of them.

(B) Fusion at decision level

To the best of our knowledge, so far only two works address information fusion at measurement level: the one in [Barni and Costanzo, 2012b] based on Fuzzy Theory [Zadeh, 1965]; and the one in [Fontani et al., 2013] based

on Dempster-Schafer Theory of Evidence [Shafer, 1976]. The similarities in the formalisation of the two approaches are explained by the fact that both have been developed within the same research group; both approaches, in fact, propose a general fusion framework addressing the scenario in which heterogeneous tools detect image splicing. The process of fusing the outputs provided by the algorithms is carried out by combining the partial scores by means of two different extensions of traditional logic whose rationale is hinted in the sequel. Since the fuzzy-based framework represents the principal contribution of this part of the thesis, it will be thoroughly explained and validated in Chapters 6 and 7. Dempster-Shafer's (DS) Theory of Evidence is a framework for reasoning under uncertainty that allows the representation of ignorance and available information in a more flexible way with respect to Bayesian theory. The Bayesian framework often urges to apply insufficient reasoning to assign a-priori probabilities, thus introducing extraneous assumptions. Dempster-Shafer's theory, instead, abandons the classical probability frame and allows to reason without a-priori probabilities through a new formalism. Specifically, it allows to combine evidences coming from different sources, interpreting them as "belief" on propositions, and provides a formalism for turning logical operations on propositions into operations among sets.

Despite their common background, a direct comparison between the techniques in [Barni and Costanzo, 2012b] and [Chetty and Singh, 2010] is precluded by the fact that they work at incompatible levels (respectively decision and feature level). On the contrary, a comparison between earlier versions of the methods in [Barni and Costanzo, 2012b] and [Fontani et al., 2013] has been presented in [Fontani et al., 2012]. It came out that fuzzy decision fusion slightly outperforms Dempster-Shafer fusion for low probabilities of false alarm; moreover, both methods perform better than a learning method relying on SVM (see Sec. 7.3.4 for more on this topic).

(C) Fusion at measurement level

Recently, [Cozzolino et al., 2013] extended the fusion of digital image forensic outputs to the abstract level. The approach adopted by the authors is very similar to those in [Barni and Costanzo, 2012b] and [Fontani et al., 2013]. Five tools, three of which already considered in the latter two works, are used to

detect cut & paste forgeries; then, their outputs are fused at abstract level by means of weighted majority voting, Behaviour Knowledge Space and Naive Bayes and at measurement level by means of product, sum and Dempster-Shafer rules. The experimental results obtained by the authors confirm that the fusion process ensures a significant improvement in performance with respect to using each tool separately.

5.3 Foundations of Fuzzy Theory

We now briefly introduce the most important concepts behind Fuzzy Theory and Logic allowing the reader to fully understand the working principles of the fuzzy-based decision fusion framework of the next chapters. It is not the aim of this section to provide an exhaustive overview of all the aspects of Fuzzy Theory, for which we refer to [Zadeh, 1965; Sugeno, 1985; Terano et al., 1992].

Fuzzy sets theory was conceived in 1965 by Lotfi Zadeh as an extension of classic set theory [Zadeh, 1965]. From this initial concept, a multi-value fuzzy logic has been derived in subsequent years as an extension of Boolean logic. According to Zadeh, the main rationale behind fuzzy logic is the observation that despite people do not require precise, numerical information input for their reasoning, they are capable of highly adaptive control. If such capability could somehow be transferred to systems, they would perhaps be more effective and easier to implement. Moreover, he also claimed that "*as the complexity of a system increases, our ability to make precise and yet significant statements about its behaviour diminishes until a threshold is reached beyond which precision and significance become almost mutually exclusive characteristics*" [Zadeh, 1973]. Fuzzy logic was designed to deal with imperfect information, which in the real world is more often the norm than the exception. Zadeh defined *computing with words* the methodology of dealing with incomplete, unreliable or partially true information.

In order to understand the way fuzzy logic works, three concepts must be introduced: fuzzy sets, fuzzy operators and if-then rules. In the following each of such concepts is briefly described.

5.3.1 Fuzzy sets

Let $\mathcal{X}$ be the universe set and $\mathcal{C} \subseteq \mathcal{X}$ be a set contained in $\mathcal{X}$; then $\mathcal{C}$ can be represented by its characteristic function:

$$\mu_{\mathcal{C}}(x) = \begin{cases} 1 & \text{if } x \in \mathcal{C} \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

Sets characterised as in Eq. (5.1) are also called crisp sets. Fuzzy set theory extends this concept. A fuzzy set $\mathcal{F} \subseteq \mathcal{X}$ is defined through the following generalised characteristic function:

$$\mu_{\mathcal{F}}(x) : \mathcal{X} \rightarrow [0, 1]. \tag{5.2}$$

The values of such a function are continuous in $[0, 1]$ rather than limited to $\{0, 1\}$. The function $\mu_{\mathcal{F}}(x)$ is called *membership function* and associates to each element $x \in \mathcal{X}$ a grade of membership, that is a real number in the interval $[0, 1]$.

In Zadeh's framework, an element x can belong (or not belong) to a given fuzzy set $\mathcal{F}$ with a certain grade of membership. Let, for instance, X be the space of all temperatures. In classical set theory a temperature x is either hot or not hot. In fuzzy theory, x can be at same time hot and not hot with degrees $\mu_{hot}(x)$ and $\mu_{\overline{hot}}(x)$. A value of $\mu_{hot}(x)$ near 1 indicates a high degree of membership of x in the fuzzy set `hot`, a value near 0 a high degree of membership in $\overline{\texttt{hot}}$.

Similarly to classic sets, Zadeh has defined operations like intersection, union and complement to be applied to fuzzy sets [Zadeh, 1965]. Let A and B be two fuzzy sets and $\mu_A(x)$, $\mu_B(x)$ their membership functions. The basic set-operators can be generalised as follows:

$$\begin{aligned} \mu_{A \cap B}(x) &= \min(\, \mu_A(x), \mu_B(x) \,) \\ \mu_{A \cup B}(x) &= \max(\, \mu_A(x), \mu_B(x) \,) \\ \mu_{\bar{A}}(x) &= 1 - \mu_A(x). \end{aligned} \tag{5.3}$$

By relying on the operations defined by Eq. (5.3), Zadeh also demonstrated the validity of the basic properties of crisp sets operations like commutativity, associativity, distributivity and De Morgan's law.

5.3.2 Fuzzy operators

By interpreting the values of membership functions as truth values, it is possible to extend the concepts of fuzzy sets theory so that a multi-valued fuzzy logic is obtained [Zadeh, 1965]. Classic Boolean logic requires that a proposition is either true (1) or false (0): there are no other possible values to assign. Based on real world experience, fuzzy logic can claim that a proposition is not always totally false or totally true but true or false to some grade in the interval $[0, 1]$. Doing so, it is possible that a proposition is true, *more or less* true, *somewhat* true and so on.

Since Boolean logic can be seen as a particular case of fuzzy logic where one can only assign values 0 and 1 to membership functions, the extension of logical operators is quite simple and intuitive. In a nutshell, fuzzy AND, OR and NOT can be obtained directly from Eq. (5.3) by parallelising these operators respectively to intersection, union and complement as follows:

$$\begin{aligned} \mu_{A \wedge B}(x) &= \min(\, \mu_A(x), \mu_B(x) \,) \\ \mu_{A \cup B}(x) &= \max(\, \mu_A(x), \mu_B(x) \,) \\ not(A) &= 1 - \mu_A(x). \end{aligned} \tag{5.4}$$

5.3.3 If-then statements

If-then rules are the basic instructions that permit to define the behaviour of a system by means of commands that are easily understandable by a machine. The definition of such rules is a critical step in the process of building a fuzzy control system. It is the aim of if-then rules to establish a linguistic relationship between the description of a situation and an action to be performed. A simple example of this kind of relationship could be the rule *if obstacle is too close then reduce speed to low*. More specifically, if-then rules are based on the fuzzy logic principles outlined above and define how fuzzy sets and logic operators interact with each other by means of membership functions.

More rigorously, let $x_1, .., x_n$ and $y_1, .., y_m$ be fuzzy variables (i.e. variables that can assume as value the label of a fuzzy set) and let $A_1, .., A_n$ and $B_1, .., B_m$ be fuzzy sets. An if-then rule can be defined as follows:

$$\begin{array}{lllllllll} \text{IF} & x_1 \text{ is } A_1 & \wedge & x_2 \text{ is } A_2 & \wedge & \ldots & \wedge & x_n \text{ is } A_n \\ \text{THEN} & y_1 \text{ is } B_1 & \wedge & y_2 \text{ is } B_2 & \wedge & \ldots & \wedge & y_m \text{ is } B_m. \end{array} \tag{5.5}$$

The first part of the rule (introduced by IF) is called antecedent or premise; the second part (introduced by THEN) is called consequent or conclusion; the rule itself is called implication. While the structure of the antecedent is quite standard, a consequent can be defined in different ways (see [Sugeno, 1985; Mamdani and Assilian, 1975]). The consequent form used in Eq. (5.5) is compliant with Mamdani's model [Mamdani and Assilian, 1975] and represents the most common methodology in fuzzy applications due to its simplicity. It is also possible to construct compound rules by means of simple or nested conditional structures such as "if-then-else", such as for example:

$$\text{IF } x_1 \text{ is } A_1 \text{ THEN (IF } x_2 \text{ is } A_2 \text{ THEN } y_1 \text{ is } B_1 \text{ ELSE } y_2 \text{ is } B_2). \tag{5.6}$$

Structures as in Eq. (5.6) can always be decomposed in a set of basic if-then rules as in Eq. (5.5) [Sivanandam et al., 2007]. Even though the number of expressions composing a rule is arbitrary, expressions belonging to the antecedent and to the consequent are combined separately. Regardless of the model, in most cases the adoption of one rule only is not effective: there is need of a set of two or more rules that can play off one another, so that a system can react correctly to a large number of situations.

5.3.4 Fuzzy inference systems

The basic concepts introduced so far are the building blocks of Fuzzy Inference Systems (FIS). Simply put, a fuzzy inference system is nothing else but a set of fuzzy rules that convert inputs into outputs. More specifically, Fig. 5.2 shows the five blocks composing a FIS: i) a database of fuzzy if-then rules; ii) a database of membership functions and fuzzy sets; iii) a Fuzzification Interface converting crisp input values into fuzzy entities; iv) a Decision-Making Unit performing the reasoning based on rules; and v) a Defuzzification Interface converting the reasoning into a crisp output.

Each of the blocks contributes to the interpretation of a set of if-then rules as in Eq. (5.5) as follows. When crisp inputs are fed to the system, the *Fuzzification Interface* begins by converting them to fuzzy sets. More specifically, it assigns a degree of membership to each input by relying on the membership functions and on the fuzzy sets stored in the system's database. The outputs of the Fuzzification Interface are fuzzy values according to which a certain

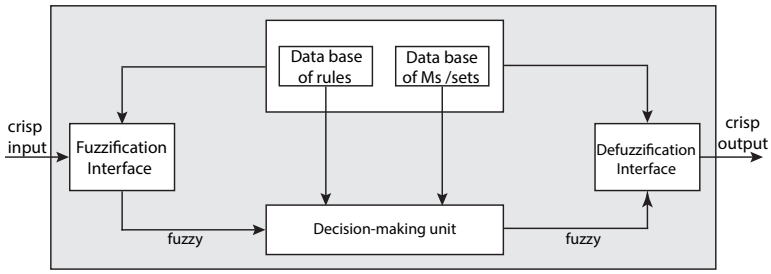


Figure 5.2: *General scheme of a Fuzzy Inference System.*

decision must be taken. This is the core task of the actual reasoning and it is carried out by the *Decision-Making Unit*. All the degrees of membership are combined with the rules of behaviour. Specifically, if the antecedent consists of more than one term, the fuzzy logic operators defined in Eq. (5.3) are applied to resolve the antecedent into a single value called *degree of support* (or strength) of the rule. The degree of support is used to shape the output fuzzy set. The consequent of a rule, in fact, assigns to the output an entire fuzzy set that is truncated according to the degree of support of the rule. Usually a fuzzy system features several rules, each of which contributes with its own truncated output set. However, to make a decision one needs to look at a single output fuzzy set, thus requiring some kind of aggregation procedure. The most common method of aggregation is the max criterion. The output of the Decision-Making Unit is a fuzzy set that a system typically cannot directly use to make a final decision. Therefore, a process of conversion from fuzzy quantities to a crisp global value is required. This is the task of the *Defuzzification Interface*; such process can be performed in several different ways [Leekwijck and Kerre, 1999], the most common of which is the centroid method (also referred to as center of gravity or center of area).

In the next chapter all these stages will be explained in detail and exemplified in the context of Image Forensics.

Chapter 6

A fuzzy fusion framework for Image Forensics

"The color of truth is gray"

André Gide

In this chapter we present a new framework for the fusion of the outputs produced by forensic algorithms [Barni and Costanzo, 2012b].[1] By relying on fuzzy logic principles, such framework allows the cooperation of multiple tools by overcoming typical problems such as heterogeneous outputs, noise and error proneness. Fuzzy logic has demonstrated to be useful in those applications that need to be robust against noise, approximate or imprecise inputs [Terano et al., 1992; Sugeno, 1985]. For this reason, a system based on it may also help to deal with the incomplete or conflicting outputs provided by different forensic algorithms and to resolve them into a single final value. Moreover, one of the main advantages of fuzzy logic is the capability to address problems whose mathematical or statistical models are hard to define. In this way one may design automated frameworks that resort to the experience and the knowledge of human operators to mimic their behaviour. Imagine how a forensic analyst would face the problems of uncertainty and fusion: first off all, she would tweak the tools at her disposal by gathering as much information as possible (e.g. which ones are the most trustworthy, on what kind of images they work, how they interact with each other), thus tackling with the uncertainty problem. Then, she would run all the tools on the image under analysis and exploit the previously gathered knowledge to make a final decision, thus tackling with the fusion problem.

[1] This work extends the previous study in [Barni and Costanzo, 2012a].

The proposed framework translates into practice the above rationale by means of basic fuzzy logic principles. Several advantages are brought by its usage: it outperforms canonical approaches like SVM and logical disjunction; it is easily scalable and allows incremental addition of new forensic tools without requiring new training (as opposed to Machine Learning solutions); it does not require mathematical models; it addresses the fusion problem in a sound yet intuitive way. To the best of our knowledge, the usage of fuzzy logic to address the problem of uncertainty in Image Forensics has been very limited in the past, given that the only technique proposed so far is the one in [Chetty and Singh, 2010]. A direct comparison with such method is however not possible, since the two frameworks work at two incompatible levels according to the categorisation in [Jain et al., 2005].

We divide the formalisation of this chapter into two stages: Sec. 6.1 addresses all the non-fuzzy aspects regarding the gathering of knowledge on the employed algorithms; based on such knowledge, Sec. 6.2 explains how fuzzy logic can be used to automatically mimic the forensic analyst's reasoning. We use several examples to support our theoretical results, one of which is examined in detail in Sec. 6.3.

6.1 Gathering information about employed tools

We now formalise the problem of fuzzy-based information fusion in Image Forensics according to the outline below:

- definition of a common output format for forensic tools;
- definition of tools' behaviour depending on manipulations;
- design of membership functions of the fuzzy system;
- conversion of tools' outputs into inputs of the fuzzy system;
- derivation of fuzzy inference rules.

It is worth noting that only the last three tasks involve fuzzy set theory and logic, whereas the first two are carried out before the inference system is built and represent the knowledge passed to the system by the forensic analyst.

6.1.1 Output of the image forensic tools

Let $\mathcal{T}$ be a set of K forensic tools for detecting whether an image I is tampered or not. Since every forgery is described by at least one footprint, each tool $t_i \in \mathcal{T}$ analyses a feature set in I and generates an output quantifying its presence. At the end of the detection stage, K outputs are produced.

The first step of every combination technique working at measurement level consists in setting a common format for the partial scores. In this case each output consists of a pair of values (D, R) as in Definitions 1 and 2.

Definition 1. *Let $t_i \in \mathcal{T}$, $i = 1..K$, be a forensic algorithm. $D_i \in [0, 1]$ is the degree of detection, i.e. a measure of the presence of the tampering trace(s) that t_i is looking for.*

Definition 2. *Let $t_i \in \mathcal{T}$, $i = 1..K$, be a forensic algorithm. $R_i \in [0, 1]$ is the degree of reliability, i.e. a measure of the confidence of t_i in the provided detection value D_i.*

D does not necessarily need to be a probability and normally changes from image to image; R can either be a constant value depending only on the overall performance of the tool or vary according to the image characteristics (e.g. size, color, format, visual content). In order to define R, some information about the tool's performance, drawn either from theory or experimental analysis, is required. The hypothesis of knowing such information does not affect the generality of the formalisation, since the research community is well accustomed to benchmarking forensic algorithms upon releasing them. The concept of reliability, in fact, is strongly related to the behaviour of the tool "on the field": the more the tool performs correctly (e.g. assigning an image to the right class, or localising accurately a manipulation), the more it is reliable. Therefore, as long as the analyst can express it with a value in $[0, 1]$, R can be derived from any information on performance. For instance, R could be derived from the area under the ROC curve of the tool or could depend on JPEG quality factor or image size, if these properties somehow influence the tool's output. In the following we explain how the analyst makes use of reliabilities to merge the detection scores.

6.1.2 Behaviour of the forensic algorithms

Now that we established a common domain for each tool, the fusion engine must be informed about their expected behaviour and mutual relationships. In other words, we are interested in defining how the tools are supposed to react when a certain known manipulation is carried out on the image. The conclusions drawn at this stage represent the knowledge that the human operator will transfer, in the form of rules, to the fuzzy system.

Suppose that all K tools are working in ideal conditions, without any disturbance from external factors reducing their effectiveness; in this favourable state, each tool always provides the correct answer on the authenticity of a given image. As a consequence, it is possible to define the expected output of each tool of the set.

Depending on the nature of the manipulation, a tool may or may not be able to identify a region as tampered. Let us indicate the capability of detecting the tampering with Y and the incapability with N. Therefore, if we have K tools, each kind of manipulation (or absence of manipulation) is identified by one or more K-dimensional sequences of Y and N. In other words, for each manipulation m we answer to the following questions: can tool t_1 detect m? Can t_2? ... Can t_K? By doing so, to each manipulation we can associate an array that we interpret as the expected (*ideal*) collective answer of the tools when there is no external disturbance. Clearly, also the absence of tampering has its own array(s).

It is convenient to organise such arrays into two different tables, one for the ideal answers in presence of tampering and one for the ideal answers in absence of tampering. In the following we will use the symbols T_{TRUE} and T_{FALSE} for such tables; similarly, we will use the symbol T_{DOUBT} to indicate the table of unexpected (*non-ideal*) K-uples of tools' outputs belonging neither to T_{TRUE} nor to T_{FALSE}. Since the definition of these tables depends on the tools and is based on the knowledge of their performances, in the following we assume that they are always available.

To clarify the above concepts, consider the toy example of a set of two tools $\mathcal{T} = \{t_1, t_2\}$. Assume that t_1 (t_2) considers a region within an image as tampered if traces of aligned (misaligned) double compression are found. The

ideal tables corresponding to this simple scenario are shown in Tab. 6.1, whose columns are explained in the following. If the tools analyse a region to which an aligned double compression was applied, the sequence (Y, N) is expected; on the contrary, if the region has undergone misaligned double compression, (N, Y) is expected; if the region has not been tampered with, then there will not be traces of compression, neither aligned or misaligned; therefore, (N, N) is expected; finally, if the sequence (Y, Y) is produced, something went wrong, at least partially.

	Tampering tables			
Tool	T_{TRUE}		T_{FALSE}	T_{DOUBT}
t_1	Y	N	N	Y
t_2	N	Y	N	Y

Table 6.1: *Toy example of ideal behaviour of the tools.*

6.2 Introducing the role of fuzzy inference

In this section we translate the knowledge of the human operator, in the form of tables T_{TRUE}, T_{FALSE} and T_{DOUBT}, into ideal and non-ideal rules intelligible to a fuzzy inference system.

6.2.1 Definition of fuzzy sets and membership functions

There are various methods to define the appropriate membership functions, either based on human understanding of the underlying problems, on specific algorithms or logical procedures. Nevertheless, the soundness of our formalisation does not depend on the shape of membership functions, which are strongly related to the application at hand and can not be generalised.

Intuitively, detection and reliability can either be considered `low` or `high`, where by low and high we mean fuzzy sets characterised by membership functions in contrast with each other. Similarly, we can assign different degrees of intensity to the presence of tampering derived from pairs (D, R) of all tools. In this thesis we defined five fuzzy sets for the presence of tampering: `very weak`, `weak`, `neither weak nor strong`, `strong` and `very strong`. Even though we

will not put these principles in practice until Chapter 7, a visual example of possible membership functions is depicted in Fig. 6.1, to provide the reader with a rough idea of the concepts discussed above.

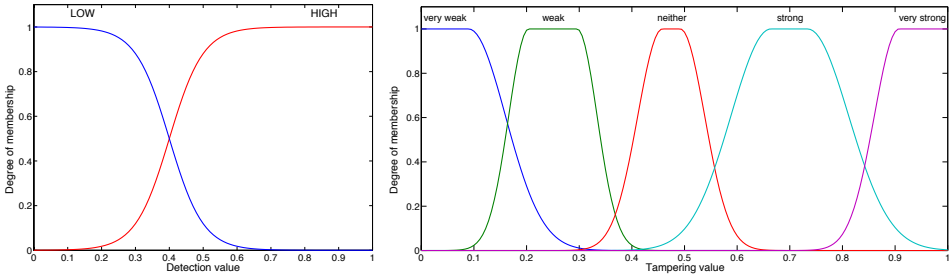


Figure 6.1: *Example of membership functions for D and R (left) and tampering (right).*

6.2.2 Automatic construction of ideal rules

The set of rules corresponding to ideal cases of presence or absence of tampering are now derived. Basically, detection is used to define both the antecedent and the nature of consequent (i.e. tampering, not tampering) of each ideal rule; reliability is used to define the strength of tampering (very weak, very strong, etc.) in the consequent.

(A) Detection as a fuzzy variable

The first step is the definition of the relationship between detection D and the expected presence (Y) or absence (N) of tampering for a single tool. Generally, no tool is either wholly capable or incapable of detecting a certain tampering. More often, a tool is capable or incapable of detecting a tampering to some degree, depending on the characteristics of the analysed image.

Definition 3. *A forensic tool is capable of revealing a manipulation if its detection is high and incapable if its detection is low:*

$$\begin{aligned} Y &= \textit{detection is } \texttt{high} \\ N &= \textit{detection is } \texttt{low}. \end{aligned} \tag{6.1}$$

In Definition 3, D is a fuzzy variable, `high` and `low` are fuzzy sets.

The extension to K tools is immediate: first, we substitute each element Y or N of a given column of T_{TRUE} or T_{FALSE} with the corresponding expression of Eq. (6.1); then, we put all the elements in AND relationship. Given the values $D_1, .., D_K$ and a column $\mathbf{C}$, the final expression describes to which degree the inputs belong to the tampering case $\mathbf{C}$. For example, in a hypothetical scenario of four tools, the rule antecedent corresponding to $\mathbf{C} = (Y, Y, N, N)$ would be: D_1 `high` $\wedge$ D_2 `high` $\wedge$ D_3 `low` $\wedge$ D_4 `low`.

(B) Reliability as a fuzzy variable

To deal with reliability, we follow a different reasoning: in ideal cases, the forensic analyst is willing to fully trust the joint indication of the tools if they are all reliable. Should some of the tools not be reliable enough, their combined detection can still be trusted but only to a lesser extent.

Definition 4. *Let $\mathcal{F}_1, .., \mathcal{F}_n$ be the fuzzy sets available for the intensity of tampering y; let $\mathcal{T} = \{t_i\}_{i=1...K}$ be a set of forensic tools. Then, the most intense set is assigned to y if* most of the tools *are highly reliable, the less intense otherwise.*

In other words, this reduces to choosing the most intense fuzzy set (`very strong` or `very weak`) or the less intense fuzzy set (`strong` or `weak`) depending on reliabilities.

It goes without saying that it is very unlikely that all the tools are highly reliable at the same time. More often, some of them will be very reliable and other less, depending on several unpredictable factors falling into the category of noise and uncertainty. Therefore, it is of paramount importance to clarify the meaning of the expression *most of the tools* of Definition 4. Even though the final choice is left to the designer of the system, the most immediate and simple solution is majority: if R of more than half of the tools is high, we assign the most intense fuzzy set to the consequent. There could be other ways to achieve the same goal, such as exploiting again the human operator's knowledge to identify a subset of most trustful tools. Adopting the former solution, however, allows to avoid two issues: i) to define trustworthy subsets

that are valid in general;[2] ii) to trust only a subset of tools would go against the idea of fusion of all contributions including those affected by uncertainty. The experimental validation of Chapter 7 demonstrates that even the simple majority voting is sufficient to outperform traditional methods, thus hinting that greater benefits could be brought by finer solutions.

(C) Example

To clarify the above concepts, we extend the example of Sec. 6.1.2 to encompass the construction of ideal rules. Once the procedure is clear, the generalisation to an arbitrary number of tools is straightforward. Consider the sequence $(Y, N) \in T_{TRUE}$ describing the expected behaviour in presence of a cut & paste tampering with aligned JPEG grids. The resulting fuzzy rule is expressed as follows:

$$\begin{aligned} &\text{IF} && \big(\, D_1 \texttt{ high} \wedge D_2 \texttt{ low}\big) \\ &\text{THEN} && \big[\,\text{IF } (R_1 \texttt{ high} \wedge R_2 \texttt{ high}) \text{ THEN tampering is } \texttt{very strong} \\ & && \quad \text{ELSE tampering is } \texttt{strong}\big]. \end{aligned} \tag{6.2}$$

Although correct, the rule of Eq. (6.2) is not expressed in the form presented in Sec. 5.3.3. Consequently, we reduce it to a standard form by starting from the expression inside square brackets, which can be decomposed in two contributions as follows (see [Sivanandam et al., 2007]):

$$\begin{aligned} &\text{IF} && \big(\, D_1 \texttt{ high} \wedge D_2 \texttt{ low}\big) \\ &\text{THEN} && \big[\,\text{IF } (R_1 \texttt{ high} \wedge R_2 \texttt{ high}) \text{ THEN tampering is } \texttt{very strong}\big] \end{aligned} \tag{6.3}$$

$$\begin{aligned} &\text{IF} && \big(\, D_1 \texttt{ high} \wedge D_2 \texttt{ low}\big) \\ &\text{THEN} && \big[\,\text{IF } (\overline{R_1 \texttt{ high} \wedge R_2 \texttt{ high}}) \text{ THEN tampering is } \texttt{strong}\big]. \end{aligned} \tag{6.4}$$

These two new compound rules can be further rewritten [Sivanandam et al., 2007] as:

$$\begin{aligned} &\text{IF} && \big(\, D_1 \texttt{ high} \wedge D_2 \texttt{ low}\big) \wedge \big(R_1 \texttt{ high} \wedge R_2 \texttt{ high}\big) \\ &\text{THEN} && \text{tampering is } \texttt{very strong} \end{aligned} \tag{6.5}$$

$$\begin{aligned} &\text{IF} && \big(\, D_1 \texttt{ high} \wedge D_2 \texttt{ low}\big) \wedge \big(\overline{R_1 \texttt{ high} \wedge R_2 \texttt{ high}}\big) \\ &\text{THEN} && \text{tampering is } \texttt{strong}. \end{aligned} \tag{6.6}$$

[2] This problem becomes even more complex as the number of cooperating tools increases.

Now both rules are in the form used in Eq. (5.5). The first rule tells that, given an image and an ideal case, i.e. (Y, N) in the example, if D_1 and D_2 have a high grade of membership in their respective sets and both the tools are reliable, then the most intense consequent corresponding to tampering can be assigned. The second rule tells that if one of the tools (or both) is not reliable,[3] then a less intense level of tampering is assigned to the consequent.

By repeating the above procedure for the remaining ideal cases, we can automatically compose the corresponding set of fuzzy rules without any intervention from the forensic analyst.

The extension of the reasoning to a general case of K tools is quite easy. The K tools will produce rules characterised by the same compound structure of Eq. (6.2), that can be reduced to a set of ideal rules by means of Eqs. (6.3)–(6.6).

6.2.3 Automatic construction of non-ideal rules

Constructing if-then rules for non-ideal cases belonging to T_{DOUBT} is not much different than the ideal cases. However, when a non-ideal case occurs, no support from theory or experiments is available, hence further reasoning is necessary. Consequently, we first bring the non-ideal case back to something we know (i.e. an ideal case); then, we use a procedure similar to that of ideal cases.

(A) Mapping doubt into known behaviours

The first task is carried out by means of a mapping strategy that takes into account the reliability of the various tools. Let $\widehat{\mathbf{C}}$ be a non-ideal sequence belonging to T_{DOUBT} and $\mathbf{C}$ an ideal sequence belonging either to T_{TRUE} or T_{FALSE}. Consider now the binary sequence created by assigning values 0 and 1 to N and Y respectively. Then, the distance between sequences $\widehat{\mathbf{C}}$ and $\mathbf{C}$ can be computed, e.g. by means of the following weighted Hamming distance:

$$d(\widehat{\mathbf{C}}, \mathbf{C}) = \sum_{i=1}^{K} R_i \cdot \mathrm{XOR}\left[\widehat{\mathbf{C}}(i), \mathbf{C}(i)\right], \tag{6.7}$$

[3] Recall that according to De Morgan identity: $\overline{R_1\,\texttt{high} \wedge R_2\,\texttt{high}} = \overline{R_1\,\texttt{high}} \vee \overline{R_2\,\texttt{high}}$.

where: K is the number of tools; R_i is the reliability of the i-th tool; XOR is the bitwise exclusive-OR; $\widehat{\mathbf{C}}(i)$ and $\mathbf{C}(i)$ are the i-th bits of $\widehat{\mathbf{C}}$ and $\mathbf{C}$. In this way it is possible to give more importance to the most reliable tools.

By resorting to Eq. (6.7), the distance of $\widehat{\mathbf{C}}$ from all the M ideal sequences is computed. Then, the closest ideal sequence is selected as follows:

$$\mathbf{C}_{min} = \underset{n=1,2...M}{\operatorname{argmin}} \Big[d(\widehat{\mathbf{C}}, \mathbf{C}_n) \Big]. \tag{6.8}$$

Note that Eqs. (6.7)–(6.8) do not define fuzzy operations, since the mapping is performed before the inference system is built.

(B) Building if-then rules from mapping

Since the mapping is an approximation based on experimental parameters, it is not wise to lean towards presence or absence of tampering as much as in Eq. (6.2). Hence, we make available for the consequent only the less intense fuzzy set (`strong` or `weak`), regardless of reliability. Conversely, we build the antecedent exactly as in the ideal case.

Once again, we clarify the concepts by means of the example of Sec. 6.1.2. This time we consider the non-ideal sequence (Y, Y), in which both t_1 and t_2 claim that the analysed region has been tampered with. Since these tools are mutually exclusive, the sequence is doubtful. Suppose that the ideal case at distance d_{min} is $\mathbf{C}_{min} = (Y, N)$, that is a case indicating the presence of tampering. The resulting if-then rule is:

$$\begin{aligned} &\text{IF } \big(D_1 \texttt{ high} \wedge D_2 \texttt{ high} \big) \\ &\text{THEN tampering is } \texttt{strong}. \end{aligned} \tag{6.9}$$

Note that in Eq. (6.9) we did not take into account again the role of the reliability because it was already exploited for mapping.

Although we rarely observed such a behaviour in our experiments, it is possible that two or more tools are equally reliable, thus generating more than just one $\mathbf{C}$ at distance d_{min} from the non-ideal case $\widehat{\mathbf{C}}$. In this situation, the system acts as follows: i) if all sequences $\mathbf{C}$ at distance d_{min} belong to T_{TRUE} or T_{FALSE}, then the first of the set is chosen and the task is carried on normally as described in Eq. (6.9); ii) otherwise, the system resorts to the fuzzy set `neither weak nor strong` for the consequent.

6.2.4 Schematisation of the complete framework

Now that we described all the blocks composing it, we can schematise the proposed fuzzy fusion framework. Before fusing, we build the set of ideal and non-ideal rules as in Secs. 6.2.2–6.2.3. Then, we fuzzify the input pairs (D, R) coming from the forensic tools by means of membership functions of Sec. 6.2.1 and we compute the strength of each single rule by resolving logical operators. We aggregate all the rules to produce the output membership function, which is finally defuzzified. Finally, we compare the resulting crisp number measuring the intensity of tampering with a threshold. Values above such thresholds indicate that the image is not authentic. The whole procedure is schematised in Fig. 6.2 and exemplified in the next section.

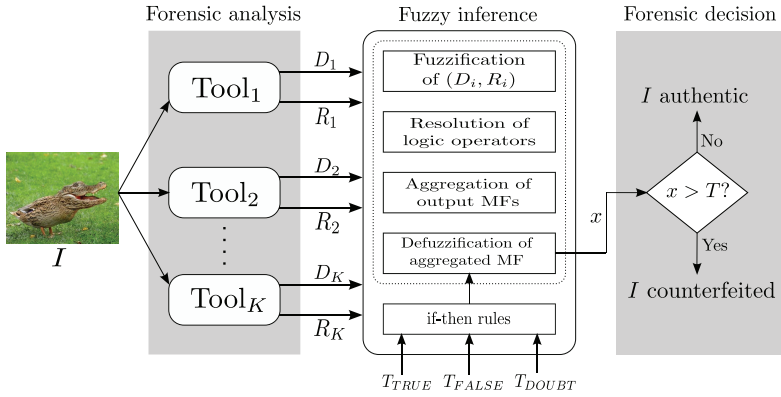


Figure 6.2: *Schematisation of the fuzzy fusion framework.*

6.3 The full toy example

In this section we complete the example that we have been using throughout the chapter by showing the inference process (fuzzification, resolution of logical operators, reasoning, defuzzification) in numbers and how, depending on its outcome, a decision can be made on the authenticity of the image under analysis.

Recall that the two tools $\mathcal{T} = \{t_1, t_2\}$ are mutually exclusive in their search of forgery. They are based on the principle that artefacts of aligned (misaligned) double JPEG compression within a certain image region denote a forgery. Clearly, if the former feature is present, the latter must be absent and vice-versa. The ideal behaviour of t_1 and t_2 is shown again in Tab. 6.2, while the forgeries corresponding to its columns are summarised in Tab. 6.3.

Tool	**Tampering tables**		
	T_{TRUE}	T_{FALSE}	T_{DOUBT}
t_1	Y N	N	Y
t_2	N Y	N	Y

Table 6.2: *Toy example of ideal behaviour of the tools.*

Case	**Tampering procedure**
(Y, N)	The region has been compressed twice with aligned grids
(N, Y)	The region has been compressed twice with misaligned grids
(N, N)	The region has not been compressed twice
(Y, Y)	Doubtful case not supported from theory

Table 6.3: *Tampering procedure for the toy example.*

By applying the principles of Secs. 6.2.2–6.2.3, we derive the following set of rules:

$$\textbf{(Rule1A)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{low}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{very strong}$$

$$\textbf{(Rule1B)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{low}) \quad \wedge \quad \overline{(R_1\,\texttt{high} \wedge R_2\,\texttt{high})} \quad \rightarrow \quad y\,\texttt{strong}$$

$$\textbf{(Rule2A)}: \quad (D_1\,\texttt{low} \wedge D_2\,\texttt{high}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{very strong}$$

$$\textbf{(Rule2B)}: \quad (D_1\,\texttt{low} \wedge D_2\,\texttt{high}) \quad \wedge \quad \overline{(R_1\,\texttt{high} \wedge R_2\,\texttt{high})} \quad \rightarrow \quad y\,\texttt{strong}$$

$$\textbf{(Rule3A)}: \quad (D_1\,\texttt{low} \wedge D_2\,\texttt{low}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{very weak}$$

$$\textbf{(Rule3B)}: \quad (D_1\,\texttt{low} \wedge D_2\,\texttt{low}) \quad \wedge \quad \overline{(R_1\,\texttt{high} \wedge R_2\,\texttt{high})} \quad \rightarrow \quad y\,\texttt{weak}$$

$$\textbf{(Rule4)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{strong}.$$

Some observations are in order. First of all, we omitted for the sake of clarity the step in which rules labelled with "**B**" are further simplified by resolving the negation. In practice, the system generates three new rules for each of them, where the consequent takes into account the sub-cases of: $(R_1\,\texttt{high} \wedge R_2\,\texttt{low})$; $(R_1\,\texttt{low} \wedge R_2\,\texttt{high})$; and $(R_1\,\texttt{low} \wedge R_2\,\texttt{low})$. Therefore, the complete set consists of 13 rules. Note that **(Rule4)** describing the doubtful case has been mapped here to a case of tampering according to the numerical input values that we will introduce in the sequel.

For the sake of simplicity, we consider only four of the above rules. Such an assumption does not alter the final outcome of the example, since the excluded rules would not bring noticeable contributions to the computed fuzzy quantities.[4] The chosen rules are the following:

$$\textbf{(Rule1A)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{low}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{very strong}$$

$$\textbf{(Rule1B1)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{low}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{low}) \quad \rightarrow \quad y\,\texttt{strong}$$

$$\textbf{(Rule3A)}: \quad (D_1\,\texttt{low} \wedge D_2\,\texttt{low}) \quad \wedge \quad (R_1\,\texttt{high} \wedge R_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{very weak}$$

$$\textbf{(Rule4)}: \quad (D_1\,\texttt{high} \wedge D_2\,\texttt{high}) \quad \rightarrow \quad y\,\texttt{strong}.$$

The rules are graphically represented by means of the corresponding input and output fuzzy variables and fuzzy sets in Fig. 6.3.

[4] Obviously, this assumption does not hold as the numbers of tools and rules increase.

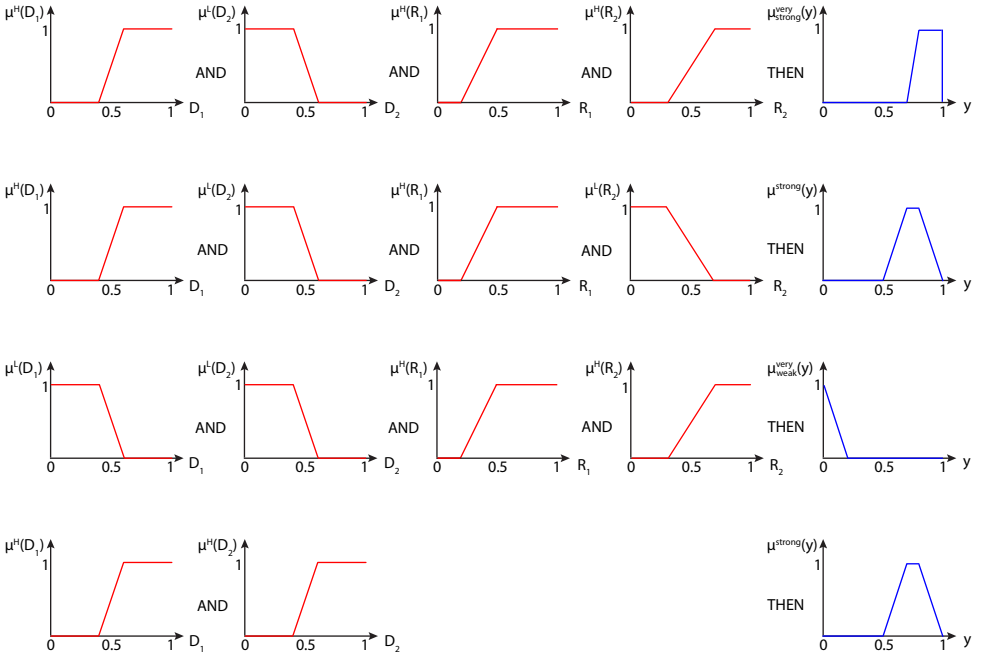


Figure 6.3: *Graphical representation of if-then rules.*

Consider now the following input values:

$$D_1 = 0.9,\ R_1 = 0.8,\ D_2 = 0.2,\ R_2 = 0.6.$$

The four crisp values are turned into fuzzy quantities by means of the membership functions. The procedure is shown in Fig. 6.4.

Following fuzzification, we compute the supports of the rules by resolving logical operators. In this case, we use the **min** function because all the membership functions are in AND relationship. Then:

$$\begin{aligned} \text{support}(\textbf{Rule1A}) &= \min\left[\mu^{\texttt{high}}(D_1), \mu^{\texttt{low}}(D_2), \mu^{\texttt{high}}(R_1), \mu^{\texttt{high}}(R_2)\right] \\ &= \min(1, 1, 1, 0.75) = 0.75 \end{aligned}$$

$$\begin{aligned} \text{support}(\textbf{Rule1B1}) &= \min\left[\mu^{\texttt{high}}(D_1), \mu^{\texttt{low}}(D_2), \mu^{\texttt{high}}(R_1), \mu^{\texttt{low}}(R_2)\right] \\ &= \min(1, 1, 1, 0.25) = 0.25 \end{aligned}$$

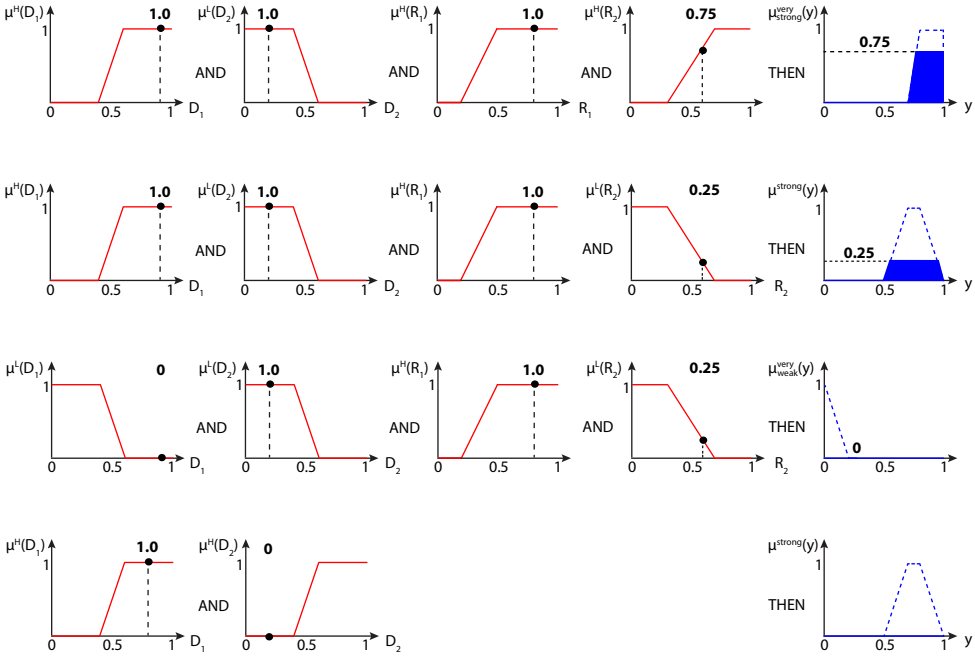


Figure 6.4: *Fuzzification of inputs and resolution of logical operators.*

$$\begin{aligned} \text{support}(\textbf{Rule3A}) &= \min\left[\mu^{\texttt{low}}(D_1), \mu^{\texttt{low}}(D_2), \mu^{\texttt{high}}(R_1), \mu^{\texttt{high}}(R_2)\right] \\ &= \min(0, 0, 0, 0.25) = 0 \end{aligned}$$

$$\text{support}(\textbf{Rule4}) = \min\left[\mu^{\texttt{high}}(D_1), \mu^{\texttt{high}}(D_2)\right] = \min(1, 0) = 0.$$

At this point, the output fuzzy set representing the consequent of each rule is truncated according to its corresponding support (rightmost column of Fig. 6.4). The truncated sets are then aggregated, e.g. by means of max function, as shown by the thick blue line in Fig. 6.5. By doing so, we obtain the final output fuzzy set gathering all the contributions of the rules.

The final task of the inference system is defuzzification: a crisp value x can be obtained from the set of Fig. 6.5, e.g. by computing the centroid of the area under the curve. In this case, $x = 0.8545$. The authenticity of the analysed image is decided based on the comparison with a threshold T (usually 0.5). In this case, then, it turns out that the analysed image is a counterfeit. It is

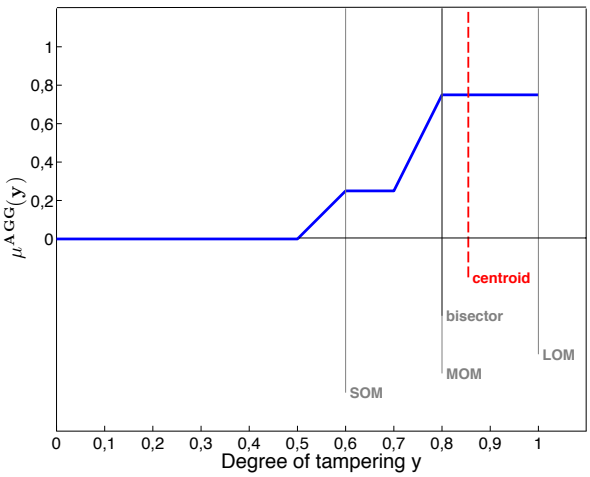


Figure 6.5: *Fuzzification of the proposed example depending on the chosen method: centroid, bisector, Smallest-Medium-Largest Of Maxima.*

worth noting that the decision would not change if we had chosen a different defuzzification method (see lines and names grayed out in Fig. 6.4).

6.4 Concluding remarks

A possible problem with the proposed framework is that the number of rules increases exponentially with the number of variables (that is the dimension of the input space). When *many* tools are employed, this may become a problem that should not be underestimated. As a matter of fact, the exponential explosion of the number of rules is a well known weakness of all fuzzy systems. An excessive number of rules can be the cause of several serious drawbacks, such as: the difficulty of giving a meaningful linguistic description of the scenario; loss of generality, transparency and effectiveness; the increase of computational burden required to control the system. Although the scientific literature has carefully analysed this problem proposing a number of methods to tackle with it (see [Delgado et al., 1997; Zeng and Keane, 2006]), it is not the aim of this work to deal with this situation. When needed, the proposed

framework could be enriched by introducing hierarchical rules clustering or by adopting an approach like the one we used to deal with non-ideal rules to reduce the number of rules composing the system.

The necessity to employ more tools cooperating with each other is a consequence of the typical generation process of a real-world forgery, which is often the product of multiple processing. Several problems may arise when a single decision must be made from outputs that are heterogenous, discording or incomplete. To cope with these problems, in this chapter we formalised a novel framework based on fuzzy logic. The next chapter will be devoted to the experimental validation of the framework in a forensic scenario addressing cut & paste detection.

Chapter 7

Performance evaluation of the fuzzy fusion framework

"He that would perfect his work
must first sharpen his tools"

Confucius

WE SHOW in this chapter how to build a practical implementation of the fusion framework described in the previous chapter. Our ultimate goal is to validate the ideas on which the inference system is based on a realistic image forensic scenario of cut & paste forgery detection, whereby a region from a source image is pasted on a target image with the intent to alter its meaning. More specifically, five recently proposed tools collaborate to decide whether an image has undergone said forgery by analysing a suspicious region within it. The principles of detection underlying the tools are based on JPEG characteristics or, more precisely, on their presence (or absence) within the region under analysis with respect to the rest of the image. Depending on how the forgery is carried out, e.g on the number of compressions steps and grid (mis)alignments, a forensic tool may or may not be able to detect it. This is the typical scenario that we can model with the tables of ideal and non-ideal outputs we discussed in Chapter 6. Moreover, all the typical complications of real-world scenarios are present: forensic tools of varying reliability, different types of images, realistic manipulations in which JPEG footprints are attenuated by processing aiming at creating a convincing fake.

The chapter is organised as follows. Sec. 7.1 describes the forensic tools whose outputs are going to be merged and defines their behaviour in presence of manipulations; additionally, details on the choice of the image data sets used

in the experiments are also provided; for the sake of reproducibility, Sec. 7.2 provides all the parameters of the system; Sec. 7.3, compares the performance of the fuzzy fusion framework to those of a method based on logical OR.

7.1 Tools and image data sets

The presence of JPEG artefacts can be exploited to find out whether an image has undergone a cut & paste forgery. The basic idea is the following. Cut & paste is commonly carried out by taking a region R from a source image S and pasting it into a target image T, thus creating a fake image F. Let either the source or the target images (or both) be JPEG compressed and suppose that the fake image F is saved in JPEG format after the manipulation.[1] The superposition of multiple JPEG compression stages, either with aligned or misaligned 8×8 grids, characterised by different quality factors, typically brings into the fake image a number of peculiar footprints.

By looking for inconsistencies in the number of compressions in certain regions with respect to the rest of the image, a great deal of forensic methods can reveal the occurred manipulation. This work makes use of $K = 5$ of them, namely the methods proposed by Luo *et al.* [Luo et al., 2007], Lin *et al.* [Lin et al., 2009], Farid [Farid, 2009a] and Bianchi *et al.* [Bianchi and Piva, 2011; Bianchi et al., 2011]. For the rest of the chapter they will be referred to as t_A, t_B, t_C, t_D and t_E; we now briefly outline their working principles.

1. Tool t_A determines whether a region has been cropped from a JPEG image with quality QF_1 and pasted without preserving grid alignment on a second image, that afterwards is JPEG compressed with quality factor $QF_2 > QF_1$. Detection relies on a statistical analysis of image blockiness. When used to detect cut & paste tampering, regions where JPEG grids are not aligned are considered as tampered [Luo et al., 2007].

2. Tool t_B determines whether a region has been cropped from a JPEG image or from an uncompressed image and pasted on a JPEG target image. In this case the untouched region undergoes a double compression caus-

[1] Given the widespread usage of JPEG storage, this seems to be a reasonable assumption.

ing its DCT coefficients to be doubly quantised, leaving a characteristic trace called Double Quantisation (DQ) effect [Lin et al., 2009].

3. Tool t_C determines whether a region has been cropped from a JPEG image and pasted while preserving grid alignment. This result is achieved by looking for the so called JPEG ghosts, obtained by differencing the image under analysis and several of its re-compressed versions: forged regions present small differences with respect to those of the rest of the image [Farid, 2009a].

4. Tool t_D detects the presence of non-aligned double JPEG compression by relying on a single feature depending on the integer periodicity of the DCT coefficients. Intuitively, the method evaluates how a subset of the DCT coefficients (the DC coefficients, on which the quantisation effects are more evident) clusters around a given lattice for any possible JPEG grid shift. This measure is compared with a threshold to decide whether grids are aligned or not [Bianchi and Piva, 2011].

5. Tool t_E is a direct improvement of t_B and discriminates between original and forged regions in JPEG images, under the hypothesis that the former are compressed twice while the latter just once. Such a task is performed by relying on two specific probability models for the DCT coefficients of regions that are JPEG compressed once and twice. This method provides better discriminating performance with respect to t_B, especially when $QF_2 < QF_1$ [Bianchi et al., 2011].

For a more in-depth description of the above techniques, we refer to the respective papers. In Sec. 6.1.1, we stated that each tool must produce a detection value $D \in [0, 1]$. For t_A, such value is obtained by applying a probabilistic SVM classifier [Platt, 1999]; for t_B and t_E, it corresponds to the median of the probability map of the analysed region [Lin et al., 2009; Bianchi et al., 2011]; for t_C, it is equal to the KS statistics in [Farid, 2009a]; for t_D, it is a simple normalisation in $[0, 1]$ of the un-thresholded statistic [Bianchi and Piva, 2011].

7.1.1 Ideal behaviour of the forensic tools

The next step consists in the construction of the tables T_{TRUE} and T_{FALSE}. According to the principles underlying the employed tools and according to our preliminary experimental analysis, we identified four classes of tampered images, for which the tools ideally provide different quintuples of answers. By relying on such an analysis, we obtained Tab. 7.1. For the sake of brevity, quintuples belonging to T_{DOUBT} have been omitted but can be easily derived. For a detailed description of the four classes of tampering appearing in the table, see next section and in particular Tab. 7.2. Note that the two pairs

	T_{TRUE}				T_{FALSE}
Tool	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_4$	$\mathcal{C}_5$
t_A	Y	Y	N	N	N
t_B	N	Y	N	Y	N
t_C	N	Y	Y	Y	N
t_D	Y	Y	N	N	N
t_E	N	Y	N	Y	N

Table 7.1: *Expected behaviour of the* 5 *tools. Omitted combinations belong to* T_{DOUBT}.

of tools (t_A,t_D) and (t_B,t_E) work under the same operative conditions (in particular, t_E has been developed to overcome some limitations of t_B). For this reason, their answers should always be in agreement with each other.

7.1.2 Construction of image data sets

We have collected three data sets consisting of original and tampered images:

1. a large data set of $1,600$ images for the general evaluation of the method;
2. a smaller data set of 400 textured natural images to better highlight the benefits brought by the fuzzy approach in a particular case where one of the tools shows an erroneous behaviour;
3. a data set of 60 images simulating real-world tampering.

The first two synthetic data sets share the same generation procedure. Starting from a set of images, we defined four classes of images simulating cut & paste tampering. We designed each class so that at least one tool (or a pair of tools) is able to detect the presence of the manipulation.[2] Each tampering is the result of small variations of a typical cut & paste procedure, as summarised in Tab. 7.2. More specifically, all manipulations have been conducted by substituting the 256×256 central block of the image. In order to bypass most of the technical limitations of the algorithms, we applied the following criteria: $QF_1 \in \{55, 60, 65, 70\}$ and $QF_2 - QF_1 = 20$. We conducted our tests on the 256×256 central area of each image.

Class	Tampering procedure
$\mathcal{C}_1$	Outer region is compressed once. Inner region is compressed twice with misaligned grids
$\mathcal{C}_2$	Outer region is compressed twice with aligned grids. Inner region is compressed twice with misaligned grids
$\mathcal{C}_3$	Outer region is compressed once. Inner region is compressed twice with aligned grids
$\mathcal{C}_4$	Outer region is compressed twice with aligned grids. Inner region is compressed once
$\mathcal{C}_5$	Non-tampered images. The image is compressed once with random quality factor: $QF \in \{70, 75, 80, 90\}$

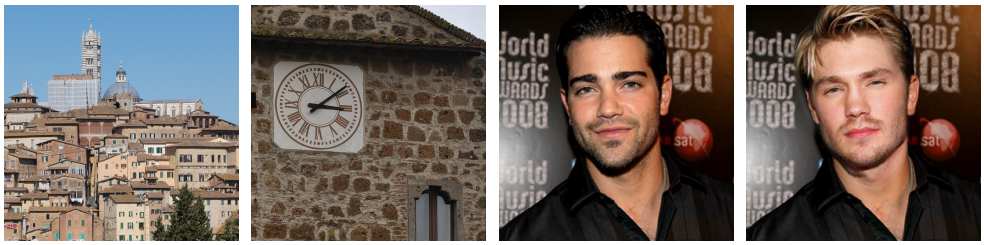
Table 7.2: *Tampering classes. Each class is created by varying the number of compressions with aligned or non-aligned grids.*

The first data set is originated by 100 uncompressed TIFF images with different visual content (landscapes, cityscapes, people). Each original image has been used to create two tampered images, thus leading to 200 images for each class and a total of 800 tampered images. An equal number of non-tampered images, simply compressed once, complete the data set.

The second data set derives from the observation of a peculiar behaviour of t_B. This tool, in fact, tends to claim as tampered a specific type of nat-

[2]For example, the tampering belonging to class $\mathcal{C}_1$ in Tab. 7.2 is a double JPEG compression with misaligned grids, therefore only t_A and t_D are supposed to detect it.

ural images, i.e. those containing textures and regular geometric edges (e.g. buildings, walls, squares), compressed once with a very high quality factor. The other tools do not show any particular behaviour on this specific category of images. The erroneous behaviour of t_B generates doubtful cases that cause an OR-based method to provide wrong results, thus creating a typical scenario in which the fuzzy system should perform better. The plausibility of the experiment is guaranteed by the fact that such images are very common in real-world scenarios. To build this second data set, we used 50 natural images whose central regions contain textures and regular edges, compressed once with native camera quality factor $QF_1 = 100$, to create 200 tampered images and 200 original images with the same procedure of Tab. 7.2. Again, we conducted both tampering and testing on the 256×256 central area. A few examples of such images are depicted in Fig. 7.1.



(a) **(b)** **(c)** **(d)**

Figure 7.1: *Examples of the images used for the experiments. (a)-(b): second data set; (c)-(d): third data set. In particular, (c) is original while (d) is tampered by pasting a new face into (c).*

The third data set originated from a simple consideration: rarely, in real-world scenarios, a tampering is obtained by playing around only with JPEG compression on well defined square regions. Even an unexperienced user will resort to several tools provided by some image editing software, in the attempt to cut & paste regions of irregular shape and variable size. After that, she will likely spend some time correcting inconsistencies in color, size and edges. Finally, most of the times the partial/final result will be stored in JPEG format.

To mimic this situation, we created a set of images of convincing visual quality by using several popular processing tools. As for the subjects of the manipulations, we chose images containing frontal faces. The reason behind this choice is twofold: it is very common to stumble on manipulated faces on the Web due to the meaningful message they convey (e.g. satirical or political); fixed postures allow to create good photomontages rather easily.

Therefore, 30 original images were used to created 30 fakes by substituting the original faces. Tampering was performed by means of Adobe Photoshop. A variety of processing tools have been used including: geometrical manipulations (scaling, rotation, horizontal flip); color manipulations (brightness and contrast correction); enhancement of pasted region's borders. In case of multiple JPEG compressions, all these processing steps tend to attenuate or eliminate the JPEG artefacts of the oldest compression step. To avoid the complete loss of such traces, we paid particular attention to quality factors before and after the processing. Since Photoshop defines JPEG quality with linguistic terms rather than with numerical values, medium quality corresponds to QF_1 and maximum quality to QF_2 (recall that all tools perform better when $QF_1 < QF_2$). The experiments have been conducted on the bounding boxes containing the faces. Figs. 7.1 (c)-(d) provide an example of an original image and its tampered counterpart.

7.2 Experimental settings

The set of parameters of the fuzzy approach is rather small. We opted for the Mamdani's model for if-then rules; we implemented the AND operator with the min function to combine antecedents; the intensity of the consequent is based on the majority criterion applied to the reliabilities of the tools; if-then rules are aggregated by means of the max function; defuzzification is carried out by means of the centroid method. The system features 10 inputs ($D_{A,B,C,D,E}$ and $R_{A,B,C,D,E}$) and one output (*tampering*).

(A) Detection

In order to calculate D, instead of considering only the detection value of each tool in the specified region, we carried out two separate analyses: one on the

region under investigation (D_{inner}) and one on the rest of the image (D_{outer}). Then, we fed their absolute difference $D = |D_{outer} - D_{inner}|$ to the fuzzy fusion system. This approach is, in fact, more robust to false positives. Given a tool, if no tampering has occurred, the difference between the outputs of the detectors on the inner and outer regions should be small (ideally 0). On the other hand, if the region has been tampered with, the difference between the values should be very noticeable (ideally 1). It goes without saying that, due to the uncertainty usually affecting forensic tools, the system often works with differences D that are quite distant from their ideal values.

(B) Reliability

While defining R, it turned out that t_A, t_D and t_E are more reliable when the second JPEG quality factor QF_2 is high. For this reason reliabilities increase linearly depending on QF_2. We derived the coefficients of the linear transformation of Tab. 7.3 from the curves of accuracy as a function of QF_2 published by the authors of the tools [Luo et al., 2007; Bianchi and Piva, 2011; Bianchi et al., 2011].

	Second JPEG quality factor QF_2				
Tool	**60**	**70**	**80**	**90**	**100**
R_A	0.7292	0.825	0.8975	0.9642	0.9655
R_D	0.695	0.748	0.795	0.8267	0.9083
R_E	0.65	0.735	0.82	0.92	1

Table 7.3: *Reliability of t_A, t_D and t_E depending on QF_2.*

Reliabilities of t_B and t_C do not seem to be affected by QF_2, therefore, following previous tests carried out on a separate data set, they are set to the following constant values: $R_B = 0.4$ and $R_C = 0.85$. In Sec. 7.3.3 we evaluate the robustness of the fuzzy system to variations of reliability scores.

(C) Membership functions and if-then rules

In our implementation we opted for smooth membership functions for the variables: sigmoid for inputs and combination of Gaussians for the output.[3] Fig. 7.2 (left) shows that detection (but also reliability) can belong to two fuzzy sets: `low` and `high`. The point where the two functions cross (*crossover point*) is where maximum fuzziness is measured, since an input value is characterised by the same degree of membership for both classes. Values to the left of this point have a higher degree of membership in the fuzzy set `low` ($\mu_{low} > \mu_{high}$); values to the right of this point have a higher grade of membership in the fuzzy set `high` ($\mu_{low} < \mu_{high}$). More details on the choice of points of maximum fuzziness will be provided in Sec. 7.3.1. Fig. 7.2 (right) plots the output

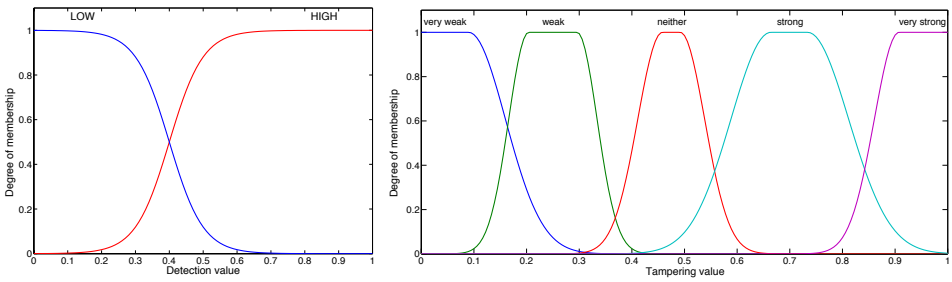


Figure 7.2: *Smooth membership functions for system variables. Left: input detection (in this case on point of max fuzziness* 0.4*). Right: output.*

membership functions representing the intensity of tampering. In this case, five fuzzy sets are defined: from left to right `very weak`, `weak`, `neither weak nor strong`, `strong` and `very strong`.

[3] Piecewise trapezoidal membership functions have been also employed with slightly lower performances; the corresponding curves are omitted for the sake of brevity.

7.3 Experimental validation

7.3.1 Evaluation procedure

The experiments started with a separate analysis of the forensic tools. To this aim, a dedicated data set of $1,600$ images has been created according to the procedures of Tab. 7.2 but starting from a different set of original images. The reason behind using a new data set rather than the one described in Sec. 7.1.2 is the following. At this point, we are still evaluating separately the performance of each tool: this stage is meant to provide the optimal setup for the aggregation of tools. If such parameters were tuned on a data set then used again to evaluate the overall performance, incorrect results may be produced.[4] The Receiver Operating Characteristic (ROC) of each tool was computed only on those subsets of the data set that satisfy the assumptions the tool relies on: t_A and t_D on classes $1, 2, 5$; t_B and t_E on classes $2, 4, 5$; t_C on classes $2, 3, 4, 5$.

With detailed information about the performance of each tool at disposal, we computed the ROC curves obtainable by using the fuzzy and OR-based fusion as follows.

1. **Sampling the ROC of each tool.** Firstly, the probability of false alarm P_{fa} of each curve is sampled with step 10^{-3}. Then, for each value of P_{fa}, its corresponding probability of correct detection P_d is retrieved. The pair (P_{fa}, P_d) is used to calculate the threshold in $[0, 1]$ that needs to be applied to the detection values provided by the various tools in order to obtain those probabilities. By repeating this procedure for all curves, five thresholds for each value of P_{fa} are obtained and organised in quintuples $\mathbf{thr} = [thr_A, thr_B, thr_C, thr_D, thr_E]$.

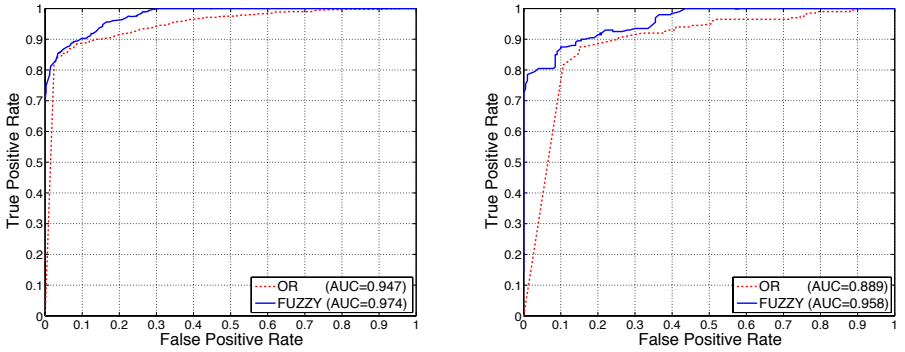
2. **Aggregated ROC of the OR method.** The values of each **thr** are directly used as binary thresholds. For example, thr_A is used to threshold D_A (the output of tool t_A), thr_B to threshold D_B and so on. The authenticity of the analysed region is verified by OR-ing the five binary values.

[4] From a Machine Learning perspective, this corresponds to performing both training and testing stages on the same data.

3. **Aggregated ROC of the fuzzy method.** The values of **thr** are used to set the point of maximum fuzziness for `high` and `low` membership functions of D. For example, thr_A is used for the membership functions of D_A, thr_B for those of D_B and so on. Once membership functions are defined, the process of fuzzification, reasoning and defuzzification is carried out. In order to make a final decision on image authenticity, the defuzzified value of tampering presence is compared with the binary threshold set to 0.5. Note that only membership functions of detection are set according to **thr**, while those of reliability have a fixed crossover point set to 0.5.

7.3.2 Results and discussion

We measured the accuracies of OR-based and fuzzy-based fusion in terms of Area Under Curve (AUC). Fig. 7.3 (a) shows the results we obtained on the data set of $1,600$ images. The results of the two methods are very close to



(a) Synthetic $1,600$ **(b) Synthetic 400**

Figure 7.3: *ROC curves for the two synthetic data sets.*

each other, with fuzzy approach slightly outperforming logical OR (+2.7% AUC). This can be explained by noting that the image classes have been designed so that at least one tool is ideally able to detect the tampering. No unknown tampering altering the analysed features was introduced. In addition, the number of tools is still quite limited. For these reasons, this is a

case that is likely to be managed satisfactorily even by a simple OR operator. Nevertheless, the fuzzy method already performs better.

In order to better highlight the potentiality of the fuzzy framework, the same test is performed on the second data set. Recall that we created such data set of highly textured natural images to trigger a specific weakness of one of the tools. By doing so, we introduced uncertainty that should highlight the superior performance of our method. Fig. 7.3 (b) confirms that, as expected, the benefits brought by the fuzzy system are now more significant (+6.9% AUC).

Consider now the data set of tampered faces, that is the closest representation of a real-world scenario. As clearly shown in Fig. 7.4, the benefits brought by the fuzzy approach are evident (+11.2% AUC). Moreover, a large portion of such gain is located in the leftmost part of the curve that corresponds to low $P_{fa} \leqslant 0.15$, i.e. the working condition that are likely to be used in practice. In conclusion, the last results are an encouraging step towards the correct understanding of what happens in real-world scenarios, where unknown processing is likely to introduce doubtful cases that the fuzzy approach can handle more efficiently.

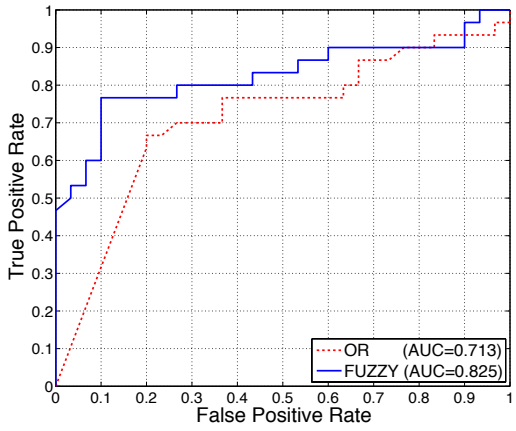
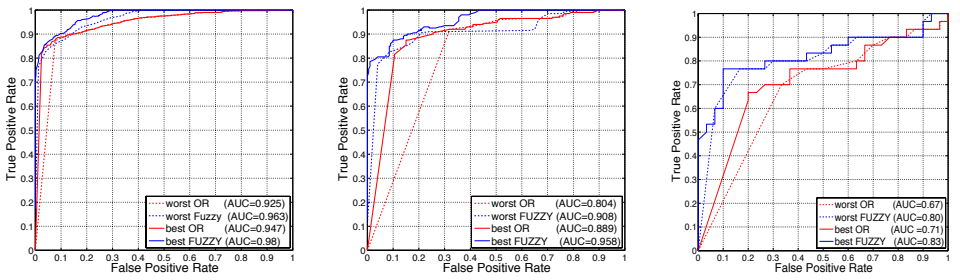


Figure 7.4: *ROC curves for the data set of tampered faces.*

7.3.3 Robustness to variations of parameters

Reliabilities R_A, R_C and R_E have been derived from the papers implementing their respective algorithms, while R_B and R_C were defined experimentally. Although this is the typical example of the operator's knowledge being passed to the fuzzy system, the assignment of constant values may appear as an arbitrary choice depending on experimental data. Therefore, this section evaluates the robustness of the proposed approach with respect to relatively small variations of reliability. To this aim, we iterated the experimental procedure of Sec. 7.3.1 for variations of R_B in $[0.3, 0.5]$ and R_C in $[0.7, 0.9]$ with step 0.05. The results are shown in Fig. 7.5: for each data set we display the ROC curves corresponding to the best (solid lines) and worst (dotted lines) performances of both fusion methods.



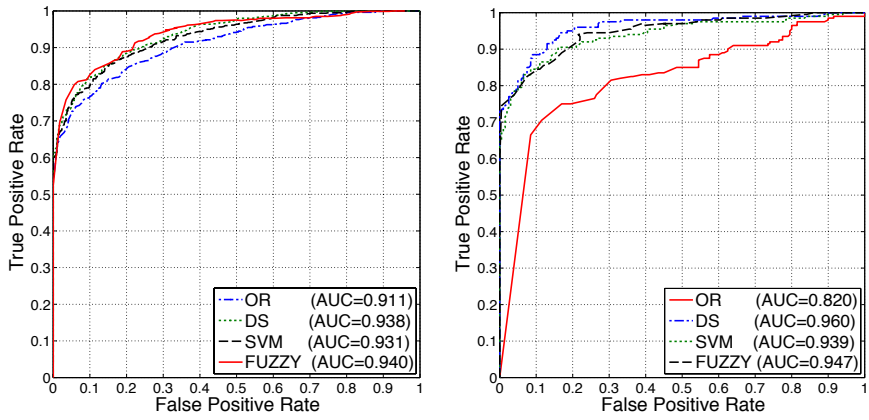
Synthetic $1,600$ **Synthetic** 400 **Realistic** 60

Figure 7.5: *Robustness to variations of R_B and R_C. Solid lines: performance's upper bound; dotted lines: performance's lower bound.*

The small differences in terms of AUC between the two curves on all data sets (1.7%, 5% and 3% respectively) highlight the robustness of the fuzzy approach when we assign suboptimal values to reliability. It is worth noting that the minimum performance granted by the fuzzy fusion continues to be superior to that of the optimised OR method.

7.3.4 Comparison with other fusion techniques

A comparison between different fusion methods, including an earlier version of the system discussed in this chapter, has been presented in [Fontani et al., 2012]. More specifically, Fontani *et al.* compare four methods: Fuzzy [Barni and Costanzo, 2012b], Dempster-Shafer [Fontani et al., 2011],[5] logic OR and SVM. The methods were used to merge the outputs of 3 forensic tools, i.e. t_A, t_B and t_C, on the two synthetic data sets of Sec. 7.1.2. From the ROC curves of Fig. 7.6 (left) we can observe that the performance of all the methods in terms of AUC are very similar on the large data set of $1,600$ images; as we already observed, we designed such data set so that at least one tool can detect the tampering, thus no uncertainty was introduced. Conversely, on the noisy data set of 400 images of Fig. 7.6 (right) the difference between the simple OR and the rest of the methods is evident. Tab. 7.4 summarises the true positive rate of the examined techniques for a selection of low false positive rates.



Synthetic $1,600$ **Synthetic** 400

Figure 7.6: *Comparison of different fusion methods.*

[5] This work has been extended in [Fontani et al., 2013].

	False Positive Rate				
Method	**0**	**0.05**	**0.1**	**0.15**	**0.2**
DS	0.54	0.77	0.84	**0.88**	**0.91**
Fuzzy	**0.6**	**0.8**	0.84	0.85	0.88
SVM	0.51	0.77	0.84	**0.88**	**0.91**
OR	0.56	0.76	0.81	0.84	0.86

Table 7.4: *Accuracy of different fusion methods for low false positive rates.*

7.3.5 Computational complexity

When K forensic tools are employed, there are 2^K possible K-uples belonging either to T_{TRUE}, T_{FALSE} or T_{DOUBT}. Each column of these tables corresponds to an if-then rule in the form of Eq. (6.2) or Eq. (6.9), thus resulting in 2^K compound rules. In practice, it is not possible to deal directly with rules in such a form, hence the resorting to Eqs. (6.3)–(6.6) and to a particular indexing for fuzzy sets, that is necessary to apply the majority criterion to reliabilities. The final number of basic rules amounts to 2^{2K}. In the presented case of $K = 5$ tools the system consists of 32 cases generating a set of $1,024$ if-then rules. On a common desktop configuration (3 GHz dual-core processor, 4 GB RAM, 32 bit OS), the optimised version of our code resulted in the following execution times: about 1 second to build T_{TRUE}, T_{FALSE} and T_{DOUBT} (this operation is performed only once for each data set); 0.2 seconds to build the fuzzy inference system and 0.5 seconds to resolve all rules (these operations are performed once for each image).

7.4 Concluding remarks

In this chapter we implemented and validated a fuzzy inference system to deal with uncertainty when using multiple forensic detectors. Usually each forensic technique deals with a single type of manipulation, whilst a real tampering is often the result of several processing. It is therefore necessary to employ more tools cooperating with each other. Several problems arise when we must take a single decision by looking at heterogenous, discording or incomplete outputs.

The results are promising, nevertheless several issues still need to be explored, including: i) implementing a wider set of forensic tools working on different manipulations; ii) devising a strategy to tackle the exponential growth of if-then rules occurring every time that a new tool is added to the system. For example, we could use the weighted Hamming distance currently employed to map non-ideal cases or a hierarchical clustering; iii) testing the accuracy on a large real-world data sets of tampered images, e.g. gathered from the Web; iv) comparing the fuzzy method against other soft decision approaches, like Bayesian, Support Vector Machines and Neural Networks, on extended data sets; v) extending the theoretical framework to the most complex case where the suspicious tampered region is not known a priori. In this scenario we can imagine that each tool produces a tampering map automatically localising the forged region. As a matter of fact, the process of localisation may introduce uncertainty. Once the region has been located, the fusion can proceed as described in this chapter.

Part III

Digital Image Counter-Forensics

Digital Image Counter-forensics is the art of misleading forensic analysis by hiding, removing or falsifying the traces that forensic algorithms are looking for. Although this discipline is still in its infancy, researchers have successfully bypassed both source and forgery detection by means of methods hiding the presence of compression artefacts, histogram manipulations, resampling or median filtering. Most of the times, however, counter-forensic techniques are not perfect and leave traces on their own. Such traces can then be exploited by the forensic analyst, thus feeding a circle improving the tools at the community's disposal.

Despite the satisfactory results obtained so far by Counter-forensics, there are still some categories of algorithms that have not been challenged, such as those based on salient point detectors. The features extracted by such detectors, in fact, are robust against several processing tools and generally invariant to geometrical manipulations; moreover, their high distinctiveness makes them the ideal choice for matching-based forensic detection (e.g. copy-move). In this part of the thesis we target the most popular of such feature extractors, i.e. the Scale Invariant Feature Transform (SIFT) and we attempt to impair forensic detectors based on it. Our contribution is threefold: i) we propose a new attack to remove SIFT keypoints from images and we apply it successfully to bypass a state-of-the-art SIFT-based copy-move detector; ii) we develop three different algorithms to detect global and local keypoint removal; and iii) we propose an attack to inject fake keypoints and we attempt to mislead the removal detectors introduced previously.

Chapter 8
Introduction to Counter-forensics

"*What were you expecting? Once the process of falsification is set in motion, it won't stop. We are in a country where everything that can be falsified has been falsified* [...]. *The result is that nobody can be sure what is true and what is false, the political police simulate revolutionary actions and the revolutionaries disguise themselves as policemen.*"

If on a winter's night a traveler
Italo Calvino

THE TOPIC of this chapter is Digital Image Counter-forensics. Sec. 8.1 informally introduces the meaning and the role of counter-forensic analysis and explains the reasons why it can be useful to the whole forensic community. Secs. 8.2–8.3 formally approach the problem of Counter-forensics from a theoretical point of view, as proposed in [Kirchner, 2011; Böhme and Kirchner, 2012]. In these works, a model for Image Forensics is first elaborated and then extended to embrace the counter-forensic scenario. Such a framework will be adopted for the rest of the dissertation. The chapter ends with the overview of counter-forensic techniques in Sec. 8.4.

8.1 What is Counter-forensics?

The first decade of Image Forensics has produced a large number of algorithms, some of which has been described in Chapter 2. The robustness of these algorithms is generally evaluated against the most common processing tools, such as compression or resampling. However, little or no importance has been given to the study of countermeasures specifically devised to bypass the analysis of the forensic algorithms. Anyone with such a purpose is commonly referred to as adversary, i.e. a party who has the same knowledge on signal

processing as the forensic analyst and some reasons to mislead a specific image forensic investigation.

A goal like this can be pursued by hiding, removing or falsifying traces of illicit processing, so that the counterfeited image appears authentic. In other words, the adversary does not limit anymore herself to manipulate an image but she also wants that manipulation to be *undetectable*. All the solutions in this sense fall into the discipline called *Counter-forensics* (also known as anti-forensics or tamper hiding). Most of the times, similarly to the algorithms they are supposed to fool, counter-forensic techniques are not perfect and leave traces on their own. Such traces can then be exploited by the forensic analyst to detect the adversary's misdoing and gain a new advantage. Far from being a mere *divertissement*, studying the mutual interaction between these two parties can bring several advantages, such as:

- providing data about real-world scenarios. Forensic techniques are generally validated in a "friendly" laboratory environment. Intelligent counter-forensic methods can instead provide valuable insights on real-world performance;

- building better tools. The weaknesses exploited by counter-forensic methods can either push towards devising improved or brand new, more secure, forensic algorithms;

- providing directions for theoretical formalisations. Research community has already begun to study adversarial challenges to forensic analysis more rigorously. Encasing Counter-forensics into a sound framework would ultimately allow it to evolve from the *art* of misleading to the *science* of misleading;

- providing tools for improved privacy. Sometimes the information that a forensic algorithm can extract from an image may violate the privacy of its rightful owner. Erasing sensitive footprints by means of Counter-forensics could be seen as an effective form of self-defense.

Given the above, the growing interest on Counter-forensics should not be surprising. There are, however, some caveats. The potentially virtuous circle, defined by any new iteration of forensic algorithm, counter-forensic scheme

and new forensic algorithm, risks to lean towards a cat and mouse game, i.e. *a contrived action involving constant pursuit, near captures, and repeated escape*,[1] leading to a sterile stalemate situation. This is a risk connected to the fact that Counter-forensics is still in its infancy, often looking more like the work of "craftsman" rather than that of a scientist. To address this potential issue, the counter-forensic scenario has been recently cast into more rigorous theoretical frameworks, including the one introduced in the next section.

8.2 Formalisation of the forensic problem

The next two sections review the formalisations of Image Forensics and Counter-forensics as a hypothesis testing problem, as recently proposed by Kirchner and Böhme [Böhme and Kirchner, 2012; Kirchner, 2011]. Such framework will be adopted throughout the rest of the thesis.[2] The authors first systematise the procedure of generating a digital image and then develop a terminology for original, authentic and counterfeited images. Furthermore, the forensic analysis is modeled as a decision problem and Counter-forensics is contextualised accordingly. In the following, only those concepts that are instrumental to the purpose of the dissertation are summarised. The interested reader may consult [Kirchner, 2011] for a full account of the complete theoretical framework.

8.2.1 Image generation process

A digital image is the representation over a finite set of discrete symbols of one of all the possible natural phenomena $\mathcal{N} \in \mathscr{N}$. An image I is generated by means of a function, namely $\texttt{generate} : \mathscr{N} \times \Theta \to \mathscr{I}$, where Θ is the set of all possible generation parameters and $\mathscr{I}$ is the set of all possible images. More precisely, the generation process is influenced by a subset of parameters $\theta \in \Theta$, depending both on the acquisition device and on the characteristics of the natural scene.

[1] The Mirrian-Webster Online Dictionary, http://www.merriam-webster.com/dictionary/cat%20and%20mouse

[2] It is worth noting that Kirchner and Böhme's is one of the possible problem formalisations. For alternative formulations, see Sec. 8.4.6.

The function `generate` can be further decomposed into two steps: a first stage in which the device captures the natural scene and a second stage in which the acquired scene is conveniently elaborated to produce a suitable output image. Fig. 8.1[3] outlines the generation process. The scene is first captured by a function $\texttt{acquire} \in \mathscr{A} : \mathscr{N} \to \mathscr{I}$, chosen from the set of all possible acquisition functions $\mathscr{A}$. Then, a function $\texttt{process} \in \mathscr{P} : \mathscr{I} \to \mathscr{I}$ taken in the set of all processing functions $\mathscr{P}$ is applied. The pairs $(\texttt{acquire}, \texttt{process})$ belong to the space $\mathscr{A} \times \mathscr{P}$ of all the possible combinations of acquisition and processing functions.

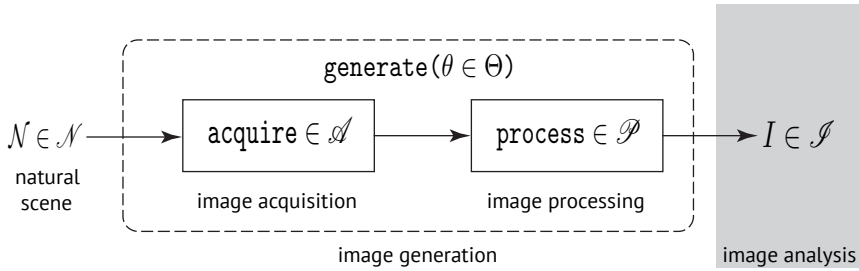


Figure 8.1: *Outline of the image generation process.*

There are some aspects of the above scheme that are worth noting. First of all, `process` can also be a cascade of more than one independent function, i.e. $\texttt{process} = [\texttt{p}_1, \texttt{p}_2, \ldots, \texttt{p}_K]$, $\texttt{p}_k \in \mathscr{P}$, $k \in [1, K]$. Post-processing can be carried out directly by the acquisition device, by software working on device's output or by both. Furthermore, not necessarily post-processing has to follow the acquisition stage. In this case, `process` corresponds to the identity function $\perp_{\mathscr{P}} : I \to I$.

8.2.2 Original images, authentic images and forgeries

With the support of the definitions given in previous section, it is possible to introduce the notions of originality and authenticity of a digital image in terms of the pair $(\texttt{acquire}, \texttt{process})$.

Definition 1. *An image I is original if it is the result of the generation functions* $(\texttt{acquire}, \perp_{\mathscr{P}})$.

[3] Fig. 8.1 reproduced from [Kirchner, 2011] with kind permission of the authors.

The formal definition of originality given by Kirchner and Böhme is very strict: to retain it, an image cannot undergo any kind of processing, not even those embedded more and more often in commercial acquisition devices. Any image in contrast with Definition 1 is considered *processed*.

Definition 2. *An image is processed if it is the result of the generation functions* (`acquire,process`) $\in \mathscr{A} \times \mathscr{P} \backslash \{\perp_{\mathscr{P}}\}$.

From the point of view of a forensic analyst, not all processing is the same. While some operations may be acceptable because they are simply meant to enhance an image or represent it in a convenient way, others may not be allowed as one could exploit them to intentionally alter image semantics. Therefore, it becomes necessary to introduce the notion of *authenticity*. Let $\mathscr{P}_{\text{legitimate}} \subseteq \mathscr{P}$ be the subset of acceptable processing functions. Then, an image is defined *authentic* as follows.

Definition 3. *An image is authentic if it is the result of the generation functions* (`acquire,process`) $\in \mathscr{A} \times \mathscr{P}_{legitimate}$.

Now let $\mathscr{P}_{\text{illegitimate}} = \mathscr{P} - \mathscr{P}_{\text{legitimate}}$ be the subset of processing functions compromising the authenticity of an image. Then, an image is defined *manipulated* as follows:

Definition 4. *An image is manipulated (i.e not authentic) if it is the result of the generation functions* (`acquire,process`) $\in \mathscr{A} \times \mathscr{P}_{illegitimate}$.

In practice, the distinction between legitimate and illegitimate processing, and consequently between authentic and not authentic images, is fuzzier than theoretically, as it is not known beforehand but rather varies significantly from application to application.

8.2.3 Image Forensics as a classification problem

Normally, the function `generate` leaves into the image specific traces, that can be used by the forensic analyst to deduce information on the generation process. By doing so, it is possible to discriminate between different generation processes and also to draw conclusions on the authenticity of an image.

As a consequence, forensic analysis can be modeled as a classification problem, whereby the generation space $\mathscr{A} \times \mathscr{P}$ is partitioned into subspaces $(\mathscr{A} \times \mathscr{P})_{(\mathcal{C}_i)}$, according to different classes $\mathcal{C}_i \in \mathscr{C}$. Each class is determined by a certain range of the generation parameters $\Theta^{(\mathcal{C}_i)}$. All the images of a class share the same identifying traces, i.e. $\mathbf{I}^{(\mathcal{C}_i)} = \left\{ I \in \mathscr{I} \mid I = \texttt{generate}(\theta \in \Theta^{(\mathcal{C}_i)}) \right\}$. Hence, any forensic algorithm can be defined as follows.

Definition 5. *A digital image forensic algorithm is a function* `decide` : $\mathscr{I} \rightarrow \mathscr{C}$ *assigning an image* $I \in \mathscr{I}$ *to a class* $\mathcal{C} \in \mathscr{C}$.

The number of partitions (and thus of classes) depends on the application at hand. Consider the following two examples.

- The source identification problem requires to define a class $\mathcal{C}_k$ for each acquisition device. Consequently, the generation subspaces are determined by the family of acquisition functions $\mathscr{A}_k$ of each device, and correspond to $(\mathscr{A}_k \times \mathscr{P})_{(\mathcal{C}_k)}$.
- The forgery detection problem requires (see Definition 4) to define two classes, i.e. $\mathcal{C}_0$ for authentic images and $\mathcal{C}_1$ for manipulated images. According to which processing tools are considered legitimate and which illegitimate, the corresponding generation subspaces are $(\mathscr{A} \times \mathscr{P}_{legitimate})_{(\mathcal{C}_0)}$ and $(\mathscr{A} \times \mathscr{P}_{illegitimate})_{(\mathcal{C}_1)}$.

8.3 Formalisation of the counter-forensic problem

In Counter-forensics, the goal of the adversary is to bypass the analysis of one or more forensic tools. Any solution serving such purpose is called *attack* or counter-forensic scheme.

Definition 6. *Given an image* I*, a function* `attack` $\in \mathscr{P}$ *is a composition of processing functions that is used to produce a manipulated image* $J = \texttt{attack}(I)$ *capable of misleading a forensic algorithm* `decide`.

From a classification perspective, the purpose of a counter-forensic attack could be twofold: i) to remove or hide specific image properties qualifying the image as a member of a certain class $\mathcal{C}_k$; and ii) to make an image of

class $\mathcal{C}_k$ appear as if it were belonging to another class $\mathcal{C}_l \neq \mathcal{C}_k$. Whenever `attack` manages to alter an image $I_{(\mathcal{C}_k)}$ (shortly $I_{(k)}$) in such a way that `decide` erroneously assigns it to $\mathcal{C}_l$, the resulting image will be indicated with $J_{(\mathcal{C}_{\hat{l}})}$ (shortly $J_{(\hat{l})}$).

The attack of Definition 5 produces a manipulated image. The capability of this image to fool a certain forensic algorithm depends on the notion of *vulnerability* of the forensic technique.

Definition 7. *A digital image forensic algorithm* `decide` *is vulnerable to a counter-forensic scheme* `attack` *if, given an authentic image* I*,*

$$\exists\, \texttt{attack} \in \mathscr{P},\ J = \texttt{attack}(I) : \texttt{decide}(I) \neq \texttt{decide}(J)$$

subject to constraints:

(a) *the probability of finding* $\texttt{attack}(I)$ *is not negligible within a given complexity bound*

(b) *images* I *and* J *are semantically equivalent.*

The first constraint of Definition 8 ensures that the attack is feasible in terms of time and computational efforts. Intuitively, the second constraint ensures that the semantic content of the manipulated image cannot be distinguished from the authentic one. The notion of semantic equivalence, formally explained in Definition 8, is based on the assumption that the correspondence between a natural scene and an image depicting it can be somehow measured.

Definition 8. *Two images* I *and* J *are semantically equivalent if:*

$$\exists\, \mathcal{N} \in \mathscr{N} : |\mathrm{dist}(I, \mathcal{N}) - \mathrm{dist}(J, \mathcal{N})| < d$$

where $\mathrm{dist} : \mathscr{I} \times \mathscr{N} \to \mathbb{R}_+$ *is an arbitrary metric of the semantic distance between an image and a natural phenomenon and* d *is a given threshold.*

Now, if one considers again the two typical forensic classification problems of Sec. 8.2.3, the corresponding counter-forensic strategies are the following.

- Countering source identification. The authentic image $I_{(k)} \in \mathcal{C}_k$ is manipulated to produce the attacked image $J_{(\hat{l})} = \texttt{attack}(I_{(k)})$ such that $\texttt{decide}(J_{(\hat{l})}) = \mathcal{C}_l$, where $\mathcal{C}_l \neq \mathcal{C}_k$. Depending on the objective of the adversary, class $\mathcal{C}_l$ can be any class or a specifically targeted class.

- Countering forgery detection. The authentic image $I_{(0)} \in \mathcal{C}_0$ is firstly altered according to the adversary's objective, thus producing the manipulated image $I'_{(1)} \in \mathcal{C}_1$. Then, a semantically equivalent image $I'_{(\hat{0})} = \texttt{attack}(I'_{(1)})$ is elaborated in such a way that $\texttt{decide}(I'_{(\hat{0})}) = \mathcal{C}_0$.

In [Böhme and Kirchner, 2012], a categorisation of counter-forensic techniques is also proposed. The methods are subdivided according to: whether they address the security or the robustness of a forensic algorithm; the stage of the image generation process at which they are carried out; the range of countered forensic algorithms. Since the theoretical definition of robustness and security is a very complex task which is not the focus of this dissertation, only the second and third categorisations are adopted in the sequel.

The first category subdivides counter-forensic attacks into *integrated* and *post-processing*. An attack is integrated (Fig. 8.2, bottom blocks[4]) if it replaces or interacts with the generation process, by defining a new pair $(\texttt{acquire}', \texttt{process}')$ such that no peculiar footprints are generated or such that traces of a target class are falsified. In other words, integrated attacks prevent the formation of footprints, rather than deleting them at a later stage, which is exactly the behaviour of post-processing attacks (Fig. 8.2, top blocks).

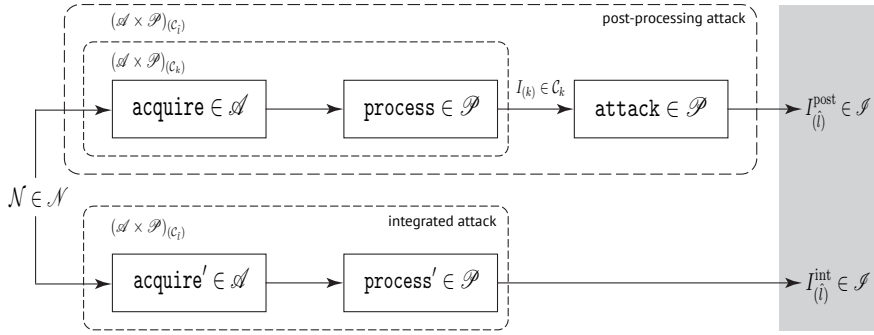


Figure 8.2: *Distinction between post-processing and integrated attacks.*

The second categorisation takes into account the ultimate target of the adversary. If the countermeasure has been specifically tailored to remove traces detectable by a particular forensic technique known to the adversary, then such

[4]Fig. 8.2 reproduced from [Kirchner, 2011] with kind permission of the authors.

an attack is defined *targeted*. On the contrary, if the attack attempts to leave unaltered as many authentic features as possible in order to hide forgeries even to unknown forensic methods, then it is defined *universal*.

So far, the majority of counter-forensic attacks are targeted, as universal attacks are harder to design (for example, see [Barni et al., 2012]). Despite one's intuition, the combination of multiple targeted attacks does not necessarily results in a good universal attack. The problem with such solution, in fact, is twofold: firstly, as it was shown in Chapter 6 from the perspective of the forensic analyst, the interactions between heterogenous tools are not trivial to model; secondly, the accumulation of instruments has detrimental effects on the forged image's quality.

8.4 A brief state-of-the-art of Counter-forensics

This section organises the counter-forensic algorithms proposed so far according to the scenario they address. It is not our aim to provide a thorough analysis of the state-of-the-art on Counter-forensics, but rather to provide some examples of the algorithms that have been devised so far. Therefore, we only briefly describe each technique; we refer to the corresponding paper for a detailed description of the algorithm.

8.4.1 Hiding JPEG or the paradigm of *cat* & *mouse* game

Compressing an image with JPEG generates two peculiar footprints: the comb-like shape of the Discrete Cosine Transform (DCT) coefficient histogram, which contains empty bins caused be the quantisation stage; and the blocking artefacts in the spatial domain caused by the block-based coding approach. Forensic analysis leverages on such artefacts to infer information on the history or on the source of JPEG images. In the past few years, several counter-forensic schemes have been developed to hide traces of compression. Some of these attacks inspired new forensic detectors, which in turn inspired new attacks. Fig. 8.3 graphically schematises what is probably the best example of *cat* & *mouse* game in Image Forensics.

The first attempts to render JPEG footprints forensically undetectable are the techniques in [Stamm et al., 2010a] and [Stamm et al., 2010b]. In [Stamm

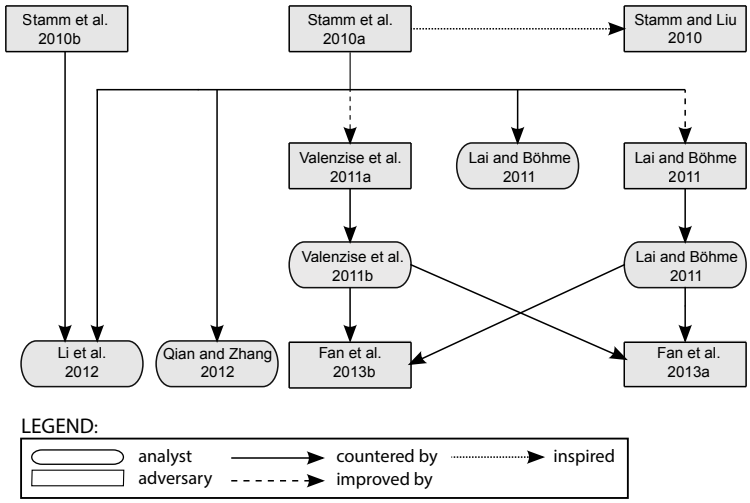


Figure 8.3: *Succession of JPEG forensics and Counter-forensics.*

et al., 2010a], the quantisation gaps in the DCT histogram are reduced by spreading the coefficients with an additive noise called *anti-forensic dither*. The distribution of this noise depends on the value of DCT coefficient that is being modified, on the quantisation step and on the estimation of DCT coefficient distribution prior to compression. As a result, the DCT coefficient distribution in the attacked image approximates as closely as possible the distribution of the unquantised coefficient. In [Stamm et al., 2010b], the traces of blocking artefacts are hidden by means of smoothing based on median filtering, followed by the addition of a Gaussian noise whose variance depends on the JPEG quality factor of the manipulated image. The effectiveness of the two attacks, which can be combined to impair different kinds of detector, is demonstrated against the detector of doubly compressed images in [Fan and de Queiroz, 2003].

Not long after Stamm *et al.*'s studies, [Lai and Böhme, 2011] discovered that anti-forensic dither leaves peculiar traces, and thus a new forensic detector was proposed. Furthermore, the authors modified the original dither in such a way to also fool their own new detector. Finally, a second detector revealing the traces left by the improved dither was developed. When [Stamm

et al., 2010a] cannot estimate the distribution of a coefficient, in fact, this is left unaltered. Such a behaviour generates the two footprints exploited by the first detector, that is the high number of zero valued high frequency coefficients and the small absolute value of unaltered AC coefficients. The above detector is fooled by improving the estimation stage of the anti-forensic dither on the problematic coefficients. The final improved detector, by borrowing from steganalysis the notion of calibration [Fridrich et al., 2003b], uses as cue the ratio of the variance of high frequency coefficients between the manipulated image and its slightly cropped (calibrated) version. Dithered images, in fact, are characterised by a higher ration than authentic images.

The cost of anti-forensic dithering in terms of image quality has been evaluated in [Valenzise et al., 2011b], where it emerged that the impact of the attack is perceptually not negligible. Tests conducted on an improved, content-aware version of the original scheme yield similar results. Based on this preliminary analysis, the same authors presented a new detector capable of discriminating between original images and anti-forensically dithered images [Valenzise et al., 2011a, 2013]. The detector relies on total variation (TV), i.e. the ℓ_1 norm of first-order spatial derivatives, to compute a measure of re-quantisation "noise", that is significantly higher in dithered images. Valenzise *et al.*'s detector, however, is not robust to the counter-forensic scheme in [Fan et al., 2013b], which removes JPEG blocking artefacts by means of a minimisation criterion based on TV. Moreover, Fan *et al.*'s method also includes a de-calibration stage that impairs the detector in [Lai and Böhme, 2011].

In [Li et al., 2012], the authors observe that random DCT modifications as in [Stamm et al., 2010a] destroy the correlations between the coefficients within each 8×8 block and between each block and the adjacent ones. Therefore, again by borrowing from steganalysis [Chen and Shi, 2008], they evaluate such correlations to expose anti-forensic dither. The authors also claim that their method outperforms the one in [Valenzise et al., 2011a]. In [Qian and Zhang, 2012], the abnormal distribution of decimal values of the DCT coefficient of a dithered image with respect to an unprocessed one is used as a proof of manipulation. [Fan et al., 2013a] argue that anti-forensic dither's assumption of Laplacian distribution for the unquantised DCT coefficient is not always accurate. Therefore, they propose a non-parametric DCT histogram

smoothing based on image restoration and calibration. Given a JPEG image, first they estimate its uncompressed version by means of image restoration. Then, they estimate the noise by subtracting the restored image and a calibrated version in DCT domain. Finally, the so obtained noise is added to the starting JPEG image to remove its DCT histogram gaps. The proposed technique is then used to neutralise the detectors in [Lai and Böhme, 2011] and [Valenzise et al., 2011a].

To conclude, it is worth noting that the idea behind anti-forensic dithering has also been extended to encompass the case of Wavelet-based compression [Stamm and Liu, 2010].

8.4.2 Countering other JPEG compression-related methods

Understanding whether an image has been JPEG compressed multiple times can be useful for a number of forensic applications, including the detection of image splicing. In this scenario, a typical detector looks for a mismatch between the number of compressions of manipulated regions with respect to the rest of the image (see the tools of Chapter 7). Expectedly, counter-forensic schemes attempt to hide the manipulation by uniforming the number of compressions, as in [Chunhui et al., 2012], where double quantisation artefacts are removed. The idea underlying the scheme in [Milani et al., 2013] is similar. This tool attacks detectors based on Benford's law of first digit (FD), like the one in [Milani et al., 2012], by altering the FD's probability mass function to match that of a single compressed image.

8.4.3 Hiding traces of image histogram manipulations

Contrast enhancement can be used to destroy forensically relevant traces. There exist several forensic algorithms exploiting the fact that the grayscale histogram of a contrast enhanced image typically exhibits two characteristics: impulsive peaks when multiple pixels are mapped to the same bin and gaps where no pixel is assigned to a bin. The method in [Cao et al., 2010a] allows to remove these peaks and gaps with an approach that is quite similar to the JPEG dither. During the contrast mapping stage, in fact, each pixel is perturbed with a Gaussian dithering whose variance depends on the original pixel value. The attack, implemented in two equivalent versions (integrated into the

contrast remapping and post-processing), successfully impairs the well known detector in [Stamm and Liu, 2008].

Recently, [Barni et al., 2012] tackled with the same problem from a more general point of view and proposed an universal post-processing attack targeting forensic algorithms based on first order statistics (i.e. image histogram). The authors claim that their technique can hide any kind of manipulation leaving traces in the image histogram. The idea behind the technique consists in modifying the histogram of a manipulated image in such a way to match that of another, totally different, authentic image drawn from a database. First, a new target histogram is chosen according to minimum distance and maximum shape similarity criteria. Then, the pixel values of the manipulated image are changed (i.e. moved from a bin to another) according to a mapping function minimising perceptual distortion. The effectiveness of the method is proved by invalidating the contrast enhancement detector in [Stamm and Liu, 2008]. Not long after the above counter-forensic schemes were introduced, a new contrast enhancement detector capable of discovering the traces they leave when applied to color images was devised [Lin et al., 2013]. In fact, when the two attacks are applied to color images, the natural inter-channel similarities of high-frequency components are altered.

8.4.4 Hiding resampling and median filtering

Evidence of resampling is particularly useful for forensic algorithms detecting cut & paste forgeries, since pasting a portion of image into another image typically requires adaptations (e.g. resizing or rotating) to create a convincing forgery. State-of-the-art detectors exploit the space-periodic dependency between neighbouring resampled pixels [Popescu and Farid, 2005]. Such periodicity can be avoided with the integrated attack in [Kirchner and Böhme, 2007; Kirchner and Bohme, 2008]. The image to be resampled is split into two contributions, low-frequency and high-frequency. The high frequency pixels are perturbed with a perceptually-driven Gaussian noise while being resampled, as opposed to the low frequency pixels, that are first traditionally resampled and then median filtered. The two contributions are finally recomposed to produce the resampled and forensically undetectable image.

Median filtering can be used for high quality image denoising and smoothing, as well as for hiding malicious image processing. Among these manipulations, there are the aforementioned removal of evidences of resampling and the anti-forensic dither of Sec. 8.4.1. Consequently, great effort has been put towards its detection, based on the fact that median filtering greatly increases the probability of two adjacent pixels having the same value and introduces correlation between neighbouring image blocks. The method in [Fontani and Barni, 2012] attempts to make such traces forensically undetectable by searching the best sliding window operator which removes median filtering footprints while maximising the similarity between the median image and its countered version. The validity of the approach is experimentally proved by successfully hindering two well known median filtering detectors [Cao et al., 2010b; Yuan, 2011].

8.4.5 Forging image source

As mentioned in Chapter 2, the source of a digital image can be accurately determined by analysing its PRNU or its CFA pattern, which vary between different camera brands and models. Moreover, cut & paste forgeries can be detected by looking at image regions whose underlying PRNU (or CFA) does not match that of the rest of the image. Methods to falsify PRNU and CFA footprints have been proposed respectively in [Gloe et al., 2007b] and [Kirchner and Böhme, 2009]. In [Gloe et al., 2007b] the authentic PRNU is removed and substituted with a target PRNU that has been previously estimated from a set of images originated by the target camera. It is possible to counter this attack by means of the so called *Triangle Test* in [Goljan et al., 2010]. However, recently [Rao et al., 2013] has shown how to successfully impair source identification methods and to significantly attack the Triangle Test. In [Kirchner and Böhme, 2009], the authentic CFA is substituted with another target pattern maximising the PSNR between the input image and the output image containing the desired pattern.

8.4.6 Counter-forensics as a Game Theory problem

In order to better understand the ultimate performance of the forensic analysis in presence of an intelligent adversary, the research community focused more and more on developing rigorous theoretical frameworks in which to cast the forensic and counter-forensic problems. The formalisation in Secs. 8.2–8.3 is not the only one proposed so far. Very recently, in fact, the two problems have been also cast into a game-theoretical framework [Barni, 2012; Stamm et al., 2012]. Game theory is a branch of mathematics devoted to the analysis of strategic situations, referred to as games, in which the success of one player depends on the choices made by the other player(s). This particular formulation, in fact, elegantly allows to understand under which conditions the challenge between adversary and analyst has a winner.

A first step in this direction is the one in [Stamm et al., 2012], where the effectiveness of an adversarial strategy is evaluated and the corresponding optimum forensic countermeasure is derived. A more rigorous and general approach is proposed by Barni and Tondi (see [Barni, 2012] and [Barni and Tondi, 2013]). In particular, the authors focus on the source identification game with known statistics. Under certain assumptions, they demonstrate that optimum strategies for both the analyst and the adversary can be derived and numerically evaluated. In [Barni and Tondi, 2012], the analysis is further extended to the case where the statistics of the source are known to the adversary only through training data.

8.5 Concluding remarks

This chapter introduced Counter-forensics, that is the art of misleading image forensic analysis by hiding, removing or altering the traces identifying a specific image manipulation. Several benefits are brought by counter-forensic research, the most evident of which is the impulse to improve existing tools or to devise new and more robust tools. Even though this discipline is still in its infancy, several popular forensic techniques have been successfully impaired. Moreover, the first attempts of casting the counter-forensic problem into a rigorous framework have been proposed.

Despite the satisfactory results obtained so far, there are still some cate-

gories of forensic algorithms that have not been challenged, like those based on physical or geometrical properties of the image scene or those based on salient point detectors. As it has been observed in Chapter 2, the former typology of algorithms is mainly used to detect cut & paste forgeries by looking for physical or geometrical inconsistencies in an image (e.g. [Johnson and Farid, 2008, 2005]). Regions where such inconsistencies are found are considered tampered, based on the assumption that it is very hard to forge an image in such a way that properties like perspective, directions of light or shadows of the spliced region are coherent with those of the authentic content.

Similarly, countering techniques based on salient point detectors (e.g. SIFT, SURF) presents several difficulties. The features extracted by such detectors, in fact, are robust or invariant to geometrical manipulations, thus permitting very accurate matching between similar or identical images or portion of images. For example, this is the reason behind the good performance of copy-move forgery detectors based on such feature extractors [Pan and Lyu, 2010; Amerini et al., 2011]. In the next chapters we will try to exploit the weaknesses of the most reliable category of copy-move detectors, that is to say those based on the most popular of feature extractors, i.e. the Scale Invariant Feature Transform (SIFT).

Chapter 9

The SIFT algorithm

"No matter how the wind howls,
the mountain cannot bow to it"

The Emperor of China
MULAN

THE AIM of this chapter is to provide the reader with a detailed description of the *Scale Invariant Feature Transform* (SIFT) algorithm. Knowing the concepts underlying SIFT, in fact, is the key to understanding its strengths and weaknesses. By relying on such information, we will be able later in the thesis to develop methods to either undermine (Chapters 10, 11 and 13) or support (Chapter 12) SIFT-based forensic analysis.

The outline of the chapter is as follows. First, some pointers to the state-of-the-art of local features detection are given in Sec. 9.1, where the terminology that will be adopted for the rest of the discussion is also provided. Then, in Sec. 9.2, all the stages composing the SIFT workflow are analysed in depth. To conclude, in Sec. 9.3 the most interesting extensions of SIFT are described.

9.1 Multi-scale feature detection

The research preceding the development of SIFT had devised a great deal of feature extractors. An exhaustive overview of such techniques is provided in [Tuytelaars and Mikolajczyk, 2008], where the authors track back to 1954 the origins of the interest on finding relevant and robust points or small regions of an image [Attneave, 1954]. Since this early work, a vast amount of techniques have been developed, improved and finally put aside in favour of more performing approaches. Without going into details here, it suffices to say that

there exist several categories of methods: based on edges or corners, on image intensity, on color, on derivatives, on human visual system, on photometric models and so on. As a matter of fact, all these early techniques suffered from a general lack of robustness.

Following the technology advancements of the last two decades and the worldwide diffusion of more demanding applications based on images and videos, the research community felt the need of features that could be not just relevant and unique, but also robust, when not completely immune, to the most common image processing tools. More specifically, the aim was to extract features not affected by geometric manipulations such as translation, scaling, rotation, reflection or combinations of them.

Among these transformations, scaling was the most critical operation to deal with. Often, not only in Computer Vision but also in physics, the best scale to analyse a certain phenomenon is not known a priori or not trivial to determine with theory or experiments. Therefore, a sound strategy should consist in including the information coming from different scales, rather than choosing a single scale and focusing only on it. In Computer Vision, this approach is referred to as *multi-scale analysis* and has been studied since the early 1980's [Burt, 1981; Crowley and Parker, 1984; Lindeberg, 1994]. In its most refined version, such approach first selects an initial scale and then extracts structures of the input image at that scale. Then, the scale is increased to obtain a coarser version of the input image and its structures are extracted again. The purpose of this approach is to separate the structures of the original image among the different scales, in such a way that finer scale structures are not propagated to coarser scales and no new structures are introduced during the process. Usually, the whole procedure is repeated from the beginning after the original image's size has been reduced. By relying on this representation, the features are computed from the image structures across different scales, in such a way that robustness to scale changes is achieved. Interestingly, this approach is surprisingly close to the way visual system of primates (and possibly of humans) works [Tanaka, 1997].

All the works on scale-space representation and its applications, in particular those by Lindeberg [Lindeberg, 1994, 1998], have been of fundamental importance to the ideas underlying SIFT. There is no doubt that SIFT has

been one of the most well-engineered Computer Vision algorithms of the last decade. With almost $30,000$ academic citations[1] and a U.S. Patent,[2] the work by David G. Lowe, introduced in 1999 [Lowe, 1999] and further refined in 2004 [Lowe, 2004] in two seminal papers, has been successfully employed in a huge variety of scientific fields, including scene (object) recognition and detection, image retrieval, image registration, image forgery detection, panorama stitching, automated navigation, tracking and 3D modeling. In [Mikolajczyk and Schmid, 2005], the authors documented the superiority of SIFT performance with respect to similar methods in most of the tests they have conducted.

9.1.1 Terminology

Before analysing in detail the technical aspects of SIFT, we introduce the terminology that will be adopted for the rest of the discussion [Tuytelaars and Mikolajczyk, 2008]. The term *feature* denotes a piece of information that is extracted from an image by a *detector* (or extractor) and that is relevant to the solution of some specific problem. If such feature is a pattern extracted from a subpart of the image and different from its immediate neighbours, then it is called *local feature*. Commonly, a local feature is extracted from a rectangular, circular or elliptical pixel neighbourhood centred in a point of interest. The visual content of these patches is often not as relevant as their spatial location and statistical or mathematical properties. These properties are represented in a convenient, highly descriptive and compact form which is often referred to as *descriptor*. To be of some practical use, local features must possess the following three main properties: *distinctiveness*, *invariance*, and *robustness* (sometimes called quasi-invariance).

- **Invariance**. A feature is invariant under a certain set of transformations $\mathcal{T}$ if its values are not affected when a transformation $t \in \mathcal{T}$ is applied. The most important invariances are those to geometric transformations and to lightning variations.

[1] So far (January 2014), the articles were referenced $7,042$ and $22,194$ times respectively.

[2] United States Patent number 6711293, "*Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image*", released on 23 March 2004 and owned by the University of British Columbia, Canada.

- **Robustness**. A feature is robust to a certain processing (e.g. noise addition, quantisation, compression) if its values are not altered excessively when such a processing is applied. This result is achieved at the cost of some loss in accuracy by reducing the impact of that specific processing on the detector.

- **Distinctiveness**. A feature is distinctive if its underlying patterns show many variations, thus making it easily distinguishable from the other features.

Ideally, invariance is the property one should always grant. However, full invariance is hard to achieve in practice, hence robustness also plays an important role. These two properties are sometimes summarised with the term *repeatability*. In addition to the above properties, good features should also possess: *locality*, i.e. support should be small enough to describe only local characteristics; *quantity*, i.e. amount of features should be sufficiently large but not excessive; *accuracy*, i.e. localisation in space and scale should be precise; and *efficiency*, i.e. computational cost should be low.

Unfortunately, it is unlikely that all local features have the three main properties outlined above; a feature missing at least one of them is considered *unstable* or *unreliable*. Unstable features should be discarded, as they suffer a number of drawbacks which may impact on accuracy, and even their mere presence raises the complexity of the system and reduces its scalability. Some unstable features are however inevitable, in consideration of the number of trade-offs existing between their properties. Among them, we mention those between distinctiveness and locality, distinctiveness and invariance, distinctiveness and robustness. In the first case, the smaller the feature neighbourhood, the less the feature can effectively describe the underlying pattern. In the second and third cases, the processing required to achieve repeatability flattens the variations of intensity patterns, thus making the features less distinguishable.

9.2 Scale Invariant Feature Transform

According to the terminology of Sec. 9.1, SIFT computes multi-scale distinctive local features that are invariant to scale and rotation and robust, to a variable extent, to affine distortions, changes in illumination, changes of 3D viewpoint, cluttering, occlusions and noise addition. These features are obtained from the neighbourhoods of salient points referred to as *keypoints*; SIFT provides a high amount of keypoints, even if dependent on image's content and size, densely distributed across the image, at a low computational cost.

The method consists of four stages: i) construction of the scale-space representation of the input image; ii) keypoint localisation and refinement; iii) assignment of canonical orientation; and iv) construction of the descriptor. First, the algorithm identifies good candidate locations across all scales; then, locations are refined according to some criteria of stability; in order to achieve rotation invariance, to each keypoint a principal orientation is assigned; finally, the neighbourhood of each keypoint is used to elaborate an unique descriptor.

9.2.1 Scale-space representation

The first SIFT stage localises candidate features that are invariant to scale changes. The input image is represented in such a way that the effect of different scales can be conveniently simulated. Such representation is the so called *scale-space representation*, which, for the sake of clarity, is introduced here in its two-dimensional formulation. The reader is referred to [Lindeberg, 1994] for the straightforward generalisation to an arbitrary number of dimensions.

Let $f : \mathbb{R}^2 \to \mathbb{R}$ be a 2D signal. Then, the scale-space representation $L(x, y; \sigma)$ of f is the convolution of f with a function $G : \mathbb{R}^2 \times \mathbb{R}_+ \to \mathbb{R}$:

$$L(x, y; \sigma) = G(x, y; \sigma) * f(x, y), \tag{9.1}$$

where $G(x, y; \sigma)$ is the Gaussian kernel such that:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{9.2}$$

The width of the Gaussian kernel, i.e. $\sigma \in \mathbb{R}_+$ is referred to as the *scale parameter*. There exist multiple ways to demonstrate that the Gaussian kernel is the only possible kernel for scale-space analysis [Lindeberg, 1994].

In order to obtain stable candidate keypoints, SIFT makes use of an extractor named *Difference of Gaussians* (DoG), which is defined as follows. Let $k\sigma$ and σ be two consecutive scales separated by a constant $k \in \mathbb{R}_+$ and let $L(x, y; k\sigma)$ and $L(x, y; \sigma)$ be their corresponding scale-space representations. Then, the DoG $D(x, y; \sigma)$ is obtained by subtracting the two scale-space representations, as follows:

$$\begin{aligned} D(x, y; k\sigma) &= L(x, y; k\sigma) - L(x, y; \sigma) \\ &= (G(x, y; k\sigma) - G(x, y; \sigma)) * f(x, y). \end{aligned} \tag{9.3}$$

The candidate keypoints are selected as the maxima and minima of $D(x, y; \sigma)$.

The DoG can be efficiently computed across all adjacent scales of the levels of a multi-scale pyramid, as pictured in Fig. 9.1.

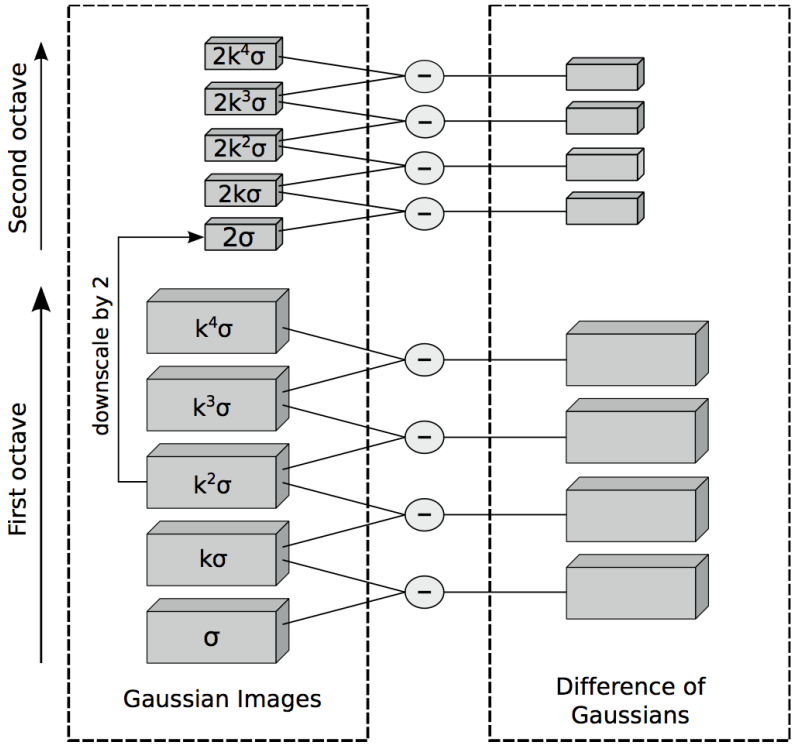


Figure 9.1: *Construction of the Difference of Gaussians (DoG).*

In practice, the input image is pre-smoothed ($\sigma_0 = 0.5$) and the result is re-

peatedly convolved with a Gaussian kernel, whose width σ (default $\sigma = 1.6$) increases by a multiplicative factor k at each step, to obtain progressively smoother images. The procedure is repeated $s+3$ times, where s is a parameter of the algorithm (default $s = 2$). The stack of Gaussian images constitutes an *octave* of scale-space (left column of Fig. 9.1). Then, Gaussian images at adjacent scales are subtracted and the Difference of Gaussians for a full octave are obtained (right column). At this point, the whole procedure is repeated for the next octaves, which are computed by downsampling by a factor 2 the Gaussian image of width $k^2\sigma$ of the preceding octave. If a larger amount of keypoints is required (usually on small images), the initial octave can be computed on the input image upscaled by factor 2.

The choice of the parameter k requires some additional considerations. In multi-scale analysis, the scale-normalised Laplacian of Gaussians (LoG) $\sigma^2\nabla^2 G$ is an operator well-known for providing very stable and truly scale-invariant extrema [Lindeberg, 1998; Mikolajczyk, 2002]. The Difference of Gaussians is a very efficient approximation of the normalised LoG [Lowe, 2004]; more precisely, their relationship is the following:

$$G(x, y; k\sigma) - G(x, y; \sigma) \approx (k-1)\sigma^2\nabla^2 G, \tag{9.4}$$

where $(k-1)$ represents a constant approximation error for all scales. It has been observed experimentally that the effect of k on the stability of DoG extrema is negligible, as long as its value is not much greater than 1. In the standard SIFT implementation such value is set to $k = 2^{1/s}$ (default $k = \sqrt{2}$).

9.2.2 Keypoint localisation and refinement

The method for localising stable local extrema of $D(x, y; \sigma)$ is displayed in Fig. 9.2. Each extremum at scale n is compared with all the DoG values within a 3×3 neighbourhood at the current scale n and at the two adjacent scales $n-1$ and $n+1$. If this value is the largest or the smallest, the point is an extremum. These 26 fast checks ensure that two keypoints do not fall too close to each other.

The extrema are just approximated locations, as they may not lie exactly on a pixel, thus requiring a resolution higher than the sampling density. While

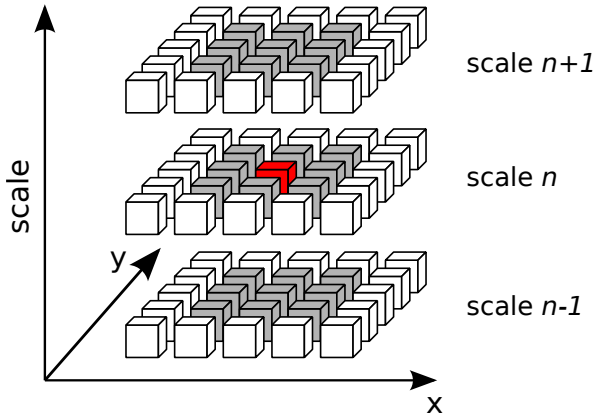


Figure 9.2: *Localisation of DoG extrema.*

this improvement may not be essential for lower octaves, it becomes fundamental for higher ones, where small differences in measurement correspond to several pixels in the base image. Therefore, the locations of extrema need to be further refined to sub-pixel accuracy, in order to increase the stability of the keypoint (the usefulness of this step has been proved experimentally in [Brown and Lowe, 2002]). To this aim, a quadratic polynomial is fitted to the DoG's magnitude values around each extremum, by means of the second-order Taylor expansion:

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}, \tag{9.5}$$

where D is the DoG computed in the keypoint and $\mathbf{x} = (x, y, \sigma)^T$ is the offset from that keypoint. The solution $\hat{\mathbf{x}}$, obtained by differentiating with respect to $\mathbf{x}$ and equating to zero, corresponds to the refined location of the keypoint:

$$\hat{\mathbf{x}} = \frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}. \tag{9.6}$$

The localisation stage generally produces a high number of candidate keypoints. However, not all these keypoints are stable in the presence of noise. Therefore, additional checks must be performed at this point. Unstable keypoints are ruled out by eliminating extrema whose contrast is too low or whose edge response is strong only in one direction.

(A) Eliminating low contrast extrema

Noise can generate extrema whose absolute values are not large enough. Therefore, they are discarded if they do not pass the following check:

$$|D(\hat{\mathbf{x}})| = \left| D + \frac{1}{2} \frac{\partial D}{\partial \mathbf{x}}^T \hat{\mathbf{x}} \right| < T_{contrast}. \tag{9.7}$$

First, the contrast value is computed in the normalised interval $[0, 1]$ of DoG by conveniently resorting again to Eqs. (9.5)–(9.6). Then, the absolute value of the contrast is compared with a threshold $T_{contrast}$ empirically set to 0.03 in [Lowe, 2004].

(B) Eliminating weak edge responses

The second stability check is based on the following rationale: keypoints sitting on top of edges are not robust, as noise can move them along the edges. On the contrary, keypoints nearby corners are very robust, as their location is, intuitively, "anchored". Consequently, the former should be discarded while the latter should be kept. This discrimination is based on the principal curvature [Schoen and Yau, 1994] of the candidate, that is a measure of how the surface around it bends in different directions.

In practice, SIFT proceeds in a way that is inspired to the Harris corner detector [Harris and Stephens, 1988]. For the sake of clarity, let D be the difference of Gaussians and let H be the Hessian matrix of second derivatives:

$$H = \begin{bmatrix} \frac{\partial^2 D}{\partial x^2} & \frac{\partial^2 D}{\partial x \partial y} \\ \frac{\partial^2 D}{\partial x \partial y} & \frac{\partial^2 D}{\partial y^2} \end{bmatrix}. \tag{9.8}$$

Now, let (α, β) be the eigenvalues of $H(x, y; \sigma)$, i.e. the derivatives computed at the keypoint. Then, the principal curvatures are proportional to the magnitude of α and β:

- if both (α, β) are small, then the region of interest is flat, i.e. its intensity values are approximately constant. This case does not occur in SIFT following the elimination of low contrast keypoints;

- if either α or β is high and the other is low, then the region of interest is an edge;
- if both (α, β) are high, then the region of interest is a corner.

By relying only on the trace and the determinant of H, one can avoid the direct evaluation of eigenvalues, thus defining the following metric R quantifying the edge/corner response of a keypoint:

$$R = \frac{\mathrm{Tr}[H(x,y,\sigma)]^2}{\mathrm{Det}[H(x,y,\sigma)]} = \frac{(H_{xx}+H_{yy})^2}{H_{xx}H_{yy}-H_{xy}^2} = \frac{(\alpha+\beta)^2}{\alpha\beta} < T_{edge}. \tag{9.9}$$

In practice, if R is smaller than a threshold T_{edge}, then the keypoint is discarded. Let now α be the largest of the eigenvalues and r be their ratio, $r = \alpha/\beta$. Then, the threshold T_{edge} can be conveniently expressed in terms of r as follows [Lowe, 2004]:

$$T_{edge} = \frac{(r+1)^2}{r}. \tag{9.10}$$

The above measure has a minimum value of 4 when both α and β are equal (i.e. in proximity of curved peaks) and increases as one of the eigenvalues grows and the other remains small (i.e. near edges). Similarly to the contrast threshold, the value of r is a configurable parameter of the SIFT algorithm and has been set empirically to $r = 10$.

9.2.3 Assignment of keypoint orientation

All the candidates that survived the refinement process are stable keypoints. While the previous stage was intended to confer scale invariance, the current stage is meant to ensure rotation invariance. This task is executed by assigning to each keypoint a dominant orientation, in such a way that the keypoint remains recognisable when the image is arbitrarily rotated.

First, the algorithm selects $L(x,y;\sigma)$ according to the keypoint's scale σ. Then, for each spatial coordinate, the gradient magnitude and orientation are computed:

$$m(x,y) = \sqrt{\big(L(x+1,y)-L(x-1,y)\big)^2 + \big(L(x,y+1)-L(x,y-1)\big)^2} \tag{9.11}$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right). \tag{9.12}$$

Once $m(x, y)$ and $\theta(x, y)$ have been computed, orientations are first weighted by their magnitudes and by a circular Gaussian window[3] of width $\sigma_G = 1.5\sigma$ and then organised into a histogram of 36 bins, each covering 10 degrees. As shown in the example of Fig. 9.3, the dominant orientation assigned to the keypoint corresponds to the highest peak of the histogram. If the histogram has other peaks whose height is comparable to that of the dominant orientation ($\geqslant 80\%$ of the dominant peak), for each of them a new keypoint, up to a maximum of 3, is generated at the same location and scale but with different orientation.

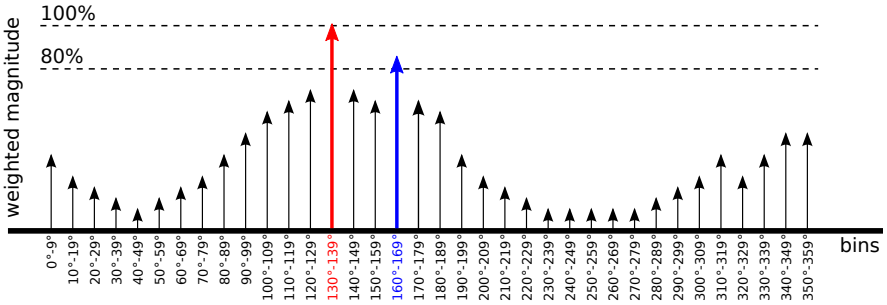


Figure 9.3: *Assignment of dominant orientation(s).*

At this stage, a keypoint $\mathbf{x}$ is represented by a quadruple $\{x, y, \sigma, \theta_{max}\}$, also called *frame*, where: (x, y) are the spatial coordinates; σ is the scale; θ_{max} is the dominant orientation.

9.2.4 Computation of SIFT descriptor

The final stage consists in computing the descriptor, that is a compact representation of the region surrounding the keypoint, as shown in Fig. 9.4. SIFT borrows the idea of histograms of gradients from previous works (see [Schiele and Crowley, 2000]). First, the 16×16 patch centred on the keypoint (at the relevant scale σ) is rotated relatively to the dominant orientation, in order to ensure rotation invariance. Secondly, the gradients of patch's pixels are computed and weighted by a circular Gaussian window of width equal to half of the patch size (hence, 8). This step is meant to grant robustness to small

[3]The window needs to be circular in order to be fully invariant to rotation.

translations of the window. Thirdly, the patch is decomposed into 16 tiles of size 4×4. For each such tile, the histogram of gradients is computed, with each of the 8 bins covering 45 degrees.

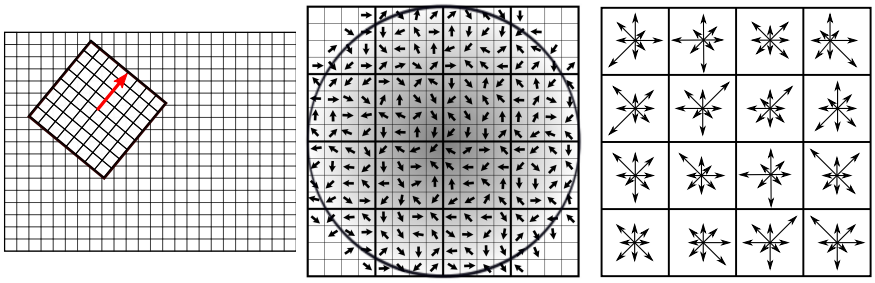


Figure 9.4: *Construction of SIFT descriptor: rotation along the dominant orientation (left); Gaussian weighting (middle); histograms of gradients (right).*

At this point, the values of the 16 tile histograms are read in counterclockwise order and concatenated into the descriptor, thus forming a vector of $16 \times 8 = 128$ elements. Finally, to attain additional robustness to changes of illumination, the descriptor is normalised in $[0, 1]$ and all values above 0.2 are thresholded and normalised again (the usefulness of this last step has been proved experimentally).

9.2.5 Matching SIFT descriptors

The original goal of SIFT was to compare two images (or parts of them) that can be deduced from each other or from another common image; to this end, keypoints are detected in both images and then matched. The simplest way to find a correspondence between two descriptors $\mathbf{p}$ and $\mathbf{q}$ is to verify whether their distance $d(\mathbf{p}, \mathbf{q})$ falls below a certain threshold. For instance, d can be the Euclidean distance in the 128-dimensional space of descriptors, i.e. $d(\mathbf{p}, \mathbf{q}) = \left[\sum_{i=1}^{128}(p_i - q_i)^2\right]^{1/2}$. However, considering the typically large amount of keypoints and the fact that not all descriptors are equally distinctive, any attempt of setting a global threshold may end causing false matches. The solution implemented by SIFT to tackle with this problem is the follow-

ing. Given a keypoint in the first image, the matching candidate at closest distance d_1 (*nearest neighbour*) in the second image is found. Then, the second best matching candidate at distance d_2 is also found. If the ratio between d_1 and d_2 exceeds a threshold, then the match is rejected. The rationale behind this check is that a match is considered correct only if the closest neighbour is sufficiently closer than the closest incorrect match. Based on experimental tests on several thousands of keypoints, Lowe set $d_1/d_2 < 0.8$ [Lowe, 2004]. In practical application, images with thousands of keypoints need to be rapidly compared with large databases, thus making the exhaustive research of nearest neighbour costly and inefficient. There exist several improvements for this and similar scenarios, including the use of specialised structures for efficient nearest-neighbour queries (e.g. KD-trees [Bentley, 1980]) and the reduction of dimensionality of the descriptor.

9.3 Extensions of the SIFT algorithm

In the last years, several variants of the original SIFT algorithm have been developed. Although all these extensions claim improved distinctiveness or robustness at a reduced computational cost, their comparison is not a trivial task, since their performances are often strongly related to specific applications. Among the most useful variants of SIFT, we mention: including color information into the descriptor [Van De Weijer and Schmid, 2006; Abdel-Hakim and Farag, 2006]; reducing the computational burden by either sampling keypoints [Bosch et al., 2006; Foo and Sinha, 2007] or by shrinking the descriptor [Ke and Sukthankar, 2004]; increasing robustness of the descriptor [Mikolajczyk and Schmid, 2005]; and ensuring invariance to affine transformations [Yu and Morel, 2011]. In the following, these techniques are briefly described. For a more in-depth analysis, the reader is referred to the respective papers.

9.3.1 Color SIFT

SIFT discards color information and works exclusively on the grayscale image. Such choice is justified by the fact that both global and local color features are often not robust to changes of illumination or viewpoint, shading and specularity. However, especially in image retrieval applications, the color is

highly distinctive and can be used to find similarities across images. As a consequence, the research community has proposed three different strategies to incorporate color information into SIFT. The first and simplest strategy is to compute SIFT keypoints separately for each image channel and then use them to find matches channel by channel. This task can be performed in any convenient color space, be it RGB, red-green blue-yellow opponent (IUV) [Hall et al., 2000] or Gaussian color model [Geusebroek et al., 2000]. The second strategy consists in computing the SIFT descriptors of the grayscale image and then augment them with color features. Finally, the third strategy consists in using the three-channel color image as input to a modified SIFT, which computes a color descriptor by means of a multi-scale pyramidal approach.

[Van De Weijer and Schmid, 2006] is the first work dealing with color SIFT and belongs to the second category. In this method, the canonical SIFT descriptor is augmented with invariant color features. More specifically, the authors compute four histograms (three of 37 bins and one of 121 bins) of color features invariant to photometric changes, according to a previously assumed physical reflectance model. All the histograms are weighted and concatenated to the canonical SIFT descriptor, which becomes a vector of 360 elements.

Inspired by the previous work, [Abdel-Hakim and Farag, 2006] proposed a method named *Colored*-SIFT (shortly, C-SIFT), which belongs to the third category. First, the input image is converted by means of a linear combination into a suitable Gaussian opponent color model, and invariant color channels are obtained according to a reflectance model. At this point, the method proceeds similarly to SIFT, by localising keypoints as extrema of DoG computed on the new channels. The procedure to obtain the descriptor is again similar to the one used by the original SIFT, but computed on gradients of color invariant channels rather than on grayscale gradients.

In the subsequent years, several variants of these approaches have been introduced, including HSV-SIFT, *rg*SIFT, RGB-SIFT and Opponent-SIFT [Van De Sande et al., 2010]. In [Burghouts and Geusebroek, 2009], some of these methods are compared against grayscale SIFT and the experiments documented that matching color descriptors grants slightly better results than grayscale matching. In particular, while the two approaches are equally invariant, color based descriptors show a greater discriminative power. In [Van

De Sande et al., 2010], several color descriptors are compared against each other. Although some of the descriptors appear to perform better than others (C-SIFT above all), there is no golden rule to determine the best one for all purposes, as performance are also related to the application, to the data set at hand and to image properties.

9.3.2 Dense SIFT

Dense-SIFT is a method to reduce the amount of sparse keypoints that are generated by the original SIFT. Such method has been introduced in [Bosch et al., 2006] and further developed in [Bosch et al., 2007]. As a matter of fact, not all SIFT-based applications can afford to elaborate the thousands of keypoints usually detected in natural images. A typical example is the classification of scenes or objects, where maximising accuracy while minimising the computational cost of comparing many keypoints against large databases is paramount. In order to be effective in these scenarios, SIFT keypoints need to be represented in a more compact fashion without compromising their invariance.

Dense-SIFT does not rely on the DoG detector for localisation, but instead forces the keypoints to fixed coordinates, where descriptors are then computed as usual. The analysis can be carried out either at fixed scale and orientation as in [Bosch et al., 2006], or at multiple scales as in [Bosch et al., 2007]. The latter solution is called PHOW (Pyramidal Histogram Of visual Words). Either way, the method begins by superposing to the input image a $M \times M$ grid of points (usually M is 5, 10 or 15), both horizontally and vertically. Then, at each point of the grid descriptors are computed over circular grayscale patches with $n > 1$ increasing radii r (usually $n = 4$ and $r = 4, 8, 12, 16$). As a result, each point of the grid is univocally characterised by n descriptors of 128 elements, which are supposed to take into account all the SIFT scales.

This model can be further elaborated to exploit the color information of the circular patches. In this case, the image is converted to the HSV color space and the above procedure is repeated separately for each channel, thus leading to n descriptors of 3×128 elements for each point of the grid (Color Dense-SIFT or PHOW-color). The information conveyed by Dense-SIFT descriptors can be further compressed: first, the descriptors are clustered; then,

a predetermined visual word is assigned to the centroid of each cluster (this approach is referred to as Bag of Keypoints [Csurka et al., 2004]); and finally, the histogram of different visual words contained in the image is stored and used for efficient comparisons.

9.3.3 PCA-SIFT

PCA-SIFT [Ke and Sukthankar, 2004] is basically a different approach to the calculation of descriptors. According to the authors, the standard SIFT descriptor, although wisely designed to cope with local pixel changes, is redundant and could be represented more compactly. Therefore, they propose an alternative shorter descriptor of 20 elements rather than 128. The PCA-SIFT begins by computing the spatial location, scale and dominant orientation of keypoints by means of the standard SIFT technique. Then, for each keypoint, a 41×41 patch (centred on the keypoint, rotated according to its canonical orientation) is extracted at the corresponding scale and rotated in such a way that the dominant orientation becomes vertical. The local vertical and horizontal gradients of each patch are then computed, concatenated and slightly cropped, thus producing a 78×39 patch, which is subsequently rearranged into a vector of 3042 elements. By assuming that the vectors can be modeled as Gaussian distributions, their dimensionality is reduced to 20 by means of Principal Component Analysis (PCA) [Jolliffe, 2002]. The authors compared the performance of image matching based on PCA-SIFT and original SIFT descriptors and came to the following conclusions: increasing the dimensionality of the PCA-SIFT descriptor does not bring any noticeable advantage; PCA-SIFT descriptor is better suited to cope with errors introduced during keypoint localisation; SIFT descriptor is better suited to cope with errors introduced during the assignment of dominant orientations. Moreover, according to [Mikolajczyk and Schmid, 2005], PCA-SIFT obtains better matching scores than SIFT for low false positive rates.

9.3.4 Affine SIFT

A change of viewpoint introduces distortions that can be modeled as affine planar transformations consisting of six parameters: translation along the two axes, zoom (scale), rotation and the two angles defining camera orientation. SIFT does not take into account the last two parameters and thus it is not affine invariant. Moreover, Lowe observed that SIFT matching accuracy nears 50% for viewpoint changes whose angle exceeds 50 degrees [Lowe, 2004]. The goal of Affine-SIFT (ASIFT) [Yu and Morel, 2011] is to introduce full affine invariance. To do so, it simulates changes in view by varying the two camera axis orientations (the latitude and the longitude) and exploits SIFT's invariance to the remaining four parameters. ASIFT has been designed only for applications that require matching. In a nutshell, it consists of three stages: i) simulation of all affine transformations; ii) research of best transformations; and iii) SIFT matching. More specifically, it proceeds as follows.

Let A and B be two images, represented by squares in Fig. 9.5. ASIFT creates n versions of A, each of which corresponds to an affine transformation with different camera angles θ and ϕ, called respectively latitude and longitude. In practice, each copy is first rotated by ϕ degrees and then tilted by means of directional sub-sampling with tilt $t = \frac{1}{\cos\theta}$. The whole procedure is repeated again for B. Such images are represented with blue parallelograms in Fig. 9.5. Let now $\mathcal{S}_A$ and $\mathcal{S}_B$ be the sets of newly generated images (including respectively A and B). In the next stages, first all the images of $\mathcal{S}_A$ are matched with all the images of $\mathcal{S}_B$ by means of standard SIFT (the arrows of Fig. 9.5). The image pairs yielding a satisfactory number of matches are used to map the matching keypoints into the starting images A and B. To keep the potentially high computational cost under control, the angles (ϕ, θ) are sampled and a two-resolution approach is adopted: the algorithm is initially applied to a low resolution version of the images and then only the best transformations are simulated again on the high resolution images. The authors observe that the time complexity of high resolution ASIFT is approximately twice the complexity of normal SIFT. Such an increase, however, is balanced by the full affine invariance (mathematically demonstrated in [Yu and Morel, 2011]) and by the superior performance when it is necessary to match images that have undergone a critical change of viewpoint.

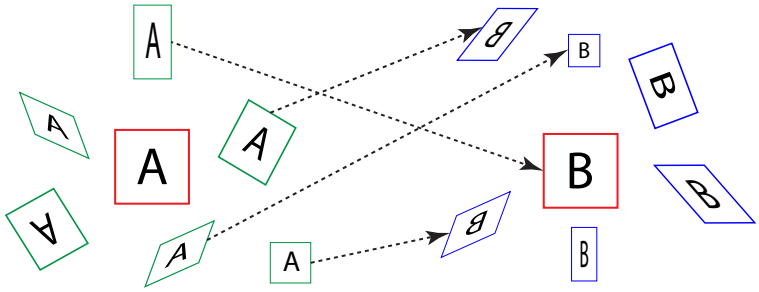


Figure 9.5: *Scheme of Affine-SIFT* [Yu and Morel, 2011]. *The images A and B (squares) are distorted by simulating camera angles (parallelograms and rectangles). The simulated images are matched with SIFT (dashed arrows).*

9.4 Concluding remarks

In this chapter we reviewed the SIFT algorithm. Such an analysis is necessary to introduce the ideas and the concepts illustrated in the sequel of the dissertation. Both adversarial schemes countering SIFT-based detection and their forensic nemeses, in fact, leverage on very specific aspects of SIFT algorithm, which need to be fully grasped.

Chapter 10

SIFT keypoint removal

THE COUNTER-FORENSIC principles of Chapter 8 are now put in practice to assess the security of SIFT-based forensic algorithms. Several compelling reasons push towards such study: they are discussed in Sec. 10.1. The few existing attacks developed so far to this purpose are outlined in Sec. 10.2. The principal contribution brought by this part of the thesis lies among them, in the form of a new keypoint removal scheme, namely the *Classification-based attack*. Such attack is detailed in Sec. 10.3 and experimentally validated in Secs. 10.4–10.5. The chapter is concluded by a short section providing additional examples of images manipulated with the aforementioned attack.

10.1 Motivations and contributions

The robustness of SIFT is universally acknowledged by the countless real-world applications based on it. However, *what about SIFT security? Is it possible for an adversary to remove SIFT features from images so that the correct detection of the algorithm is jeopardised?* Recently, the research community has begun to seek answers to these questions. It came out that the feat is not

simple but indeed feasible, as a handful of attacks available so far confirm. The most serious consequence of these studies is the awareness that if SIFT can be compromised, then all those applications relying on it may not be as secure as we thought.

At present, several applications dealing with sensitive information rely on SIFT, including: image authentication and copy-right enforcement [Lv and Wang, 2012], image retrieval [Do et al., 2010a], filtering of illegal content [Lopes et al., 2009], face and fingerprint authentication [Park et al., 2008] and detection of image forgeries [Amerini et al., 2011]. Nowadays, scenarios in which a forged image is deemed authentic and used as evidence of a fact that has never happened are alarmingly at reach. Therefore, fixing security flaws of SIFT-based applications and, if necessary, developing new and more secure techniques is rapidly becoming a pressing need. There is more. Now, it is even possible to recover the content of an image by resorting only to its SIFT descriptors [Weinzaepfel et al., 2011]. This is a significant menace to the privacy of the content's owner, should a database of SIFT features leak information.

Some interesting results on attacks against SIFT were obtained so far, although the state of infancy of this research is betrayed by the difficulty of balancing strength and perceptibility of the attacks. Quite often, in fact, the efforts of very effective attacks are frustrated by the introduction of unacceptably noticeable artefacts in the manipulated image. To overcome such limitations and to push the research forward, this part of the thesis proposes a new attack to SIFT keypoints. In particular, the intuition behind it lies on abandoning the well-established hypothesis that all SIFT keypoints have the same characteristics. The direct consequence of this assumption is that all keypoints are attacked in the same way, and often this is not the wisest of the strategies. Conversely, we demonstrate that two things are possible: *to discriminate between different categories of keypoints* and *to attack each category with customised attacks*. These intuitions are put in practice by a two-stage scheme (classification followed by *ad-hoc* attacks), that outperforms state-of-the-art-techniques and provides a good trade-off between efficacy of removal and perceptual quality of the manipulated image.

10.2 State-of-the-art of SIFT countering

Essentially, the attacks to SIFT detection can be catalogued according to how they impair the correct functioning of the SIFT algorithm: by removing authentic keypoints to trigger false negatives; by introducing fake keypoints to trigger false positives; or by modifying properties of authentic keypoints (e.g. orientation). Depending on the targeted application, the adversary may adopt one or more of these solutions.

The first attempt to test the security of SIFT has been made in [Hsu et al., 2009]. In this early work, the authors were able to compromise an authentication system based on SIFT and image hashing by deleting keypoints. More specifically, Hsu *et al.* devised two different attacks. The first attack is the Collage attack, that is the replacement of image blocks containing keypoints with blocks not containing them. The substitution is carried out according to a criterion of similarity based on Mean Square Error. The second attack superposes a local patch to the keypoint neighbourhood in such a way to introduce a second fake extremum close enough to the authentic one. By doing so, the refinement check (see Sec. 9.2.2) is forced to discard the authentic keypoint.

In 2010, Do *et al.* applied the latter technique to assess its potential threat on a SIFT-based Content Based Image Retrieval (CBIR) scenario [Do et al., 2010b]. It turns out that their CBIR system is not affected by Hsu *et al.*'s attack, since all the manipulated images are correctly indexed and their authentic versions are retrieved. The major drawback with Hsu *et al.*'s method consists in the fact that it frequently creates new keypoints in proximity of those that have been removed. Due to similarity of the corresponding descriptors, the final effect resembles more a keypoint translation than a removal.

Following this analysis, Do *et al.* focused again on the CBIR scenario, this time by devising new attacks to spatial locations [Do et al., 2010a] and to dominant orientations [Do et al., 2012] of keypoints. Two important contributions came from [Do et al., 2010a]. The first one is the Removal with Minimum Distortion (RMD) attack, i.e. a scheme capable of removing a keypoint by pushing its contrast value below SIFT's threshold. Following a symmetrical approach, low contrast values are pushed above the same threshold by means of the Forging with Minimum Distortion (FMD) attack. The second contribution is the intuition that, in practical applications, concatenating multiple

attacks may improve the final outcome. Finally, in [Do et al., 2012], matches between descriptors are inhibited by manipulating the orientation of keypoints. The authors achieve such goal by applying a visual patch, whose coefficients have been learned by means of Support Vector Machines forcing the gradient to rotate.

Recently, the CBIR scenario has been taken into account again in [Lu and Hsu, 2012], where the authors proposed two techniques for removing and adding keypoints, formulated in terms of a constrained optimisation problem. Finally, in [Caldelli et al., 2012] SIFT keypoints were removed by means of local warping attacks derived from image watermarking [D'Angelo and Barni, 2008].

10.3 A new approach: classification-based attack

The attacks proposed so far to remove SIFT keypoints are based on the assumption that all the keypoints of an image have equal properties. The consequence of this hypothesis is that such methods are applied indifferently to all the keypoints of an image. To be more specific, the only adaptive solution consisted so far in increasing the strength of the attack proportionally to the scale (and thus to the robustness) of the keypoint. Our idea is that, in general, the effectiveness of an attack may be strictly related to the properties of the keypoint that must be removed. As an example, suppose that the neighbourhood of a keypoint contains a straight vertical edge; a local warping attack, such as the one in [Caldelli et al., 2012], would probably succeed in deleting it. Unfortunately, after the attack the straight edge would be no more straight and the bending effect would be clearly visible. Perhaps a Gaussian smoothing attack may delete the keypoint as well, arguably with a significantly lower impact on quality.

In this section we propose a new SIFT keypoint removal scheme, referred to as Classification-Based Attack (briefly `CLBA`[1]), whose strategy is based on two hypotheses, supported by a preliminary experimental analysis [Amerini et al., 2013b]. The first hypothesis is that it is possible to discriminate between SIFT keypoints according to some of their local properties (in this case, the

[1] From now on all attacks will be indicated with typewriter font.

first order statistics of surrounding regions in the pixel domain). The second hypothesis is that each class of keypoints reacts differently to counter-forensic attacks. Consequently, one may devise attacks specifically tailored to a class, with reduced impact on the counterfeited image's quality.

10.3.1 General properties

According to the formalisation of Chapter 8, the `CLBA` is a *universal*, *post-processing* attack. It is universal, since it allows to remove SIFT features, thus potentially misleading any (forensic) analysis built on top of them. It is post-processing, since first the image is generated and then traces of all of its keypoints (or a subset of them) are removed. As we will explain in the following, our attack leverages on specific weaknesses of the strategy used by SIFT to refine the initial pool of DoG extrema, but also on common processing, like Gaussian blur. The properties of `CLBA` may however vary to fit the specific SIFT-based algorithm to fool. In this sense, then, the attack can be remodeled in such a way to become *integrated*[2] or *targeted*.[3]

10.3.2 General `CLBA` framework

Before describing each stage of the attack, it is useful to introduce its general structure (see Fig. 10.1).

The proposed scheme relies on an iterative procedure. At the beginning (1^{st} iteration), a grayscale image $I \in \mathscr{I}$ is fed to the system, which starts by detecting the vector of SIFT keypoints $\mathbf{kp}$. Then, for each keypoint $\mathbf{kp}(i) = \{x_i, y_i, \sigma_i, \theta_i\}$, its $N \times N$ neighbourhood is extracted as follows:

$$R_{\mathbf{kp}(i)} = I(x_i + u, y_i + u),\ u \in [-N/2, N/2]. \tag{10.1}$$

The neighbourhood is subsequently assigned to a class $\mathcal{C}_{\mathbf{kp}(i)}$ according to certain properties of its first order statistics (i.e. grayscale histogram). Depending on the assigned class, each keypoint is attacked by means of a tailored

[2] For example, in Chapter 11 it will be shown that a forensically undetectable copy-move forgery can be created by removing SIFT keypoints while counterfeiting the image.

[3] Some SIFT-based forensic detectors (e.g. copy-move) may require to remove only a certain amount of matching keypoints in specific locations, rather than all of them.

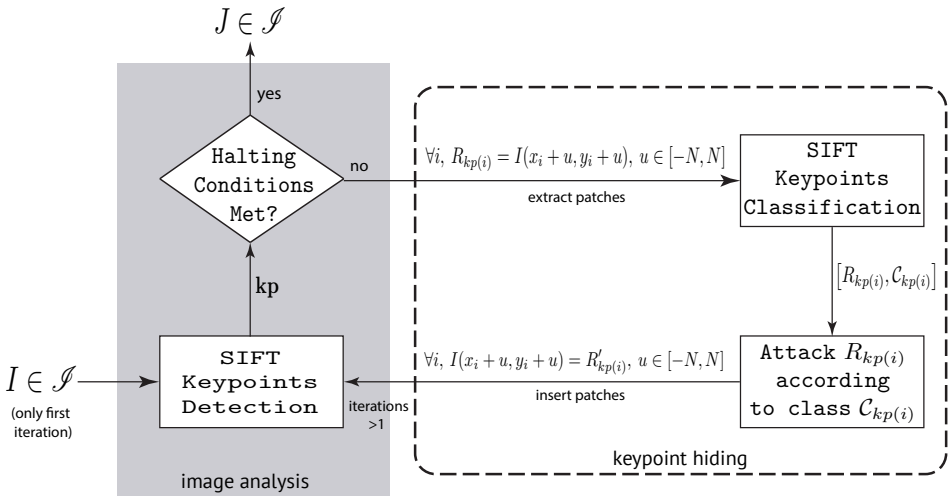


Figure 10.1: *Schematic overview of the Classification-based attack.*

procedure carried out on the $M \times M$ patch[4] centred on it ($M < N$). Finally, the manipulated neighbourhood $R'_{\mathbf{kp}(i)}$ is re-inserted in its original position:

$$I(x_i + u, y_i + u) = R'_{\mathbf{kp}(i)},\ u \in [-N/2, N/2]. \tag{10.2}$$

Once all first-iteration keypoints have been attacked, the procedure is iterated on the manipulated image. The iterations halt when particular requirements are met (e.g. percentage of removed keypoints, maximum number of iterations, minimum allowed perceptual quality), thus generating an image $J = \texttt{CLBA}(I)$. Two main reasons suggest the use of an iterative procedure:

- some keypoints are easier to remove than others. For the sake of visual quality of J, the former are attacked with less strength than the latter. Therefore, an iterative approach naturally allows to intensify the strength of the attack as more robust keypoints keep surviving;
- changes in pixel values may generate a new keypoint in the proximity of the one that is being removed. This phenomenon occurs in the neigh-

[4] For the sake of classification's accuracy, N must be sufficiently large to include significant yet local information. For the sake of imperceptibility, M must be as small as possible. Typical values, experimentally justified in the following, are $N = 32, M = 8$.

bourhoods of those candidate keypoints that were discarded because barely below the SIFT thresholds of Eqs. (9.7)–(9.10). `CLBA` does not keep track of this different category of keypoints but rather deals with them in the same way as the original ones: new keypoints introduced during an iteration *iter* are classified and attacked again at the following iteration $iter + 1$.

Although we will elaborate on the concepts that follow later, some observations are in order here. Despite J and I may look similar by visual inspection, the absence of SIFT keypoints in J, especially in those textured regions where they should be found, could be interpreted as a footprint of `CLBA`, as we actually do in Chapter 12. Obviously, the perceptibility of this effect depends on the number of removed keypoints: if we delete only a few out of the thousands that are usually found in natural images (as in Chapter 11), the manipulation is difficult to identify. Nevertheless, it may be useful to falsify SIFT features, i.e. to introduce fake but plausible keypoints into a previously attacked image (see Chapter 13). By doing so, we can obtain a new image J', still visually similar to both I and J but without plain traces of removal. Fig. 10.2 depicts the complete scenario. In practice, the adversary can opt for a single stage or both depending on the application.

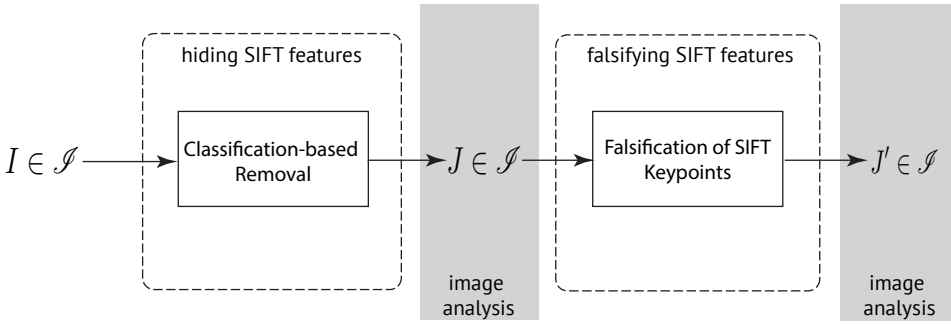


Figure 10.2: *Removal of authentic keypoints and introduction of false ones.*

10.3.3 SIFT keypoints classification

The class to which we assign each SIFT keypoint depends on the visual content of a relatively small neighbourhood around the keypoint in the pixel domain. We have chosen a classification criterion based on the grayscale histogram properties of the keypoint's neighbourhood and more specifically on the number of modes, since it provides valuable information about the local image content. It is worth noting that there exist multiple ways to discriminate among different visual content, including analysing textures or shapes. We opted for the histogram because it is computationally light and does not require learning stages or parametric models. This fact becomes extremely important when dealing with images with tens of hundreds of keypoints, each of which needs to be classified for several iterations of the attack.

We carried out the classification with a modified version of the algorithm proposed in [Chang et al., 2002], which was originally designed for image segmentation based on histogram thresholding.

(A) The original classification algorithm

The original algorithm relies on the assumption that the histogram H of large natural grayscale images can be modeled as a mixture of Gaussians f:

$$f(k) = \sum_{i=1}^{n+1} \frac{P_i}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}\left(\frac{k-m_i}{\sigma_i}\right)^2} \tag{10.3}$$

where: $k = 1 \dots 256$ are the samples of the mixture; $n+1$ is the number of histogram segments; and $(P_i,\, m_i,\, \sigma_i^2)$ are respectively the weight, the mean and the variance of the i-th Gaussian.

Chang *et al.*'s classifier is designed to estimate the model parameters minimising $|f - H|$. In a nutshell, it proceeds as follows.

1. H is Gaussian blurred to reduce the number of unstable local extrema, thus producing $\widetilde{H}$.

2. $\widetilde{H}$ is roughly segmented into approximated clusters C_i, each of which is enclosed between two consecutive local minima.

3. For each C_i, the calculation of (P_i, m_i, σ_i^2) consists of two steps:

3.1. estimating a unique optimal window w^* with minimum skewness, allowing to locate with more precision the center of the segment;[5]

3.2. estimating the Gaussian parameters in w^*, which are then refined by means of a maximum likelihood criterion.

4. The final thresholds used to segment H are computed by relying on the refined Gaussian parameters.

(B) Improved classification algorithm

In order to classify the keypoints, however, we need to know the number of modes n_{modes}, rather than the segments of H. Unfortunately, it is not possible to derive this information directly from the number of Gaussians composing the mixture, since Chang *et al.*'s algorithm tends to over-segment the histograms, thus creating rather flat segments whose weight is very small. Therefore, the original technique required some adjustments to fit the proposed counter-forensic algorithm. We modified the classification as follows:

5. Select the weight of the largest contribution $P_{max} = \max_{1 \leqslant i \leqslant n+1} P_i$.

6. Suppress all contributions $1 \leqslant i \leqslant n+1$ such that $\dfrac{P_i}{P_{max}} \leqslant \alpha$.

7. Set n_{modes} equal to the number of surviving contributions.

We let $\alpha = 0.2$ based on the following experimental procedure. We progressively increased n from 1 to 5 to create five classes C_n consisting of 500 histograms H each, such that $C_n = \{H : n_{modes}(H) = n\}$. Each histogram is generated as a mixture of Gaussians as in Eq. (10.3), with n equal to the desired number of modes and P_i, m_i and σ_i randomly chosen in the intervals $[0.01, 1]$, $[0, 255]$ and $[5, 20]$ respectively. To simulate the histogram of natural images, we perturbed each H with uniformly distributed noise.

We ran the modified classification algorithm on all the classes and we observed its accuracy as a function of α, to which we assigned values in $[0.1, 1]$.

[5] The rationale underlying this step is that skewness quantifies how asymmetric the distribution is. Gaussian distributions are symmetric, hence their skewness is equal to zero.

The metrics we chose are the Precision p_i and Recall r_i for each class:

$$p_i = \frac{Tp_i}{Tp_i + Fp_i} \qquad r_i = \frac{Tp_i}{Tp_i + Fn_i}, \tag{10.4}$$

where Tp_i is the number of histograms of C_i that are correctly classified; Fp_i is the number of histograms of $C_k \neq C_i$ that are erroneously assigned to C_i; and Fn_i is the number of histograms of C_i erroneously assigned to $C_k \neq C_i$. Hence, p_i is the fraction of histograms assigned to C_i that are truly belonging to C_i and r_i is the probability that a histogram is correctly assigned to C_i. Precision and recall can be combined with their harmonic mean, or F-score:

$$F_i = 2 \cdot \frac{p_i \cdot r_i}{p_i + r_i}. \tag{10.5}$$

The closer is F_i to 1, the more accurate is the classifier. In Fig. 10.3 we show the Precision, Recall and F-score averaged over all classes as a function of the threshold α defining the significance of a contribution to the Gaussian mixture of Eq. (10.3). We can observe that the three curves have a maximum in $\alpha = 0.2$, which is the value we assigned to the parameter.

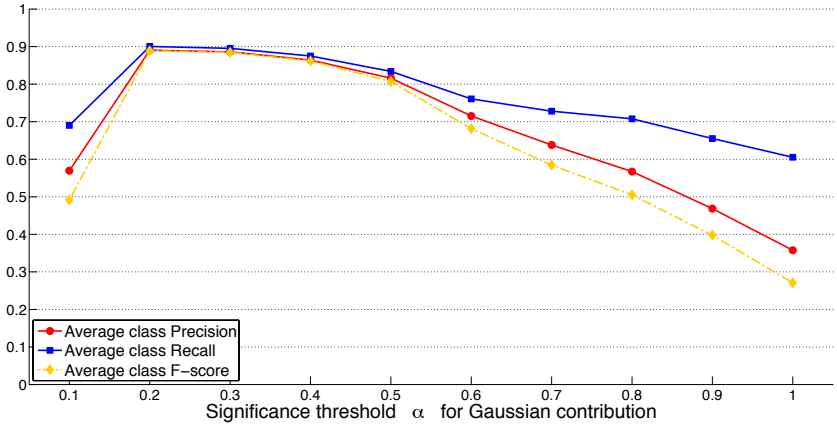


Figure 10.3: *Precision, Recall and F-score as a function of* α.

(C) Classes of keypoints

By relying on the above modified method, we classified the $N \times N$ neighbourhoods of about $120,000$ keypoints extracted from natural images with different

visual content (landscapes, people, buildings).[6] The choice of N is closely related to Chang *et al.*'s algorithm: N should be large enough in such a way that the hypothesis of Gaussianity holds. At the same time, it should be small enough to guarantee the locality of the approach. We observed experimentally that a good trade-off is obtained by letting $N = 32$.

Furthermore, experimental analysis confirmed the hypothesis that histograms tend to cluster into distinct groups. In particular, three of such groups are clearly discernible according to the modality: *unimodal*, *bimodal* and *multimodal*. The three classes are defined as follows:

$$\begin{aligned}
\mathcal{C}_1 &= \{z \in \mathbf{kp}, z = (x, y, \sigma, \theta_{max}) \ : \ n_{modes}\left(R_{\mathbf{kp}(z)}\right) = 1\} \\
\mathcal{C}_2 &= \{z \in \mathbf{kp}, z = (x, y, \sigma, \theta_{max}) \ : \ n_{modes}\left(R_{\mathbf{kp}(z)}\right) = 2\} \\
\mathcal{C}_3 &= \{z \in \mathbf{kp}, z = (x, y, \sigma, \theta_{max}) \ : \ n_{modes}\left(R_{\mathbf{kp}(z)}\right) > 2\},
\end{aligned} \tag{10.6}$$

where $\mathbf{kp}$ is the array of all keypoints and $R_{\mathbf{kp}(z)}$ is the neighbourhood of each keypoint.

Interestingly, these classes correspond to very different visual contents: uniform flat regions with low variance tend to have a unimodal histogram; edges and geometric shapes correspond to bimodal histograms; regions with high variance (resembling some sort of "noise") usually have a multimodal histogram. Fig. 10.4 provides an example of a keypoint belonging to each of the three classes.

10.3.4 Class-tailored single attacks

The second stage of `CLBA` consists of a set of class-tailored attacks, two of which are variants of techniques already known. In the following, with the term *support* we will indicate the $M \times M$ patch centred on the keypoint that is effectively manipulated by an attack. Such patch should not be confused with the $N \times N$ neighbourhood (centred on the keypoint and containing the support) that was used for the classification task.

[6] The data set can be downloaded at `http://homepages.lboro.ac.uk/~cogs/datasets/ucid/ucid.html` (see Sec. 10.4.1 for more information).

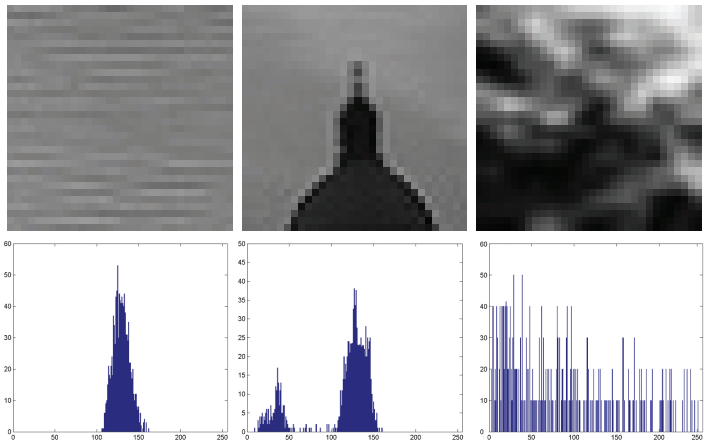


Figure 10.4: *Example of visual contents (first row) and histograms (second row) for the 3 classes of keypoints. From left to right: unimodal (water surface), bimodal (top of a building) and multimodal (foliage).*

(A) Smoothing attack

The `Smoothing` attack is a Gaussian blur flattening the local pixel values in such a way that the contrast value of keypoints slightly above the threshold of Eq. (9.7) is brought below it. The problem with this attack is that an excessive blur has a noticeable impact on the perceptual quality and, most importantly, it may even conjure against removal by introducing new keypoints.[7] It is then important to tune the attack parameters so that a good trade-off between removed keypoints, introduced keypoints and perceptual quality is ensured.

In [Do et al., 2010a], the authors considered satisfactory the results obtained by letting the size of the filtering window h_f and the standard deviation of the Gaussian kernel σ_f be respectively 3 and 1.3. We tuned our parameters according to the following experiment: first, we collected 50 images and we attacked the neighbourhoods of their keypoints with the `Smoothing` attack of increasing σ_f for a total of 30 iterations (the rationale behind the iterative approach is the same of Sec. 10.3.2); then, we measured the percentage of removed keypoints of each manipulated image with respect to the original

[7] We will see in Chapter 13 that this side effect of the `Smoothing` attack can be controlled by the adversary to create fake keypoints further misleading SIFT-based analyses.

amount. Fig. 10.5 shows the average keypoint removal depending on the standard deviation of the filter; the trend plateaus for $\sigma_f \geqslant 0.9$, suggesting that no further gain can be obtained with higher values. Furthermore, increasing σ_f damages the image quality, since up to 6 dB of PSNR (averaged over all the attacked neighbourhoods) are lost. Concerning the filter window size, we did not observe a significant dependence of the outcome of removal on h_f. Consequently, we set the `Smoothing` parameters to $h_f = 3$ and $\sigma_f = 0.9$.

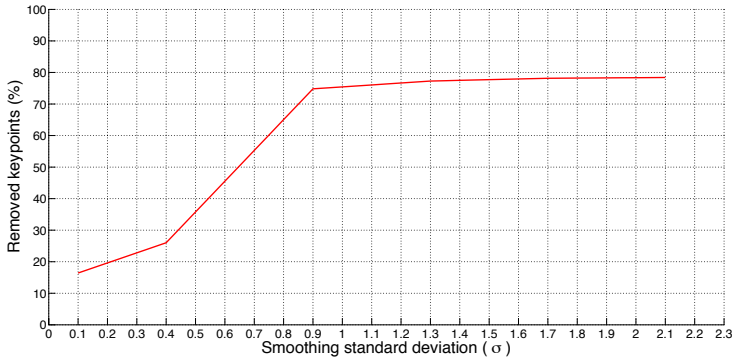


Figure 10.5: *Parameter tuning of the* `Smoothing` *attack (removal): percentage of removed keypoints as a function of standard deviation* σ.

(B) Collage attack

The `Collage` attack is a variant of the method used in [Hsu et al., 2009].[8] In general, it consists in the substitution of an authentic image patch with another patch of the same size but with different properties. In our case, the new patch should obviously not contain SIFT features and should be as similar as possible to the original one according to a similarity criterion.

`CLBA` evaluates similarity by means of the Histogram Intersection Distance [Swain and Ballard, 1991]. Let H and Q be two histograms of L bins; their

[8] The idea behind the Collage attack has been derived from Digital Image Watermarking [Fridrich et al., 2000].

intersection distance $D_{intersect}$ is evaluated as follows:

$$D_{intersect}(H, Q) = \frac{\sum_{u=1}^{L} \min\left(H(u), Q(u)\right)}{\sum_{u=1}^{L} Q(u)}. \tag{10.7}$$

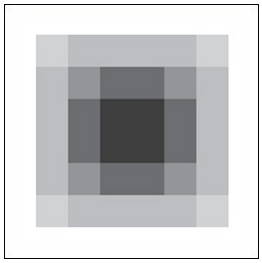
The numerator of Eq. (10.7) is the intersection of the histograms, that is the number of pixels of Q having corresponding pixels of same value in H. In order to obtain a metric in the interval $[0, 1]$, the numerator is normalised according to the number of elements of Q.

In terms of `Collage` attack, given a keypoint i, H corresponds to the histogram of the initial patch $R_{\mathbf{kp}(i)}$ containing SIFT features. Conversely, Q corresponds to the histogram of a patch not containing SIFT features. We stored a large amount of such patches in a previously created database (more details will be provided in Sec. 10.4).

Let R_{min} be the patch of the database that is most similar to the initial one (hence, whose histogram is at minimum $D_{intersect}$); to avoid visible artefacts along the borders, R_{min} is not substituted directly into the source image. Instead, we employ the following linear combination:

$$R'_{kp(i)} = W \cdot R_{\mathbf{kp}(i)} + (1 - W) \cdot R_{min}. \tag{10.8}$$

The window W is an empirical $M \times M$ weighting matrix whose elements $w_{i,j} \in [0, 1]$ are set to 1 along the patch borders and progressively decrease to 0 near the center. Fig. 10.6 shows the weighting window corresponding to $M = 8$.



$$W_{8\times8} = \begin{pmatrix} 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 \\ 1.0 & 0.8 & 0.7 & 0.7 & 0.7 & 0.7 & 0.8 & 1.0 \\ 1.0 & 0.7 & 0.5 & 0.3 & 0.3 & 0.5 & 0.7 & 1.0 \\ 1.0 & 0.7 & 0.3 & 0.1 & 0.1 & 0.3 & 0.7 & 1.0 \\ 1.0 & 0.7 & 0.3 & 0.1 & 0.1 & 0.3 & 0.7 & 1.0 \\ 1.0 & 0.7 & 0.5 & 0.3 & 0.3 & 0.5 & 0.7 & 1.0 \\ 1.0 & 0.8 & 0.7 & 0.7 & 0.7 & 0.7 & 0.8 & 1.0 \\ 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 \end{pmatrix}$$

Figure 10.6: *Weighting window of size* 8×8 *of* Eq. (10.8) *and its coefficients.*

(C) Removal with Minimum Distortion attack

The Removal with Minimum Distortion (`RMD`) attack is the same method in [Do et al., 2010a]. The idea behind it is to calculate a small patch ϵ that, added to the neighbourhood of a keypoint, allows its removal. The coefficients of ϵ are chosen in such a way that the contrast around the keypoint (at DoG level) is reduced, thus invalidating the corresponding SIFT check. In a nutshell, it works as follows. Let $\mathbf{x} = (x, y, \sigma)$ be a keypoint and let $D(\mathbf{x})$ be the DoG in $\mathbf{x}$; patch ϵ is derived from the optimisation problem:

$$\epsilon = \underset{\epsilon: D'(\mathbf{x}) = D(\mathbf{x}) + \delta}{\operatorname{argmin}} \frac{1}{2}||\epsilon||^2, \tag{10.9}$$

where δ is a parameter that controls the intensity of the attack. SIFT contrast is reduced by $|\delta|$ in such a way that the altered value $D'(\mathbf{x})$ drops below its threshold. Since the size of ϵ depends on the scale of the targeted keypoint, the final altered DoG region is $D(x+u, y+v, \sigma)$, with $(u, v) \in [-6\sqrt{2}h\sigma, 6\sqrt{2}h\sigma]$ and $h = 2^{1/3}$.

With respect to the original scheme by Do *et al.*, we introduced two small variations in compliance with the other attacks: first, we limited the size of ϵ to a maximum of $M \times M$ also for those keypoints whose spatial support would be normally larger; secondly, we used the same weighting window of Eq. (10.8) to replace the initial neighbourhood:

$$R'_{kp(i)} = W \cdot R_{\mathbf{kp}(i)} + (1 - W) \cdot \epsilon. \tag{10.10}$$

10.3.5 `CLBA`'s composition of single attacks

The class-unaware attacks of the previous section are arranged into `CLBA`'s iterative procedure, which repeatedly classifies and attacks keypoints until one of the following conditions is verified: the maximum number of allowed iterations (maxIter) is reached; or a certain percentage of keypoints has been removed (minRemoval).

Iterations are divided in two blocks: $[1, K]$ and $[K{+}1$,maxIter$]$. The first block deals with *weaker* keypoints, while the second block deals with the remaining *stronger* keypoints. The attacks associated to each block are shown in Tab. 10.1, while the pseudo-code of `CLBA` is provided in Algorithm 1.
The rationale behind the above choice is explained in the following.

	Iteration block	
$\mathcal{C}_{i=\{1,2,3\}}$	$iter \in [1, K]$	$iter \in [K+1, \text{maxIter}]$
unimodal	`Smoothing`	`Collage`
bimodal	`Smoothing`	`RMD`
multimodal	`Smoothing`	`Collage`

Table 10.1: *Attacks assigned to each class depending on iterations.*

- `Smoothing` reduces the population of less robust keypoints, regardless of their content and class, without a significant loss of quality.
- `Collage` is best suited to those patches whose content can be substituted in a fairly imperceptible way, hence uniform and noisy patches. Conversely, it is not suitable for patches containing geometric edges, because histogram similarity does not take into account shapes.
- `RMD` is a very powerful attack that does not suffer of `Collage` drawback and thus is used on the remaining class of bimodal keypoints.

10.4 Experimental validation

In this section, we evaluate experimentally `CLBA`'s performance. More specifically, we demonstrate the following two facts:

- effectiveness of `CLBA`: the classification stage followed by class-tailored attacks outperforms class-unaware attacks;
- robustness of `CLBA`: performance does not dependent significantly on the specific SIFT implementation at hand.

The performance in successfully removing keypoints is evaluated by means of the Keypoint Removal Rate (KRR) metric, which is defined as follows:

$$KRR = \left(1 - \frac{keypoints\ detected\ after\ attack}{keypoints\ detected\ before\ attack}\right) \times 100. \tag{10.11}$$

Algorithm 1: *Pseudo-code of the Classification-based attack.*

```
function J = CLBA(I, minRemoval, maxIter)
    input  : Authentic image I, target removal rate,
             maximum iterations
    output: Manipulated image J without keypoints
    iter ← 1;
    K ← 10;
    J ← I;
    removal_rate ← 0;
    while (iter ⩽ maxIter and removal_rate < minRemoval) do
        keypoints = calculateSIFT(J);
        [C1, C2, C3] = classifySIFT(keypoints);
        foreach keypoint in keypoints do
            if (iter ⩽ K) then
                J ← smoothingAttack(keypoint);
            else
                if (C_keypoint == C2) then
                    J ← RmdAttack(keypoint);
                else
                    J ← collageAttack(keypoint);
                end
            end
        end
        removal_rate ← calculate_removal_rate(I, J);
        iter ← iter + 1;
    end
```

The visual quality of the manipulated image is measured with PSNR (Peak Signal-to-Noise Ration) and SSIM (Structural SIMilarity) index [Wang et al., 2004]. For the sake of clarity, we make two assumptions. The first one is to work on grayscale images, which has no consequence on the generality of the approach, since SIFT normally ignores color information. The second one

is to limit the analysis to keypoints of the first octave, corresponding to the image at its original size ($octave = 0$). We will relax the latter assumption at the end of this section, where we will show that CLBA's trend on other octaves (i.e. $\{-1, 1, 2\}$) is comparable to that of the first one.

Finally, unless otherwise specified, the experiments are based on the VLFeat SIFT implementation of [Vedaldi and Fulkerson, 2010]. To obtain descriptors that are as close as possible to the original implementation by Lowe,[9] we set SIFT thresholds as follows: $edge\_threshold = 10$, $peak\_threshold = 4$.

10.4.1 Image data set

The experimental analysis has been carried out on the UCID image database [Schaefer and Stich, 2003], a well-known benchmark in image retrieval research community.[10] It consists of $1,338$ uncompressed (TIFF) color images, whose size is either 384×512 or 512×384 pixels and whose contents depict landscapes, cityscapes, people and man-made objects. The rather large size of this collection allows us to make conclusive statements on the performance of CLBA, while the relatively small size of the images provides a significant yet not excessive amount of keypoints to deal with. The first-octave keypoints are distributed as shown in the histogram of Fig. 10.7.

In addition, UCID images are also used to create the database for the Collage attack of Sec. 10.3.4, consisting of $150,000$ patches not containing SIFT features.

10.4.2 Effectiveness of CLBA

The comparison between CLBA and class-unaware attacks in terms of removal effectiveness and quality of the manipulation proceeded as follows.

1. Classification and attack supports are set to 32×32 and 8×8 respectively; target removal rate is set to 100%; the iteration switch K and the maximum number of iterations are set to 10 and 40 respectively.[11]

[9] Compiled binaries can be downloaded at: http://www.cs.ubc.ca/~lowe/keypoints/.

[10] The UCID data set can be freely downloaded at: http://homepages.lboro.ac.uk/~cogs/datasets/ucid/ucid.html.

[11] The choice of such values for the parameters K and maxIter, justified by data of Fig. 10.9, ensures an acceptable trade-off between removal and quality.

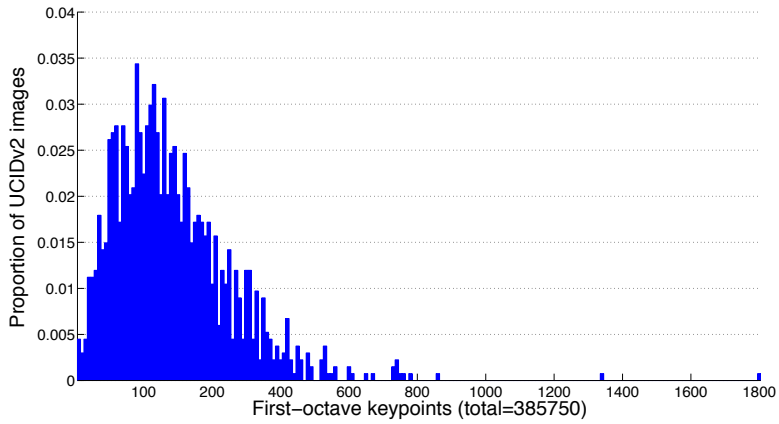


Figure 10.7: *Distribution of first-octave keypoints (UCID).*

2. Each image is attacked by means of four methods: i) `CLBA`; ii) iteration of `RMD` ($\delta = 1$); iii) iteration of `Smoothing` ($\sigma_f = 0.9$, $h_f = 3$); iv) iteration of `Collage`. Each attack is halted when either 100% removal rate or the 40^{th} iteration is reached.

3. When the algorithms halt, the *KRR*s achieved on each image are evaluated and organised in four histograms.

The envelopes of such histograms are shown in Fig. 10.8 (only for $KRR \geqslant 50\%$, since they are null otherwise). The `CLBA` provides the highest removal rates, while granting a minimum removal rate of 80% practically on the whole data set. As an example, consider $KRR \geqslant 90\%$: `CLBA` achieves such goal on $1,149$ images out of $1,338$ (86% of data set), followed by `Collage`, which achieves the same results on 468 images (35% of the data set). The `RMD` and `Smoothing` are the less effective, with 147 (11% of data set) and 12 (0.9% of data set) respectively. It is also worth noting that only `CLBA` reaches perfect removal, although only on a limited number of images (32, i.e. 2.7% of the data set). Fig. 10.9 shows the *KRR* trend for each attack depending on iterations sampled with step 5. The barely noticeable improvement in removal past the 40^{th} iteration does not justify the assignment of larger values.

The number of deleted keypoints is not the only important metric for evaluating the performance of the attacks. To be really effective, an attack also

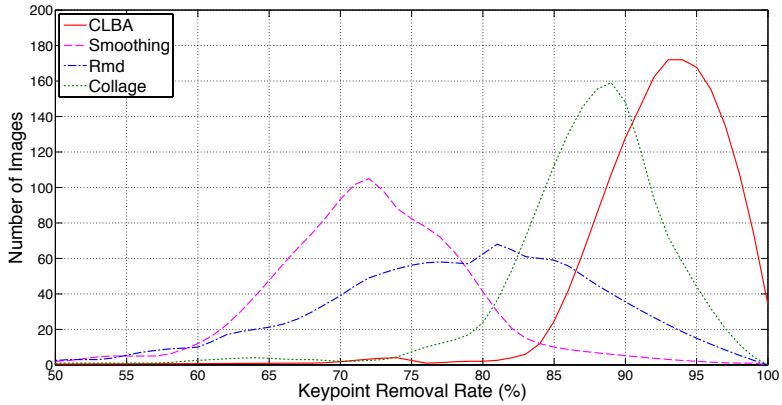


Figure 10.8: *Effectiveness of* `CLBA` *with respect to class-unaware techniques (UCID data set).*

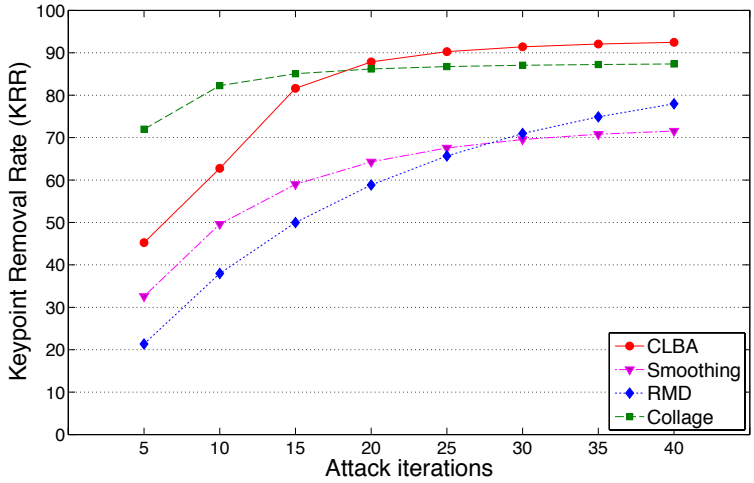


Figure 10.9: *Keypoint Removal Rate versus number of iterations for all the removal attacks.*

needs to preserve the image quality. Therefore, to assess the impact on quality, the images belonging to each bin of the histograms of Fig. 10.8 are selected and their average PSNR over all the attacked patches is computed. The results for high removal rates are summarised in Tab. 10.2, where results corresponding

to the images for which $KRR \geqslant 90\%$ are omitted because averaging on a small number of images does not produce significant results.

	Keypoint Removal Rate						
Attack	**60%**	**65%**	**70%**	**75%**	**80%**	**85%**	**90%**
`CLBA`	36.66	36.6	36.59	36.57	36.56	36.55	36.4
`Smoothing`	41.83	41.79	41.87	41.98	41.85	41.28	40.55
`RMD`	29.57	29.46	29.31	29	28.77	28.4	27.99
`Collage`	30.34	30.31	30.27	30.24	30.12	29.87	29.71

Table 10.2: *Average patch PSNR (dB) versus removal rate for the* 4 *attacks.*

Expectedly, the `Smoothing` attack has the lowest impact on image quality, but such an advantage comes at the price of the lowest removal rates. Among the remaining techniques, `CLBA` provides the best quality. One may wonder about the causes behind the poor performance of `Collage` and `RMD`: for the former, they may be related to the size or the quality of the database; as for the latter, they are undoubtedly related to the nature of the attack itself. `RMD`, in fact, although very powerful, covers the original patches with very unpleasant "dots" rather than replacing them with something more similar content-wise. Consequently, this effect quickly deteriorates the local quality, especially for those keypoints whose spatial support is large.

Fig. 10.10 provides a visual comparison of the three most effective methods. The artefacts introduced by `RMD` (bottom left) and `Collage` (bottom right) are more noticeable than those inserted by `CLBA` (top right). Such phenomena are particularly visible between the ears of the dog. Additional examples of both authentic and `CLBA`-forged images are provided in the section closing this chapter.

Figure 10.10: *Detail of an attacked region. From top left to bottom right: Authentic;* `CLBA` *(KRR =* 98%*, average patch PSNR of* 37.8 *dB);* `RMD` *(*94%*,* 25.6 *dB);* `Collage` *(*91%*,* 35.1 *dB).*

10.4.3 Robustness to SIFT implementations

There exist a number of different implementations of the SIFT algorithm, often have different outcome in terms of number or spatial location of keypoints. It is reasonable to assume that the adversary does not necessarily know which implementation the forensic analyst is going to use. Therefore, the effectiveness of the attack should not be influenced by the use of a particular software.

(A) Chosen implementations

To verify whether `CLBA` can cope with different SIFT implementations, we gathered four software:

- *VLFeat*[12] [Vedaldi and Fulkerson, 2010]. This is the "reference library" for SIFT-based applications. Other than the work presented here, [Amerini et al., 2011], [Do et al., 2010b], [Do et al., 2010a] and [Do et al., 2012] are also based on it.

[12] Freely available for download at: `http://www.vlfeat.org`.

- *Sift Legacy*[13] (also known as Matlab/C and SiftC++) [Vedaldi, 2007]. Although this is basically the predecessor of VLFeat and has been superseded by it, such software is still used (see [Caldelli et al., 2012]).
- *Rob Hess Sift Library*[14] [Hess, 2010]. Another popular implementation, due to its dependence on the well-established OpenCV Computer Vision library.
- *JIFT* (by Jun Liu).[15] Arguably the less famous implementation among the ones we used in this work, it makes use of the VXL Computer Vision library.

In order to obtain fair results, we did not tweak the attacks to the characteristics of the various detectors: the only constraint we imposed to the tools is to work on the first octave, in compliance with our initial assumption. We left all other parameters unchanged (see Tab. 10.3), with their values corresponding most of the times to those suggested in [Lowe, 2004].

	Octaves			**Thresholds**		
	Total	**Initial**	**Intervals**	**Contrast**	**Edge**	**Peak**
VLFeat	1	0	3	×	10	4
SiftLegacy	1	0	3	0.03	10	4
RobHess	1	0	3	0.04	10	0.8
JIFT	1	0	3	0.03	10	0.8

Table 10.3: *Main parameters of the employed SIFT implementations. VLFeat controls contrast by means of Peak threshold (hence the symbol ×).*

(B) Evaluation procedure

For each image, the test proceeded as follows: i) original keypoints are computed independently with all SIFT implementations; ii) images are manipulated by means of VLFeat-based `CLBA`; iii) *KRR* is evaluated according to the

[13] Freely available for download at: `http://www.vlfeat.org/~vedaldi/code/sift.html`.
[14] Freely available for download at: `http://blogs.oregonstate.edu/hess/code/sift`.
[15] Freely available for download at: `http://www.cs.man.ac.uk/~liuja/#downloads`.

number of final keypoints detected by each version of SIFT. Similarly to the procedure leading to Fig. 10.8, all values are organised into histograms, whose envelopes are shown in Fig. 10.11.

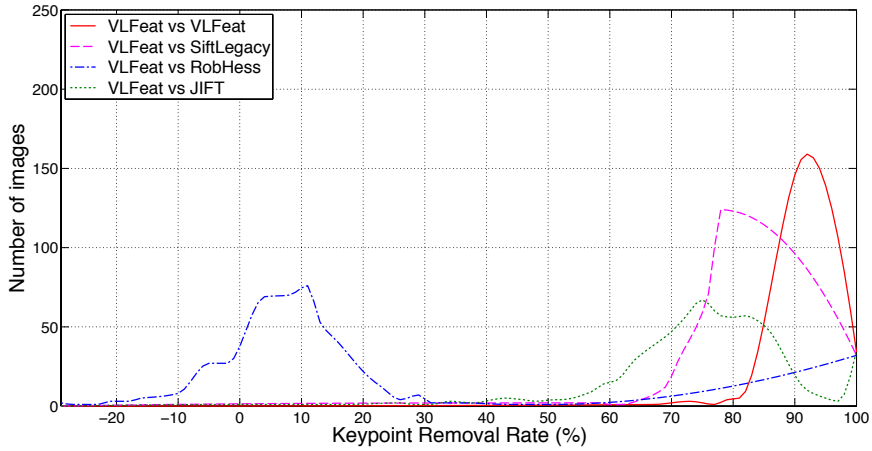


Figure 10.11: *Robustness of the VLFeat-based* `CLBA`*. Curves correspond to histogram envelopes of KRR, obtained by analysing the manipulated UCID data set by means of four different SIFT versions.*

As expected, the best results are achieved against the VLFeat-based detector (average $KRR = 92\%$), followed by SiftLegacy (80%) and JIFT (75%). However, the proposed method does not seem to be effective against RobHess, given that on average only 9.5% of the detected keypoints are removed. Furthermore, on 216 images out of $1,338$ (about 16% of data set) new keypoints are introduced by `CLBA`, thus explaining the negative rates of Fig. 10.11. A more accurate analysis revealed that this specific implementation often tends to calculate several keypoints in spatial locations different from those detected by the remaining tools. Therefore, the `CLBA` did not actually manipulate the neighbourhoods of RobHess keypoints.

This problem can be solved by modifying the base `CLBA` framework in such a way that both VLFeat and RobHess are used. This time, the union of keypoints coming from the two detectors is considered. The curves of Fig. 10.12 are obtained by following the same procedure of the previous case.

Not only the attack is now dramatically more effective against the RobHess

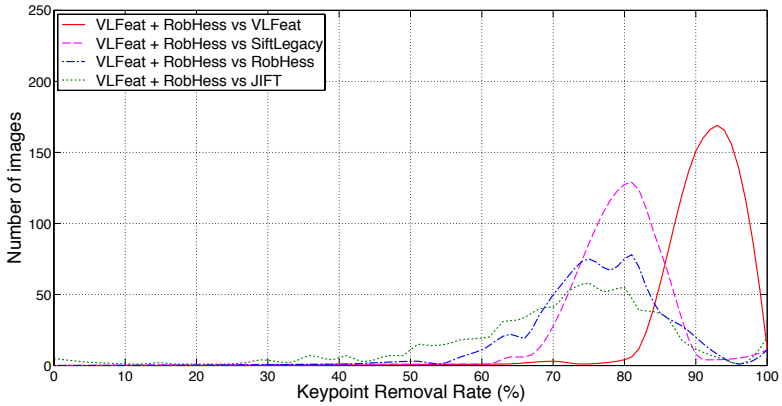


Figure 10.12: *Robustness of the VLFeat+RobHess-based* `CLBA`.

version ($KRR = 76.7\%$), but the performance remains basically the same against VLFeat (91.5%), SiftLegacy (79.8%) and JIFT (71%).

(C) A visual example

As an example, consider the test image shown in Fig. 10.14 (left). When applied to the authentic image, VLFeat and RobHess detect respectively 58 and 24 keypoints.

The results of Fig. 10.13 (top) are obtained by relying on VLFeat only; for each iteration (x-axis), we show the number of keypoints (total and for each class) that have been left into the image. We omit the iterations past 25 since there are no more variations in the number of remaining keypoints. According to VLFeat, there is only 1 keypoint left in the attacked image. However, if the same image is analysed with RobHess, 15 keypoints are still detected. Conversely, when the attack is carried out again by using both VLFeat and RobHess inside `CLBA`, the results of Fig. 10.13 (bottom) are obtained. Clearly, the amount of keypoints is higher now (82), as it corresponds to the union of the keypoints detected by the two implementations. However, the attack shows the same trend as before and, more importantly, now it is also effective against RobHess.

It is important to point out that such an improvement does not come at the cost of perceptibility. As Fig. 10.14 can confirm, loss of quality of

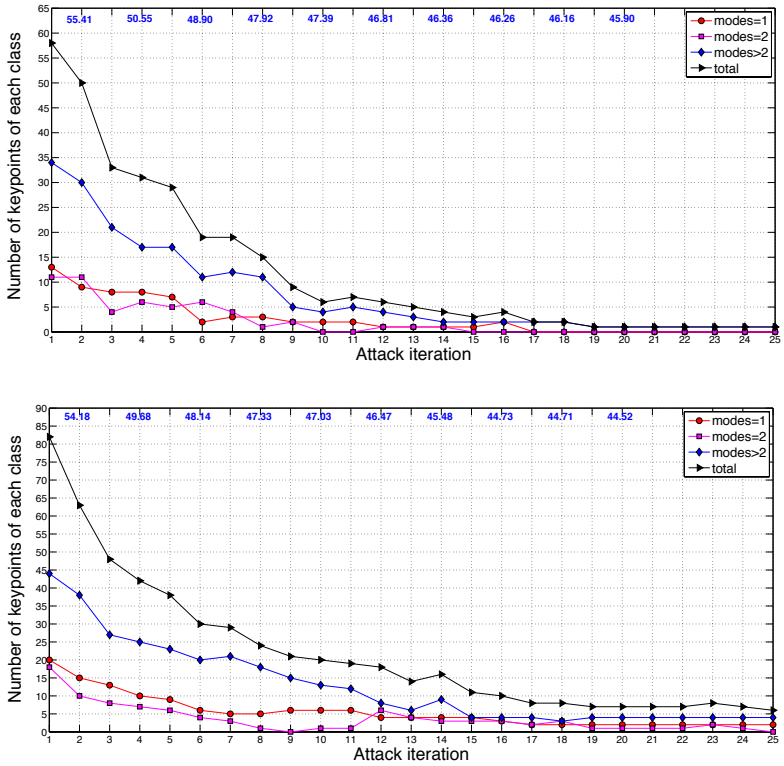


Figure 10.13: *Iterations on a test image: number and classes of keypoints. VLFeat alone (top) and VLFeat+RobHess (bottom).*

VLFeat+RobHess (middle) with respect to VLFeat alone (right) is hardly noticeable. In terms of PSNR, 44.1 dB versus 45.8 dB full-frame and 36.6 dB versus 37.7 dB for the average over all patches; in terms of SSIM, 0.971 versus 0.998 full-frame and 0.882 versus 0.891 patch average.

10.4.4 Perceptual quality assessment of `CLBA`

The quality degradation caused by `CLBA` has been evaluated in terms of PSNR and SSIM in Sec. 10.4.2. To better understand the impact of the attack on human perception, we performed two campaigns of subjective tests by resorting to crowdsourcing [Keimel et al., 2012], whereby problems are broadcast to

Figure 10.14: *Perceptibility of* CLBA *on a test image (*40*-th iteration). From left to right: authentic, VLFeat alone and VLFeat+RobHess.*

an unknown group of solvers (referred to as the crowd) in the form of an open call for solutions. In our case, we asked to rate authentic and CLBA-forged images or to discriminate between them without providing a ground truth. Even though some of the results refer to the slightly modified perceptually-driven version of CLBA presented in [Amerini et al., 2014], they still retain their validity for the original version discussed in this thesis.

(A) Subjective experiments

The first experiment, whereby 250 subjects participated to a crowdsourcing-based campaign, was devoted to verify the impact of CLBA on the perceived image quality. After the screening process to remove outliers and incomplete results, a set of 213 subjects were considered. Each subject evaluated 60 different images: 10 original images and 50 modified ones produced by CLBA at different iterations $\{1, 2, 3, 5, 40\}$. The test was performed by using an Absolute Category Rating with Hidden Reference approach (ACR-HR), which means that the images were presented one at a time in random order and were rated independently on a scale from 1 to 5, with 1 denoting poor quality and 5 excellent quality. The obtained results are reported in Tab. 10.4. It can be noticed that the average MOS (Mean Opinion Score) does not change significantly with the number of considered iterations. Moreover, the results show that the subjects were not able to discriminate between originals and CLBA-forged images, thus proving that the attack does not significantly affect the image quality.

Twenty-five subjects took part to the second test, whose purpose was to

		CLBA iterations				
	Authentic	**1**	**2**	**3**	**5**	**40**
MOS	3.502	3.531	3.553	3.521	3.488	3.507

Table 10.4: *Result of the first subjective test: Mean Opinion Score corresponding to a selection of* `CLBA` *iterations.*

assess whether authentic and forged images can be distinguished by means of a pair-wise comparison approach. The stimuli were displayed on a Panasonic BT-3DL2550 screen (1920×1080 pixels). Each subject was asked to choose, for every couple of images displayed on the screen, the one that was modified (i.e. attacked with `CLBA` at iteration $\{1, 2, 3, 5, 40\}$). For each couple, the authentic image and the modified one were randomly displayed on the left and right side of the screen. The results are displayed in Fig. 10.15, where correct detections are represented with respect to the users. It is possible to notice that it is a complicate task for users (also in a controlled environment) to distinguish the original image from the attacked one.

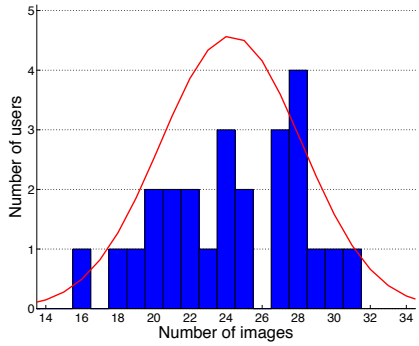


Figure 10.15: *Correct detections with respect to the crowdsourcing users.*

(B) Color comparisons

The last experiment consisted on the re-introduction of color information into the attacked image to evaluate potential differences with respect to the au-

thentic RGB image. Color restoration is carried out in the YCbCr color space first by performing the removal procedure on the luminance channel of the authentic image and then by joining the forged luminance and the authentic color channels Cb and Cr and reverting to the RGB color space. Some examples of this procedure are shown in Fig. 10.16.

(a) **(b)** **(c)** **(d)**

Figure 10.16: *Restoration of color information. (a)–(c):* `CLBA`*-forged images; (b)–(d): authentic images.* $\Delta(a,b) = 1.36$, $\Delta(c,d) = 1.92$.

The color difference ΔE between each attacked image and its authentic version, measured by means of the method in [Rajeev Ramanath et al., 2002] (see Sec. 4.1.2 and Eq. (4.4) in particular) and then averaged over all the UCID images, is 2.14; such value is smaller than the just noticeable threshold empirically set to 2.3 in [Mahy et al., 1994], thus confirming that the quality of the forged images is perceptually satisfactory.

10.4.5 Remarks on the complexity of `CLBA`

All the attacks evaluated so far are fairly lightweight in terms of computational resources. Within a single iteration, the main contribution to time complexity comes from cycling through all keypoints, while detection of SIFT features generally has a negligible impact. In Fig. 10.17, average execution times over the UCID data set are shown for each attack. Even `Collage` attack (triangular marker), whose highest complexity (evident for removal rates $\geqslant 75\%$) is the consequence of several comparisons with the database content, remains feasible (up to five minutes for 95% removal).

The comparisons are still required by CLBA (star marker), but they are limited in terms of iterations (25 instead of 40) and keypoint classes (2 instead of 3). All tests have been performed on MathWorks Matlab on a desktop configuration with a 2 GHz dual-core processor, 4 GB RAM, 32 bit OS.

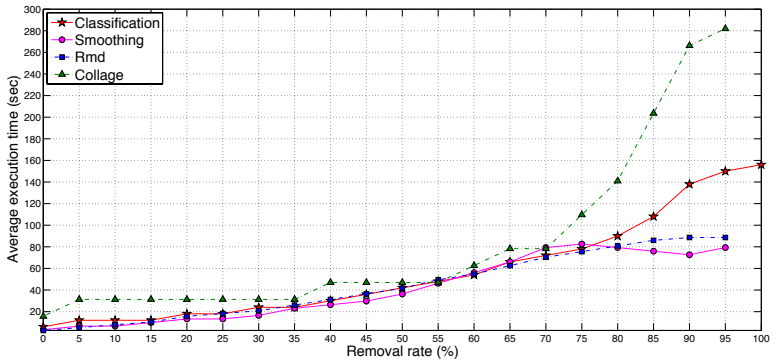


Figure 10.17: *Removal rate versus average processing time (UCID).*

10.5 Towards multi-octave keypoint removal

The tests we carried out so far considered exclusively first-octave keypoints. It is the aim of this section to relax such an assumption and evaluate the behaviour of the Classification-based attack on the remaining octaves $\{-1, 1, 2\}$. Therefore, in the sequel we will use CLBA to delete keypoints originated from images whose size has been upscaled by factor 2 and downscaled by factors 2 and 4, respectively.[16] Tab. 10.5 summarises the total number of keypoints in each octave and their average number in each image.

In order to keep the complexity of tests manageable but without losing generality, we carried out all the following tests on the first 500 images of the UCID data set.

In the first experiment all octaves are attacked by means of CLBA with 8×8 support. The outcome is certainly positive for lower octaves $\{-1, 0\}$, where average removal rates of 99.89% and 93.22% are obtained (see Fig. 10.18 (b)).

[16] To make the comparisons easier, we show again the results already obtained for the first octave.

	Octaves			
	−1	**0**	**1**	**2**
keypoints	1,236,264	388,662	127,811	39,855
average	924	291	96	30

Table 10.5: *Total keypoints per octave and average keypoints per image.*

On the contrary, results are not as satisfactory for higher octaves $\{1, 2\}$, with just 55.19% and 39.73% average removal rates.

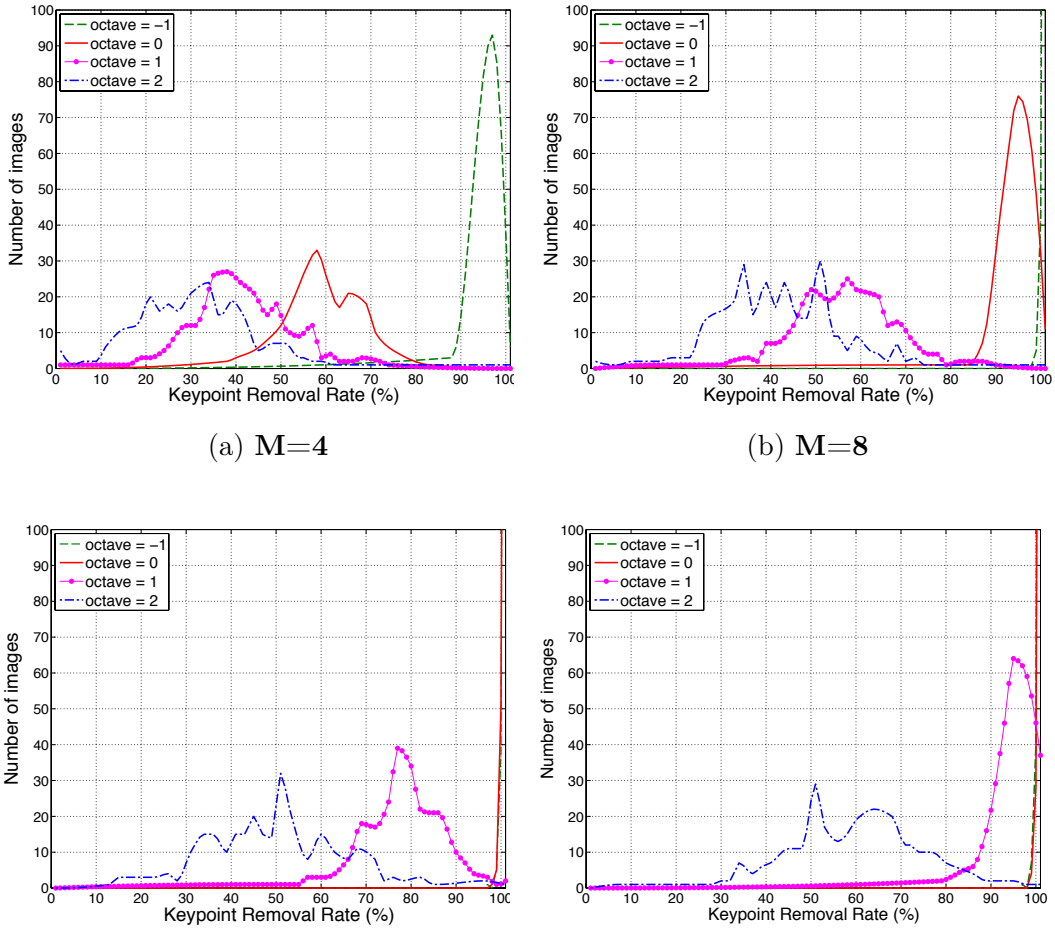
To obtain good results on higher octaves, the support of the attack must be increased (see Fig. 10.18 (c)–(d)). This observation is corroborated by the data of Tab. 10.6, describing the relationship between removal rates, octaves and support size.

Such trends can be explained by noting that `CLBA` is not a pyramidal approach: each keypoint is attacked on the base image, regardless of the octave that generated it. Every time SIFT moves to a higher octave, the keypoint's neighbourhood attacked at first octave is progressively halved. As a consequence, the manipulation tends to be compromised, thus failing to remove the stronger keypoints.

Expectedly, the problem with larger supports is perceptibility. Fig. 10.19 shows an example of the removal artefacts introduced at higher octaves. The image on the left has been attacked at octave 1 with $M = 14$, the one on the right at octave 2 with $M = 20$. The most visible artefacts correspond to the typical "dots" introduced by the `RMD` attack on bimodal keypoints; the side effects become more and more visible as the support grows. While the quality of the leftmost image of Fig. 10.19 is still acceptable even at a close inspection, this is not true for the rightmost image, see for example the bottom right portion in proximity of Pippo's mouth.

Nevertheless, according to Tab. 10.7, the quality of the manipulated images (with respect to *KRR*) remains acceptable especially for octaves $\{-1, 0, 1\}$. Tab. 10.8 concludes the analysis by summarising the average *KRR* for each octave, depending on iterations and on a selection of supports.

The procedure for multi-octave keypoint removal is summarised in Algorithm 2 for octaves $\{-1, 0\}$, but can be easily generalised to the remaining



(a) **M=4** (b) **M=8**

(c) **M=12** (d) **M=16**

Figure 10.18: *Histogram envelopes of KRR for each SIFT octave depending on attack support.* `CLBA`*'s effectiveness is proportional to support's size.*

ones. In practice, keypoints are firstly removed by means of standard `CLBA`, from the higher octave with larger support. Then, support is reduced and the manipulated image is attacked again with `CLBA` until all the octaves have been processed. The parameters better suited to each octave have been derived

Figure 10.19: *Example of removal artefacts at higher octaves (best wieved on display). Left: octave* 1, $M = 14$*; right: octave* 2, $M = 20$.

	Octaves			
	−1	**0**	**1**	**2**
M = 4	94.69	58.28	40.27	29.35
M = 8	99.89	93.22	55.19	39.73
M = 10	99.89	99.15	66	43.98
M = 12	99.9	99.87	77.05	48.78
M = 14	99.9	99.91	86.7	54.49
M = 16	99.88	99.93	93.83	59.68
M = 20	99.93	99.9	99.1	69.82

Table 10.6: *Average KRR depending on octave and on attack support.*

	Octaves			
	−1	**0**	**1**	**2**
M = 4	39.95	42.14	42.87	42.07
M = 8	35.03	40.66	40.86	38.43
M = 10	33.65	35.48	36.54	37.44
M = 12	32.44	34.23	33.08	36.68
M = 14	31.61	32.99	31.61	33.97
M = 16	31	31.98	30.28	29.31
M = 20	30.1	30.12	28.56	27.93

Table 10.7: *Average local PSNR depending on octave and attack support.*

experimentally from Tabs. 10.6–10.8.

Algorithm 2: *Pseudo-code for multi-octave* `CLBA`.

function J = *multiOctave_*`CLBA(`I, minRemoval`)`

input : Authentic image I, target removal rate
output: Manipulated image J without keypoints

octaves $\leftarrow [0, -1]$;
$M \leftarrow [8, 4]$;
maxIter $\leftarrow [40, 25]$;

$J \leftarrow I$;
for $u \leftarrow 1$ **to** *n_octaves* **do**
  $J \leftarrow$ `CLBA` (J, *octaves*(u), $M(u)$, minRemoval, maxIter(u));
end

	Attack iterations							
M = 4	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**
o = −1	61.67	77.32	87.2	91.34	92.97	93.8	94.27	94.69
o = 0	32.28	40.81	49.23	53.66	55.86	57.04	57.67	58.28
o = 1	24.96	29.63	34.7	37.32	38.74	39.45	39.75	40.27
o = 2	20.23	22.64	25.25	27.14	28	28.29	28.49	29.35
M = 8								
o = −1	73.22	91.55	97.1	98.85	99.28	99.47	99.56	99.89
o = 0	46.44	63.91	82.39	88.53	90.81	91.88	92.54	93.22
o = 1	30.2	38.42	47.57	51.51	53.21	54.04	54.45	55.19
o = 2	25.12	29.43	34.45	37.03	38.08	38.52	38.75	39.73
M = 16								
o = −1	77.56	95.87	97.44	98.49	98.99	99.32	99.49	99.88
o = 0	46.89	68.64	92.86	98.39	99.01	99.23	99.39	99.93
o = 1	37.03	50.9	78.04	86.59	89.78	91.49	92.44	93.835
o = 2	28.56	35.63	46.78	53.19	55.92	57.47	58.32	59.68

Table 10.8: *Average KRR per octave depending on iterations and support.*

10.6 Concluding remarks

In this chapter we presented a new tool capable of removing SIFT features. In contrast with the rest of the (still limited) literature, this technique first discriminates between different classes of keypoints and then attacks each class with specifically tailored attacks. Results obtained in this way are superior to class-unaware state-of-the-art attacks in terms of trade-off between keypoint removal effectiveness and perceptual quality of the manipulated image. Several aspects of the proposed method could benefit from further investigation, including: i) exploring the scenario that has emerged from the analysis of Sec. 10.4.3, where more than one implementation of SIFT interact with each other; ii) improving effectiveness of the Classification-based attack on higher octaves, where acceptable removal rates come at the cost of a significant impact on visual quality.

Additional examples of keypoint removal

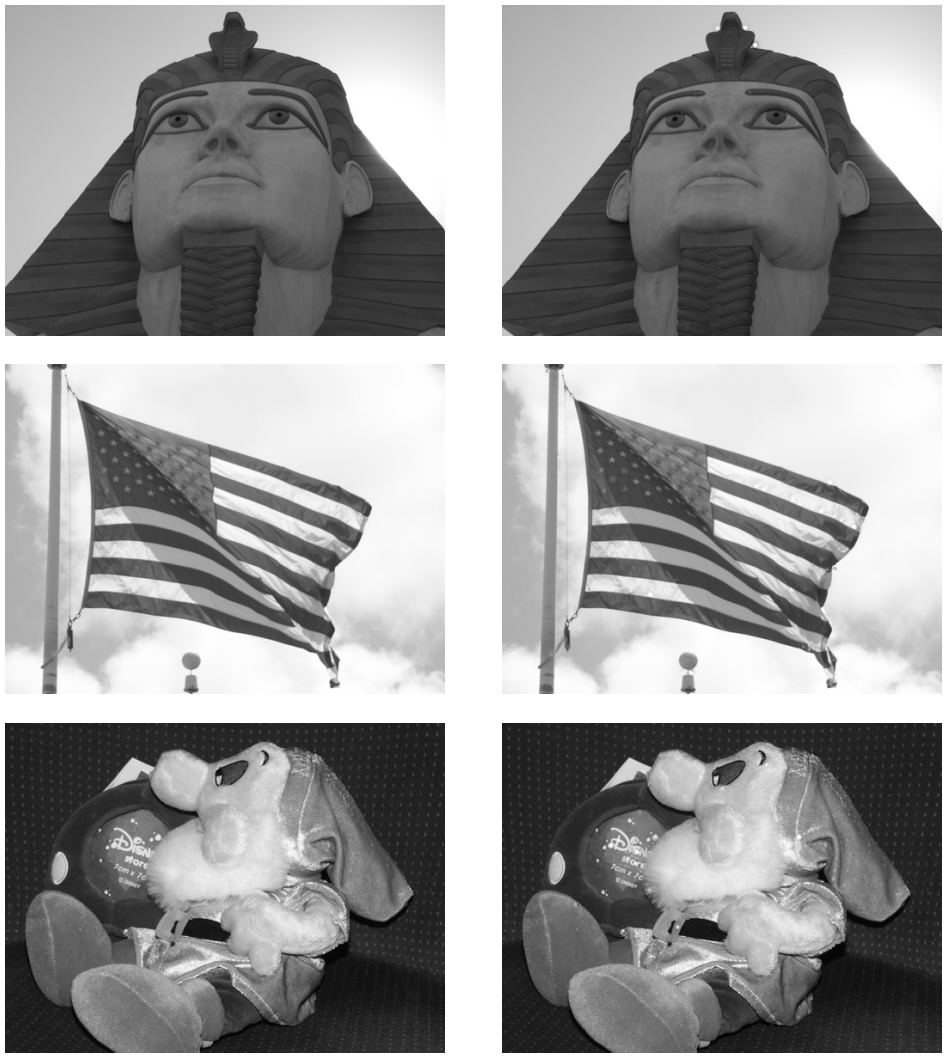


Figure 10.20: *Additional examples of images attacked with* CLBA (i). *Left: authentic; right: forged (octave* $= 0$, $M = 8$, $KRR \geqslant 99\%$ *for all images).*

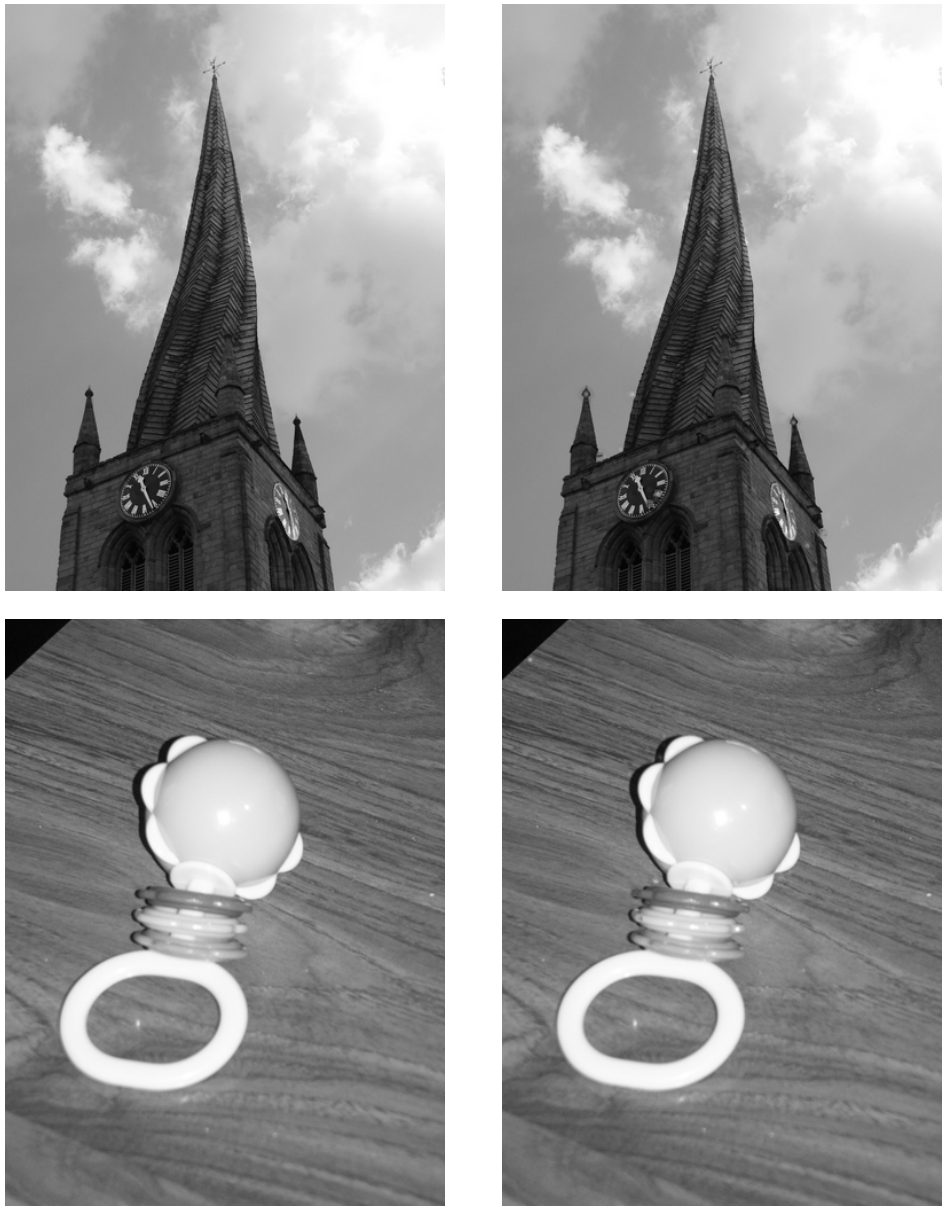


Figure 10.21: *Additional examples of images attacked with* `CLBA` (ii). *Left: authentic; right: forged (octave* $= 0$*,* $M = 8$*,* $KRR \geqslant 99\%$ *for all images).*

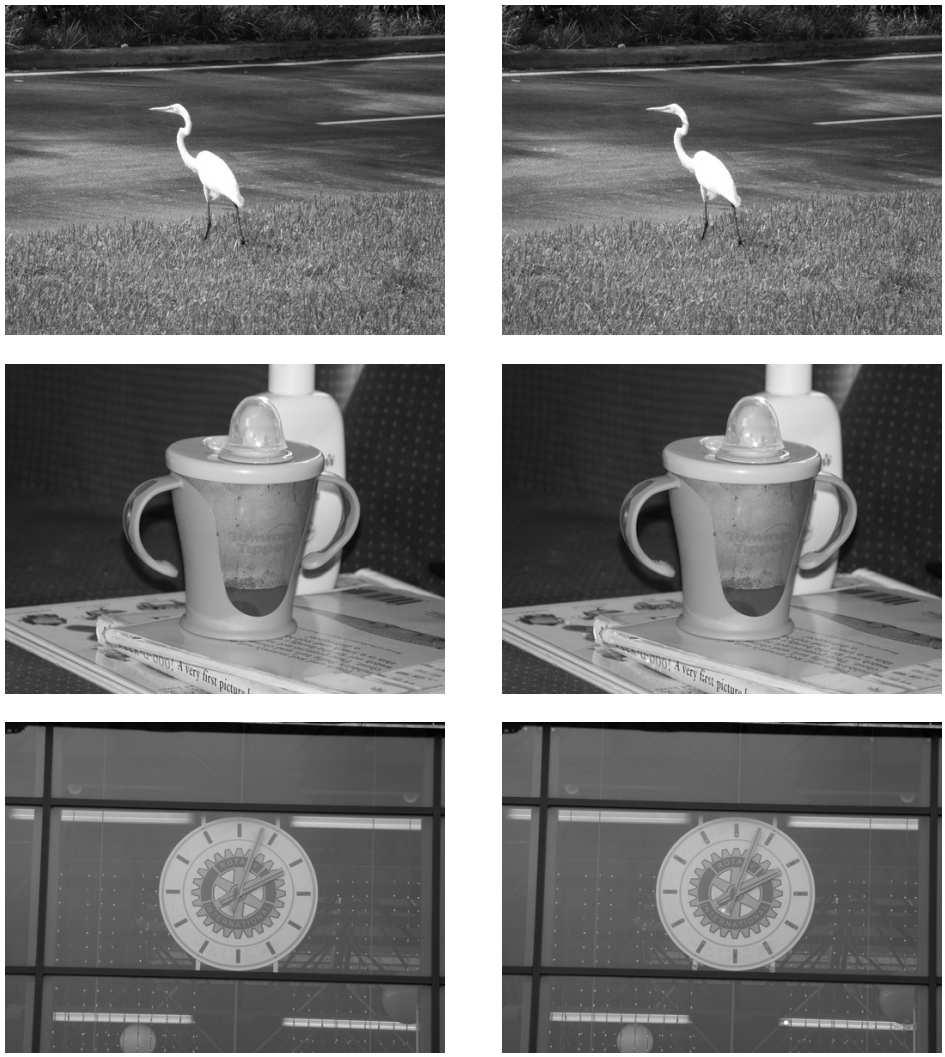


Figure 10.22: *Additional examples of images attacked with* `CLBA` (iii). *Left: authentic; right: forged (octave* $= 0$*,* $M = 8$*,* $KRR \geqslant 99\%$ *for all images).*

Chapter 11

Counter-forensics of SIFT-based copy-move detection

"The only way you can get good,
unless you're a genius, is to copy.
That's the best thing. Just steal."

Ritchie Blackmore, Deep Purple

The attack devised in the previous chapter is now applied to an image forensic scenario to impair SIFT-based detection of copy-move forgeries. We discuss the principal aspects of the new problem in Sec. 11.1. Then, in Sec. 11.2, we briefly review the SIFT-based detector targeted by the proposed counter-forensic scheme. We explain the copy-move Classification-based attack in Sec. 11.3 and we evaluate its performance in Sec. 11.4.

11.1 Introduction and motivations

Copy-move forgery, whereby a portion of an image is copied and pasted once or more times elsewhere into the same image, is one of the most common ways of manipulating the semantic content of a picture. Such manipulation can be used either to conceal undesired portions of the image or to duplicate semantically relevant content. Consequently, the objective of the forensic analyst consists in detecting image areas that are extremely similar to each other. Conversely, the goal of the adversary is to conceal those traces allowing to draw conclusions on said similarity.

The research community has put great effort in the development of algorithms to reveal copy-move forgeries [Christlein et al., 2012]. In principle, all

methods extract some kind of image features and match them with each other, in search of strong similarities interpreted as cues of the forgery. The most common categorisation of these methods is made according to the procedure of extraction: *block-based* and *keypoint-based*. In the former case, an image is divided into overlapping blocks from which some features are extracted and then ordered by similarity; in the latter case, highly descriptive robust points of an image are computed and then a unique descriptor is assigned to each of them. Duplicated content is revealed by matching similar blocks or similar descriptors. A great deal of methods falling into the second category rely on SIFT keypoints [Huang et al., 2008; Pan and Lyu, 2010; Amerini et al., 2011] because of SIFT's capability to discover correspondences between similar visual contents even after several processing.

Despite the intense activity on this topic, little attention has been paid so far to the security of existing algorithms. The first work in this sense is the one in [Nguyen and Katzenbeisser, 2011], where the security of three well-known block-based approaches is tested. It turns out that block-based detection can be mislead by resorting to very basic geometric manipulations, a weakness that is not shared with SIFT-based detectors. In this latter case, as the studies carried out in [Caldelli et al., 2012] have hinted, more challenging solutions are required. In particular, the authors were able to impair a state-of-the-art SIFT-based detector by removing all the keypoints of the duplicated regions using local warping attacks derived from image watermarking. Starting from the above results, the same authors proposed a new technique to suppress SIFT keypoints [Amerini et al., 2013b], whose effectiveness was again demonstrated in a copy-move scenario, with performance far superior to that in [Caldelli et al., 2012], both in terms of efficiency and imperceptibility. The latter work represents the main contribution of this chapter.

11.2 Copy-move forgery detection

In this section we describe the forensic detector targeted by the proposed counter-forensic scheme. Additionally, we provide some pointers to block-based detection to better understand the experimental results of the second part of the chapter (Sec. 11.4.6 in particular).

11.2.1 SIFT-based detection

Generally, the SIFT operator is applied to two images. In the case of copy-move detection, it is instead applied to one image only, since the copied part is within the same image. Expectedly, the descriptors extracted from a cloned region are quite similar to those of the source, thus making possible to discover the manipulation by matching keypoints. During this process, one can also retrieve information about the geometric transformation that has been applied to the cloned region.

The algorithm we aim to counter [Amerini et al., 2011] is based on the above rationale. In a nutshell, it works as follows (see Fig. 11.1). Given an image I, the method first extracts the keypoints $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and their descriptors $D = \{f_1, \ldots, f_n\}$. Then, the best candidate match for each keypoint $\mathbf{x}_i$ is found by identifying its nearest neighbour among the other $n-1$ keypoints, i.e. the keypoint with the minimum Euclidean distance descriptor.

Given a keypoint, a similarity vector $S = \{d_1, d_2 \ldots, d_{n-1}\}$ is defined with sorted Euclidean distances with respect to the other descriptors. The keypoint is matched only if $d_1/d_2 < T_d$ (set empirically to 0.6). By iterating on each keypoint in X, a set of matched points is obtained.

Although this set of linked, isolated keypoints already provides a rough idea of the presence of cloned areas, a clustering procedure is run for improved accuracy. To assess the existence of cloned areas, an *agglomerative hierarchical clustering* is carried out: i) each keypoint is assigned to a cluster; ii) the reciprocal spatial distances among clusters are computed; iii) the closest pair of clusters is found; and iv) the obtained pair is merged into a single cluster. The procedure is repeated until no more pairs can be merged. Upon agglomerating, the forgery is revealed if there are two or more clusters linked by at least 3 pairs of matches points.

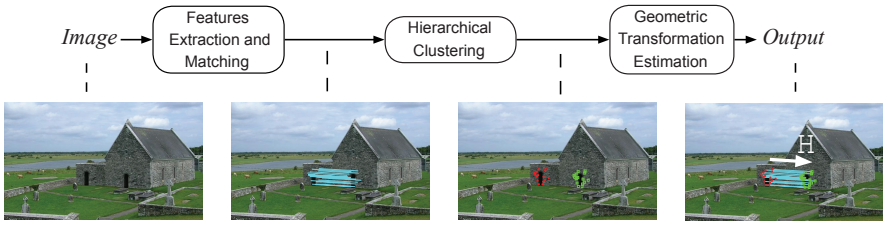


Figure 11.1: *Workflow of the forensic detector of* [Amerini et al., 2011].

11.2.2 Block-based detection

A typical block-based approach divides a grayscale image into overlapping segments by sliding a small window (by a certain number of pixel) from the upper left corner to the lower right corner. For each tile, a feature vector is computed. Such vectors are used to match blocks having the same (or very similar) representations by means of lexicographical sorting. In a nutshell, all feature vectors are arranged into a matrix, where each row corresponds to a block. The idea is that similar rows correspond to similar blocks. To better identify such relationships, rows are ordered so that similar ones come successively [Fridrich et al., 2003a; Popescu and Farid, 2004; W et al., 2006]. By doing so, consecutive pairs of rows correspond to the candidate duplicates, which can be further verified by subsequent post-processing. The robustness of popular block-based copy-move detectors has been studied in [Nguyen and Katzenbeisser, 2011]. It came out that such approaches have several weaknesses, both to simple processing like noise addition, JPEG compression, crop or rescaling and to targeted attacks like randomly changing the least significant bit of each pixel or randomly swapping pairs of low DCT coefficients. Nguyen *et al.* propose a simple attack consisting of cropping and compression that is sufficient to impair the most reliable block-based detectors.

11.3 Copy-move Classification-based attack

For the sake of simplicity, we address the case of a single pair of copy-moved regions. Such an assumption does not affect at all the generality of our approach because the algorithm can be straightforwardly adapted to multiple

versions of the same region or multiple pairs of different regions. It is worth noting that `CLBA` addresses only the first-octave keypoints, i.e. those extracted from the image at its original resolution. This fact prevents `CLBA` from being a threat to applications relying on the few, more robust higher-octaves keypoints. The attack, however, can effectively counter copy-move detection, whereby the majority of matches linking the cloned regions are extracted from the first octave. We adapt `CLBA` for the copy-move scenario to devise a new attack, called `cm-CLBA` (copy-move Classification-based Attack), whose role in the framework of Chapter 8 is depicted in Fig. 11.2.

The key idea consists in analysing the forged image with the targeted forensic detector, taking note of the matches between cloned areas and iteratively removing them. Standard `CLBA` already offers multiple ways to achieve this goal. Although their outcome is the same, however, the quality and perceptibility may greatly vary.

1. The first and most obvious solution is to use standard `CLBA` to completely remove all the keypoints from the image. However, this appears rather excessive and unnecessary.

2. The second solution consists in applying standard `CLBA` to all the keypoints of both cloned regions. This is still excessive, since to destroy a match it is sufficient to eliminate only one of its members.

3. The third solution consists in applying standard `CLBA` to all the keypoints of a single region. Still, due to processing, not all keypoints of a region always match with a corresponding one in the other and thus there is still room for improvement.

4. The fourth solution consists in attacking only the matching keypoints revealed by the copy-move detector. For better concealment of the attack, the manipulation can be distributed over the two regions. Note that if the adversary does not have access to the forensic detector, the most viable solution is the third one.

In Sec. 11.4.2, an example for each approach is provided. In the following, the last solution will be implemented in the `cm-CLBA`.

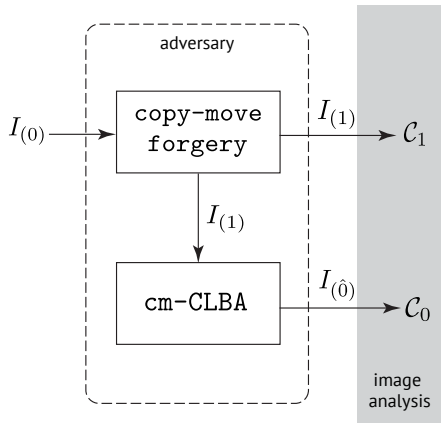


Figure 11.2: `cm-CLBA` *in the framework of* Chapter 8. $\mathcal{C}_0$ *and* $\mathcal{C}_1$ *are respectivey the classes of authentic and forged images.*

11.3.1 Adapting `CLBA` to the copy-move scenario

At each iteration, only one keypoint of each match is manipulated. Let N_m be the number of matches across the two cloned regions A_1 and A_2; first, $\frac{N_m}{2}$ matches are randomly picked and the corresponding keypoints in A_1 are erased; then, the same thing is done for the remaining $\frac{N_m}{2}$ matches by attacking only the corresponding keypoints in A_2. A difference with the standard attack is that the effectiveness is now measured in terms of removed matches. In practical terms, the halting conditions of the attack are controlled by a target match removal rate and a maximum number of iterations. Although presented here as a post-processing procedure, the attack can also be implemented in an integrated fashion. In this latter case, cloned regions are deprived of their matching keypoints and then copied into the image to create a forgery.

Before moving to the next section, two aspects are worth of note. Firstly, it is not always strictly necessary to completely remove every matching pair of keypoints, given that to cope with false positives, copy-move detectors require a minimum number of matches (3-4 usually) to spot a forgery. In spite of this, we preferred a more general solution to the problem and thus we always attempted to remove all the matches. Secondly, the occasional introduction of new keypoints in the proximity of deleted ones should be also considered.

The descriptor of a new keypoint may be similar enough to that of the old one and thus their match may be conserved. In this circumstance, the subsequent iteration of the attack will deal with it. Moreover, the imperceptibility of the attack benefits from those lucky cases in which new keypoints without matches (or false positive matches) are introduced.

11.4 Experimental validation

Four aspects concerning `cm-CLBA`'s performance will be examined:

- the perceptibility with respect to other possible ways of using `CLBA` to hide copy-move;
- the effectiveness in hiding copy-move forgeries with respect to adaptations of state-of-the-art SIFT countering methods;
- the robustness to parameter variations in the copy-move detector;
- the interactions with block based copy-move detection.

The effectiveness of the attack is expressed in terms of Match Removal Rate (MRR), wherein only matches across cloned regions are considered:

$$MRR = \left(1 - \frac{matches\ detected\ after\ attack}{matches\ detected\ before\ attack}\right) \times 100. \tag{11.1}$$

All assumptions we previously made still hold here: we limit the forgery to two cloned regions and all attacks work on first-octave keypoints of the grayscale image. We set the target MRR and the maximum iterations to 100% and 40 respectively. We leave all parameter values of `cm-CLBA` unchanged with respect to `CLBA`. Keypoint detection relies on VLFeat ($edge\_threshold = 10$ and $peak\_threshold = 4$). The detection of copy-move is performed with an implementation of the detector in [Amerini et al., 2011], with the maximum ratio between the distances of two to-be-matched points set to $T_d = 0.6$ as suggested by the authors.

11.4.1 Image data sets

The experimental analysis has been carried out on two different data sets of copy-move forgeries: a large synthetic one and a realistic one composed by 10 man-made manipulations. Each data set has a specific purpose: the former, thanks to its size, allows to fully assess `cm-CLBA`'s performance; the latter, with its realistic forgeries obtained without restrictions on the image processing at adversary's disposal, brings the attack out of a safe laboratory environment. The synthetic data set, consisting of $1,338$ images of size 512×384 or 384×512, has been created from the UCID data set (see Sec. 10.4.1) by automatically copying the central region of each image and pasting it in a random, non overlapping region of the same image.

11.4.2 Perceptibility of `cm-CLBA`

The forged image of Fig. 11.3 is used to demonstrate that the approach chosen by `cm-CLBA` grants equal *MRR*s but superior imperceptibility with respect to other solutions based on standard `CLBA`.

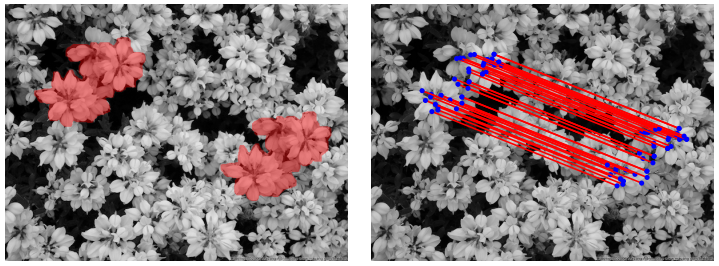
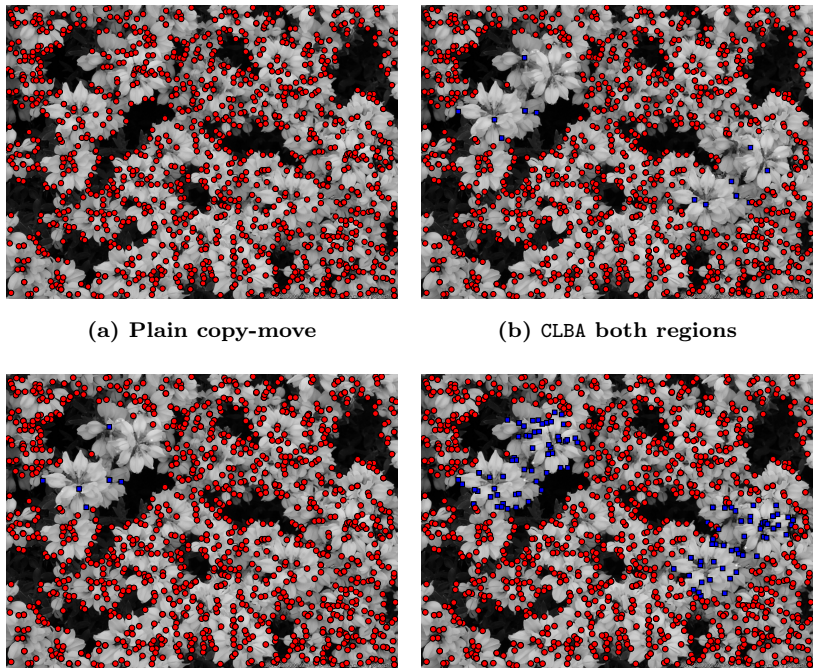


Figure 11.3: *Copy-move forgey with highlighted tampered regions (left) and output of Amerini et al.'s detector (right).*

Three different approaches have been chosen to hide copy-move: i) standard `CLBA` on both regions; ii) standard `CLBA` on a single region; and iii) `cm-CLBA` on both regions. We stress out that all the methods successfully impair the copy-move detector. The outcome in terms of population of keypoints, however, is quite different, as Fig. 11.4 confirms. Red circle markers indicate original keypoints, while blue square markers correspond to keypoints within copy-moved regions following the specified attack. It appears evident

that CLBA leaves very peculiar footprints, in the form of areas full of textures and corners but suspiciously deprived of SIFT features, as in Fig. 11.4 (b)–(c). It goes without saying that this is not the wisest approach, because quality is unnecessarily degraded and the attack is exposed too much to another kind of forensic detection.[1] This problem does not seem to concern cm-CLBA, as it is very hard to notice any difference between the plain copy-move of Fig. 11.4 (a) and the forensically undetectable image of Fig. 11.4 (d).



(a) Plain copy-move **(b) CLBA both regions**

(c) CLBA one regions **(d) cm-CLBA both regions**

Figure 11.4: *Comparison of different copy-move counter-forensic approaches. Red circle markers indicate original keypoints, while blue square markers correspond to keypoints within copy-moved regions following the specified attack. Copy-move detector did not find traces of manipulation in* **(b)**, **(c)** *and* **(d)**.

[1] For more information on this topic, refer to Chapter 12.

11.4.3 Effectiveness on synthetic data set

The UCID data set has been used to randomly create 1,338 copy-move forgeries with 2 cloned regions. All the experiments have been carried out for two tampered areas, i.e. 150×150 and 200×200 pixels. In few cases the automatic procedure did not produce detectable forgeries, since it duplicated regions with less than the minimum number of keypoints required by the forensic detector. We did not consider the contribution of such images to the results.

The envelope of the *MRR* is shown in Fig. 11.5. Regardless of the area, the proposed scheme removed more than 99% of detected matches in most of the images.

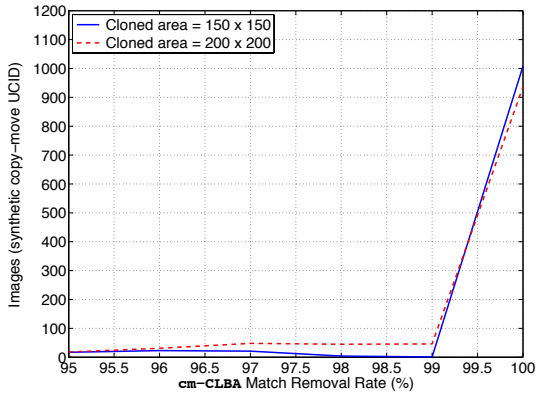


Figure 11.5: *Match Removal Rate of* `cm-CLBA` *on synthetic data set.*

The consequences of this behaviour can be better appreciated in Tab. 11.1, where we summarise the percentage of detected forgeries before and after `cm-CLBA` depending on the minimum number of matches required to reveal the forgery. The countered detector typically requires 3 or 4 matches, hence it classifies correctly only about 6% and 3% of the images, respectively.

Results are satisfactory also from the point of view of the quality (see Fig. 11.6). The average PSNR and SSIM across the two cloned regions do not change substantially with their size and reach minimum values of 41.8 dB and 0.945 for $MRR = 100\%$.

	Minimum matches for detection									
	1		**2**		**3**		**4**		**5**	
Area	Before	After	Before	After	Before	After	Before	After	Before	After
150	1	**0.19**	1	**0.1**	1	**0.06**	0.99	**0.03**	0.97	**0.02**
200	1	**0.23**	1	**0.13**	1	**0.07**	0.99	**0.03**	0.98	**0.02**

Table 11.1: *Detection accuracy of* [Amerini et al., 2011] *before and after* `cm-CLBA`.

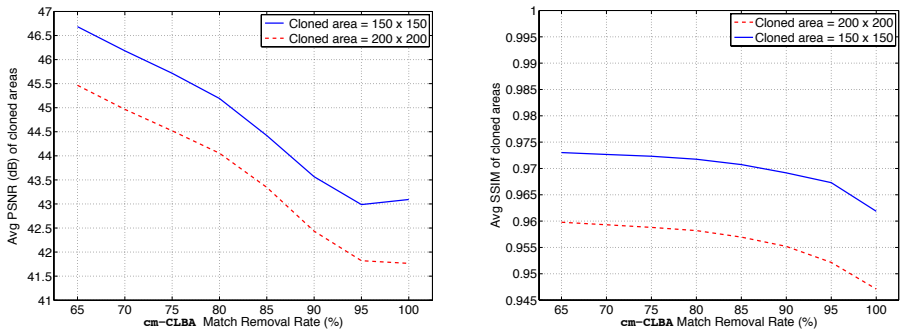


Figure 11.6: *Quality metrics of* `cm-CLBA` *on synthetic data set.*

11.4.4 Effectiveness on realistic data set

The next experiment has been carried out on 10 realistic copy-move forgeries, created manually without any restrictions on image properties or on the processing at counterfeiter's disposal. Tab. 11.2 confirms that the conclusions drawn for the general full-frame scenario of Sec. 10.4.2 remain valid for the copy-move scenario. The classification-based method represents again the best trade-off between removal (all matches deleted within the lowest number of iterations) and perceptual quality (average PSNR of 35.2 dB second only to the `Smoothing` attack). More importantly, it emerges from Tab. 11.3 that `cm-CLBA` avoids detection on all images, regardless of the threshold employed by the forensic detector.

Fig. 11.7 shows an example of copy-moved regions following the four attacks (image I_7). In this case, `Collage` can compete with `cm-CLBA` in terms of

	cm-CLBA			Smoothing		
Image (matches)	After	Iter.	PSNR	After	Iter.	PSNR
I_1 (37)	0 (**100**%)	14	41.2	12 (67.6%)	40	39.4
I_2 (66)	0 (**100**%)	20	39.9	29 (56.1%)	40	47.7
I_3 (150)	0 (**100**%)	23	48.1	103 (31.3%)	40	57.8
I_4 (41)	0 (**100**%)	15	42.8	11 (73.2%)	40	42.1
I_5 (23)	0 (**100**%)	31	47	8 (65.2%)	40	44.5
I_6 (56)	0 (**100**%)	17	45.8	27 (51.8%)	40	41.1
I_7 (55)	0 (**100**%)	28	34.3	10 (81.8%)	40	40.9
I_8 (32)	0 (**100**%)	15	45	8 (75%)	40	38.7
I_9 (52)	0 (**100**%)	19	45.8	21 (59.6%)	40	48.6
I_{10} (53)	0 (**100**%)	21	44.2	16 (69.8%)	40	46.7
Average:	**100**%	21	43.4	**63.1**%	40	45.1

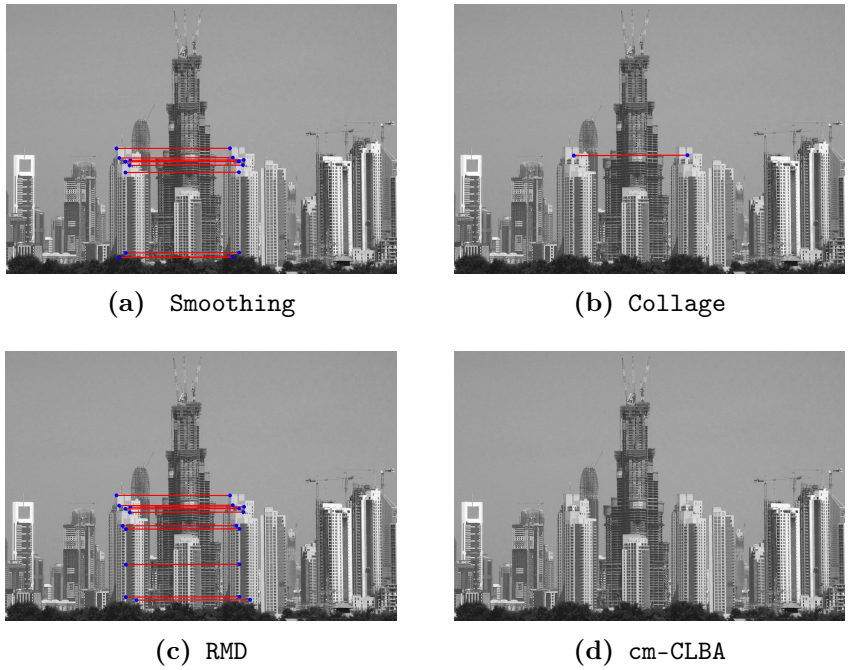
	RMD			Collage		
Image (matches)	After	Iter.	PSNR	After	Iter.	PSNR
I_1 (37)	5 (86.5%)	40	30.7	3 (91.9%)	40	28.6
I_2 (66)	21 (68.2%)	40	25.6	6 (90.9%)	40	33.7
I_3 (150)	0 (**100**%)	26	41.2	7 (95.3%)	40	43.1
I_4 (41)	9 (78.8%)	40	27.4	2 (95.1%)	40	32.2
I_5 (23)	8 (65.2%)	40	25.3	0 (**100**%)	9	28.0
I_6 (56)	7 (87.5%)	40	26.3	3 (94.7%)	40	33.3
I_7 (55)	10 (81.8%)	40	26.2	1 (98.2%)	40	28.4
I_8 (32)	1 (96.8%)	40	30.1	2 (93.8%)	40	29.4
I_9 (52)	10 (80.8%)	40	25.3	2 (96.2%)	40	33.5
I_{10} (53)	22 (58.5%)	40	29.4	3 (94.3%)	40	32.0
Average:	**80.3**%	39	28.8	**95**%	37	32.2

Table 11.2: *Results of* cm-CLBA *and other attacks on realistic data set. Tests on* 10 *copy-move manipulated images: left matches are listed according to the relative iteration.*

	Minimum matches for detection				
Attack	**1**	**2**	**3**	**4**	**5**
`cm-CLBA`	**0**	**0**	**0**	**0**	**0**
`Smoothing`	1	1	1	1	1
`RMD`	0.9	0.8	0.8	0.8	0.8
`Collage`	1	0.8	0.5	0.2	0.2

Table 11.3: *Detection accuracy of* [Amerini et al., 2011] *following* `cm-CLBA`*, depending on the minimum number of matches required to claim the manipulation (realistic data set).*

removal, although the quality of the former is significantly lower than that of the latter (28.4 dB versus 34.3 dB).



(a) `Smoothing` **(b)** `Collage`

(c) `RMD` **(d)** `cm-CLBA`

Figure 11.7: *Matches of copy-moved image I_7 after the four attacks.*

11.4.5 Dependence on the copy-move detector's parameter

The results discussed so far have been obtained by letting the threshold T_d for the copy-move detection be 0.6, as suggested by the authors of the technique [Amerini et al., 2011]. Recall that T_d corresponds to the maximum allowed ratio between the distance of two to-be-matched points across the cloned regions. Since increasing such value yields a higher amount of matches, it is important to understand whether this fact affects `cm-CLBA`'s performance.

To this purpose, we set up the following experiment. First, we created 100 large copy-move forgeries by cloning a random 400×400 region containing at least 100 keypoints. Then, we ran the CMFD on the images with progressively increasing $T_d \in [0.1, 1]$; finally, for each threshold we measured the correct and the wrong matches. Fig. 11.8 (left) shows the average number of the two categories of matches as a function of the threshold. We can observe that, while the correct matches remain approximately constant for $0.15 \leqslant T_d \leqslant 1$, the wrong matches rapidly increase for $T_d > 0.7$, to the point that they represent the 75% of all the detected matches, as shown in Fig. 11.8 (right).

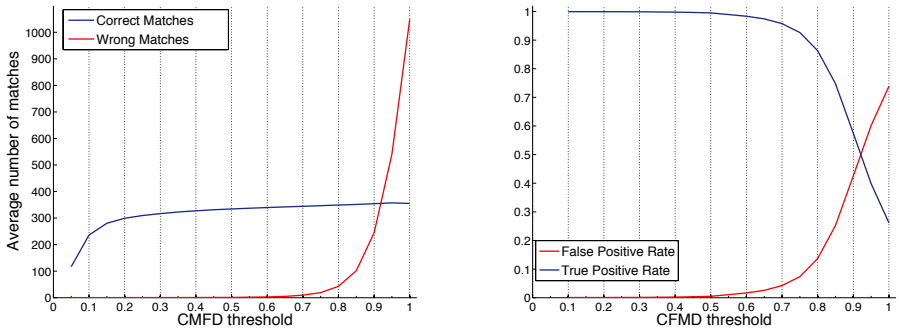


Figure 11.8: `cm-CLBA` *dependence on the copy-move detector's threshold. Left: average number of correct and wrong matches: right: fraction of correct and wrong matches.*

Based on the above experiment, we can conclude that the value originally assigned to T_d is a good trade-off between the number and the reliability of the matches and, more importantly, that variations of T_d do not affect the

performance of the attack. Since the number of correct matches does not vary significantly for $T_d \in [0.2, 1]$, the results we obtained in this chapter for $T_d = 0.6$ retain their validity for the other possible values. Furthermore, either decreasing or increasing the threshold damages the forensic analyst. In fact, low T_d values cut down the number of detected matches, thus simplifying their removal and reducing the impact on the attacked image's quality, whereas high T_d values trigger false positive matches, thus playing in favour of the counterfeiter by discrediting the trustworthiness of the forensic analysis. The average number of correct and wrong matches for the test data set are shown in Tab. 11.4; the second row is of particular interest since it highlights the incapability of the CFMD of revealing the copy-move following `cm-CLBA`, regardless of T_d's value.

	CMFD threshold									
Matches	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
Correct before	203	266	282	293	299	304	308	313	317	318
Correct after	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Wrong before	0	0	0	1	1	2	7	34	205	909
Wrong after	0	0	0	0	1	2	8	37	223	861

Table 11.4: *Average number of correct and wrong matches depending on CMFD's threshold before and after* `cm-CLBA`.

The images in Fig. 11.9 provide an example of the above observations. In the first row, the outcome of the forensic analysis is shown for $T_d = \{0.1, 0.6, 0.8\}$; the number of correct matches for the three thresholds is respectively 9, 54 and 70 and the number of wrong matches is 0, 6 and 29. Only the wrong matches are left into the image once they are attacked with `cm-CLBA` (second row).

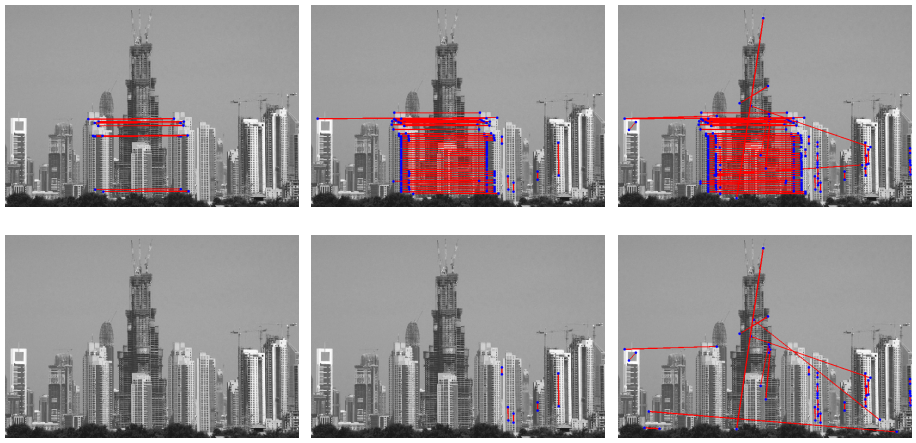


Figure 11.9: *Effect of different CFMD's thresholds on* `cm-CLBA`. *Authentic image (first row) and forged image (second row) for* $T_d = \{0.1, 0.6, 0.8\}$. *Regardless of the threshold, the CMFD is impaired and only wrong matches are left into the images.*

11.4.6 Relationships with block-based detection

In literature, there exist a number of works comparing block-based and keypoint-based copy-move detection (see [Christlein et al., 2012] for example). It would also be interesting to take into consideration the relationships between the corresponding counter-forensic schemes. *What does happen when a SIFT-countered copy-moved image is analysed by means of a block based detector?* (and vice-versa). Moreover, *could heterogeneous attacks be combined?* In this section, we provide the first exploratory answers to these questions, while a more in-depth analysis is left to future endeavours.

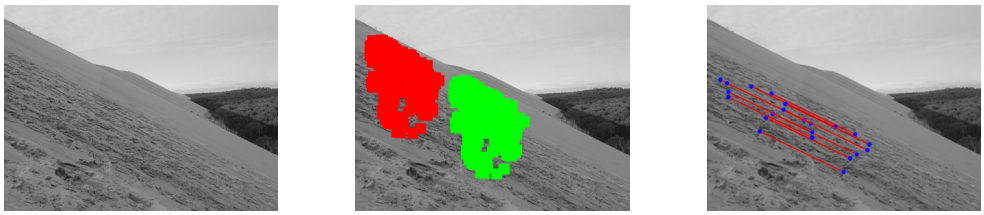
For the following test, we selected two copy-move detectors and two copy-move counter-forensic attacks: the block-based detector in [Fridrich et al., 2003a], where the lexicographically ordered feature vectors rely on low frequency DCT coefficients;[2] the SIFT-based detector in [Amerini et al., 2011]; the `cm-CLBA` attack; and the geometric attack in [Nguyen and Katzenbeisser,

[2] An implementation of Fridrich *et al.*'s detector is freely available as a Gimp pluging at: https://sites.google.com/site/elsamuko/forensics/clone-detection.

2011]. The latter scheme proved to be effective against a number of block-based detectors, including the one by Fridrich *et al.* It consists of a crop of 3 pixels (column-wise and row-wise), followed by two JPEG compressions (qualities 70 and 60) and a final resampling (bicubic interpolation) back to the size of the original image. The popular copy-move forgery we used for this experiment is shown in Fig. 11.10.

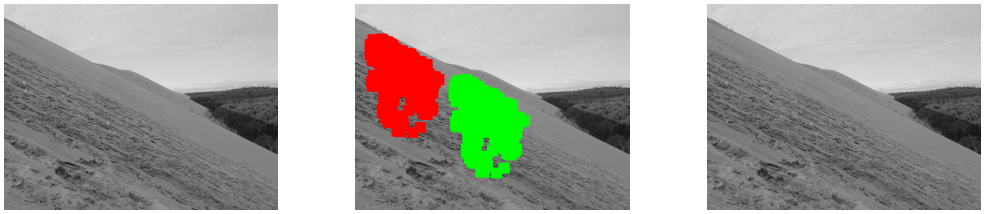
Figure 11.10: *Authentic image (left) and copy-move forgery (right).*

When invoked, both forensic detectors correctly reveal the manipulation. At this point, two (supposedly) forensically undetectable images have been obtained by means of `cm-CLBA` and the geometric attack and the detectors invoked again. Expectedly, the block-based attack does not impair SIFT-based detection and vice versa. Obviously, SIFT features cannot be removed by a simple geometric attack. On the other hand, `cm-CLBA` is designed to modify pixel values as less as possible, thus not altering the features analysed by the block-based detector. However, the cascade of the two attacks (regardless of their order) is effective against both detectors. Fig. 11.11 summarises the various phases of the experiment. In conclusion, the above analysis suggests that cooperation between different counter-forensic schemes is the key to impair a wider spectrum of detection techniques.



Copy-move **Block-based** **SIFT-based**

Geom. attack on (a) **Block-based** **SIFT-based**



CLBA on (a) **Block-based** **SIFT-based**

(d) + (g) on (a) **Block-based** **SIFT-based**

Figure 11.11: *Block-based versus SIFT-based copy-move countering. Each version of the counter-forensically forged image (first column) is analysed by means of the two detectors (second and third columns).*

11.5 Concluding remarks

In this chapter we adapted the Classification-based attack to the forensic scenario of copy-move forgery detection. We obtained several interesting results. First of all, it was possible to successfully hide copy-move tampering by removing SIFT features in the cloned areas of an image. By doing so, we hindered a state-of-the-art copy-move detector based on SIFT. Secondly, we obtained a good trade-off between effectiveness and imperceptibility of the attack by means of expedients tailored to the scenario. Finally, it was possible to interact with other copy-move hiding schemes to impair different categories of detector at the same time. Concerning possible extensions, it would be of practical interest to extend the copy-move attack to an arbitrary number of cloned regions. Moreover, said attack could be validated against similar SIFT-based detectors, such as those in [Huang et al., 2008] and [Pan and Lyu, 2010], to verify whether it retains its effectiveness.

Chapter 12

Forensic analysis of SIFT keypoint removal

"The world is full of obvious things which nobody by any chance ever observes."

The Hound of the Baskervilles
Arthur Conan Doyle

EVEN THOUGH resilient to several processing, state-of-the-art techniques for copy-move detection can be successfully challenged by manipulating SIFT features to prevent the algorithms from linking cloned image regions. Like any other processing, however, keypoint removal leaves traces into the manipulated areas, in the form of high-variance textured regions where keypoints should be found but are instead absent. Despite that, no method has been devised so far to find such footprints. To tackle with this problem and restore the efficiency of SIFT-based forensic analysis, it is then necessary to devise adversary-aware algorithms to understand whether SIFT keypoints have been artificially removed.

In this chapter we devise three forensic detectors for the identification of images whose SIFT keypoints have been artificially removed. The proposed algorithms scan image regions with sufficiently high variance in search of suspect inconsistencies in the number and in the distribution of SIFT keypoints. By relying on such algorithms, the forensic analyst can decide on the authenticity of the image as a whole or localise tampered regions within the image by means of a sliding window approach.

The chapter is organised as follows. Secs. 12.1–12.3 describe the three keypoint removal detectors and Sec. 12.4 experimentally validates their capability to reveal both global and local removal forgeries. Finally, Sec. 12.5 investigates more in-depth the robustness of the most performing tool.

12.1 Keypoint-to-Corner Ratio detector

The first keypoint removal detector is based on two simple observations. The first observation is that SIFT keypoints lie in proximity of corners, i.e. interest points where two edges intersect, which can be identified by means of methods like the Harris [Harris and Stephens, 1988] or the Shi and Tomasi [Shi and Tomasi, 1994] detectors. Such observation is supported both by SIFT theory, which selects keypoints according to a corner response metric inspired to the Harris detector, and by the experimental results that we will discuss in Sec. 12.4.3.

The second observation is that all removal attacks devised so far have been designed to preserve as much as possible image content by working on small neighbourhoods of the keypoints. As a consequence, while the number of keypoints is significantly reduced by a removal attack, the number of corners is subject to negligible variations.

If the above two assumptions hold, then we can understand whether an image has been subject to a keypoint removal attack based on the conservation of corners and on the reduction of keypoints in proximity of corners. Let $N_{corners}$ be the number of corners; consider a square patch of side d centred on each corner and let $N_{keypoints}$ be the total amount of keypoints falling into such patch. An image is labelled as forged if the ratio between the above two quantities (Keypoint-to-Corner Ratio or KCR[1]) falls below a threshold:

$$KCR = \log_{10}\left(\frac{N_{keypoints}}{N_{corners}}\right) \overset{?}{\leqslant} T_1. \tag{12.1}$$

Under the hypothesis that keypoint removal does not affect corners, the denominator of Eq. (12.1) is approximately the same for the authentic and the manipulated image. Conversely, the numerator of Eq. (12.1) is drastically reduced by the attack, hence the KCR index of the attacked image is smaller than that of its authentic counterpart. The value d defining the corner proximity and the threshold T_1 are empirically derived in Secs. 12.4.3–12.4.5.

Even though technical details are left to the sequel, in Fig. 12.1 we show an example of the population of SIFT keypoints and corners before and after

[1] In the sequel we will use the mathematical font (i.e. KCR) for the numerical index and the typewriter font (i.e. `KCR`) for the detector.

CLBA to emphasise the conservation of corners and the reduction of keypoints. To help understanding the consequences of the attack, we highlighted the differences in the last column of Fig. 12.1, where the green markers identify the deleted keypoints (top) and corners (bottom). The *KCR* index is respectively -1.6 for the authentic image and -15.7 for the forgery.

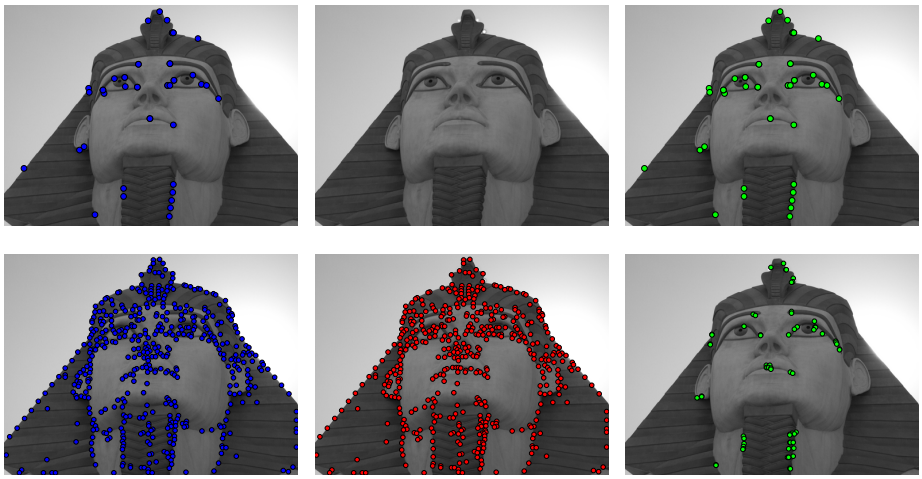


Figure 12.1: *Impact of keypoint removal on distribution of keypoints (first row) and corners (second row). First column: before* CLBA*; second column: after* CLBA*; third column: difference.*

12.2 CHI-square distance detector

The second detector is based on the observation that keypoints are concentrated in image regions characterised by high variance, due to the fact that SIFT discards candidate keypoints whose neighbourhoods have low contrast. Consequently, an image targeted by keypoint removal should exhibit high-variance regions unnaturally deprived of keypoints. To translate this intuition into a detectable footprint, we studied the distributions of SIFT keypoints in image blocks characterised by different variance. In practice, first we assigned each non overlapping block of side B of the image to one among three classes depending on its variance: *low*, *medium* and *high* (see Sec. 12.4.1 for tech-

nical details on variance-based classification); then, we examined each class in search of anomalies in the distribution of keypoints. More specifically, we adopted the following strategy.

1. Divide an image I into non-overlapping 32×32 blocks whose variance is assigned either to the *low*, *medium* or *high* class.

2. For each class of variance and given a fixed amount of bins (10 in our implementation), compute the percentage of blocks containing a certain number of keypoints.[2] The resulting percentages correspond to histograms referred to as h_L, h_M and h_H.

3. Attack the image with `CLBA` and repeat steps 1)–2) on the forged image.

The result of the above procedure is shown in Fig. 12.2, where the first row corresponds to the histograms of an authentic image and the second row to those of its tampered version. As a matter of fact, we can recognise different shapes, especially in the case of *medium* and *high* variance histograms, which reveal an anomalous percentage of textured blocks without keypoints.

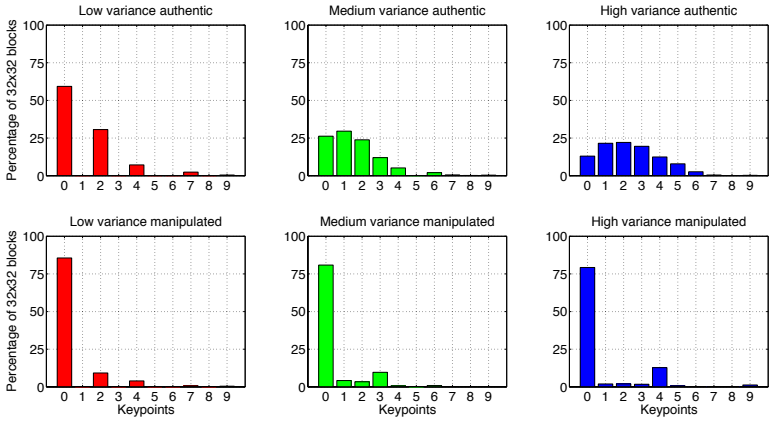


Figure 12.2: *Histograms h_L, h_M and h_H for a test image. Top row: authentic; bottom row: forged.*

[2]For example, in the case of the *medium* class, for $bin = 0$, we count the percentage of medium blocks containing 0 keypoints; for $bin = 1$, the percentage of medium blocks containing 1 keypoint and so on, until we reach the last bin.

Since one image alone is not sufficient to draw any conclusion on the reliability of the above footprint, we repeated the same procedure on a large data set of images (see Sec. 12.4.1). We accumulated each class histogram of all the images with a bin-by-bin sum and we averaged them, first for the authentic images and then for their forged versions, as follows:

$$\widehat{H}_{\mathcal{C}_k}^{(auth)} = \frac{1}{N_{auth}} \sum_{i=1}^{N_{auth}} h_{\mathcal{C}_k}^{(i)} \tag{12.2}$$

$$\widehat{H}_{\mathcal{C}_k}^{(forged)} = \frac{1}{N_{forged}} \sum_{j=1}^{N_{forged}} h_{\mathcal{C}_k}^{(j)}, \tag{12.3}$$

where: $h_{\mathcal{C}_k}^{(i)}$ is the histogram of blocks belonging to the variance class $\mathcal{C}_k$ for the i-th image; N_{auth} and N_{forged} are the number of authentic and forged images, respectively.

The resulting histograms are shown in Fig. 12.3; the trends we observed on a single image are now even more evident. We will refer to the accumulated histograms as $\widehat{H}_L^{(auth)}$, $\widehat{H}_M^{(auth)}$ and $\widehat{H}_H^{(auth)}$ for authentic images and $\widehat{H}_L^{(forged)}$, $\widehat{H}_M^{(forged)}$ and $\widehat{H}_H^{(forged)}$ for forged images. We will consider such histograms our *ground truth*.

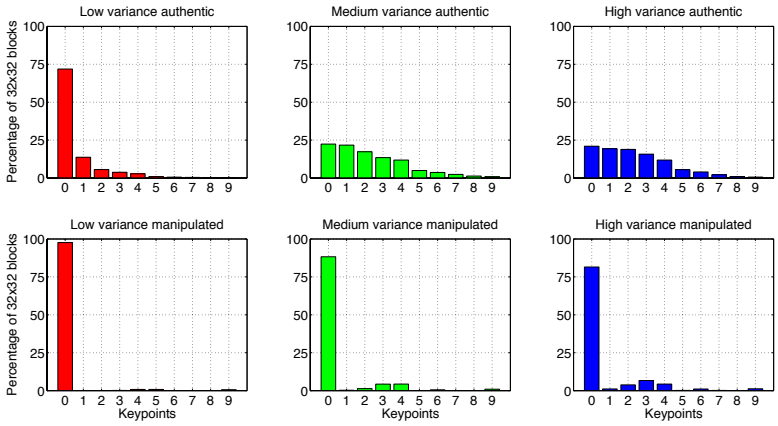


Figure 12.3: *Accumulated reference histograms for the* `CHI` *detector. Top row: authentic; bottom row: forged.*

The shape difference for *low* variance histograms is not as pronounced as for the other classes because candidate keypoints in uniform regions are less likely to pass the SIFT contrast check. Therefore, we rely only on the other two histograms. In particular, we look at the distance between the *medium* variance histogram of the image under analysis and $\widehat{H}_M^{(auth)}$ and compare it against a threshold. In our implementation we chose the chi-square distance [Pele and Werman, 2010], hence the name `CHI` detector:

$$\chi^2 = \frac{1}{2}\sum_{l=1}^{L}\frac{\left(h_M(l) - \widehat{H}_M^{(auth)}(l)\right)^2}{h_M(l) + \widehat{H}_M^{(auth)}(l)} \overset{?}{\geqslant} T_2, \tag{12.4}$$

where l denotes one histogram bin and L indicates the total amount of bins. The image is considered as forged if the distance exceeds T_2, whose value is empirically derived in Sec. 12.4.5.

Note that in Eq. (12.4) we chose the histogram of *medium* blocks but, according to our experiments, similar performance can be attained by considering the histogram of *high* variance blocks. Nevertheless, we prefer the former because it is more reliable on images characterised by few textured regions and consequently by few blocks with high variance. Finally, to exemplify the above concepts, the χ^2-score is 42.4 for the authentic image of Fig. 12.1 and 92.1 for its manipulated version.

12.3 SVM detector

The last detector is based on the same principles underlying the `CHI` detector but does not require the knowledge of the reference histograms. In fact, the histograms h_L, h_M and h_H extracted from the image under analysis are first concatenated into a feature vector $\mathbf{F} = [h_L, h_M, h_H]$ and then fed to a probabilistic Support Vector Machine [Platt, 1999] that has been trained with a large amount of examples $\mathbf{F}$ coming from authentic and forged images. According to this detector, the analysed image has been tampered with if the SVM output *out* is higher than a threshold:

$$out = Prob(I \text{ is forged}) \overset{?}{\geqslant} T_3. \tag{12.5}$$

The value of T_3 is empirically determined in Sec. 12.4.5.

12.4 Experimental validation of keypoint removal detectors

We now validate experimentally the hypotheses underlying each algorithm and discuss the performance of each technique in detecting both global and local SIFT keypoint removal.

12.4.1 Experimental setup

First-octave keypoints are removed by means of `CLBA` with 100% target removal rate and a maximum of 40 iterations, unless specified otherwise. For the detection of SIFT features we rely on VLFeat with edge and peak-difference thresholds set to 10 and 4 respectively. It is important to point out that the results we show are obtained by taking into account only the keypoints belonging to first octave (i.e. 0), in compliance with the assumption underlying `CLBA`. The conclusions drawn for the detectors, however, are general and retain their validity on all octaves.

The variance-based classification is carried out on square blocks of side $B = 32$. We begin by computing the variance map V for the whole image I, where the value of each element $V(i, j)$ is the variance within a square window centred on the corresponding image pixel $I(i, j)$. We first binarise V and then we subdivide both V and I into non overlapping $B \times B$ blocks. Let I_B be one of such image blocks and V_B the corresponding variance block; the ratio τ of pixels with value 1 with respect to the total number of pixels of V_B is computed and I_B is classified as follows:

$$\text{if } \begin{cases} 0 \leqslant \tau \leqslant \frac{1}{3} \Rightarrow I_B \in \mathit{low} \\ \frac{1}{3} < \tau \leqslant \frac{2}{3} \Rightarrow I_B \in \mathit{medium} \\ \frac{2}{3} < \tau \leqslant 1 \Rightarrow I_B \in \mathit{high}. \end{cases} \tag{12.6}$$

In Fig. 12.4 we provide an example of visual content (first row) and binary variance map (second row) for the three classes of blocks.

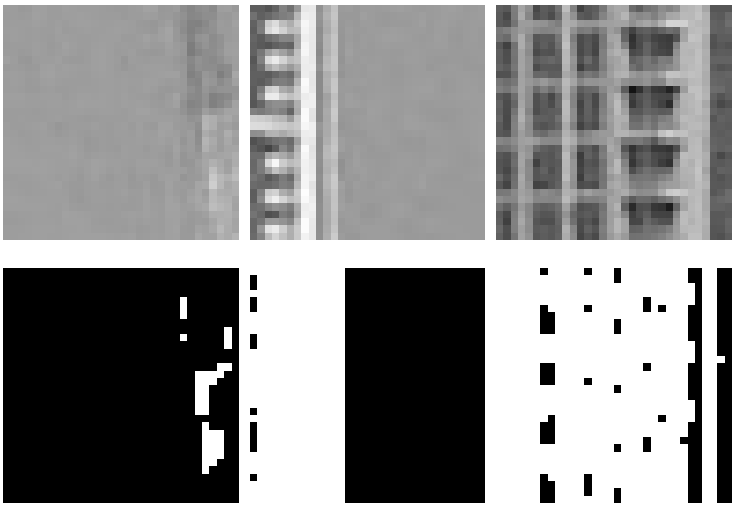


Figure 12.4: *Example of visual content (first row) and binary variance map (second row) for the 3 classes of blocks. Left to right: low, medium and high. Pixels with value* 1 *are respectively* 4%*,* 39% *and* 86%*.*

12.4.2 Image data sets

We collected two image data sets to test the algorithms, one consisting of the first $1,000$ images of the well-known INRIA Holidays data set [Jegou et al., 2008] and one consisting of the first 100 images of the ancillary INRIA Copydays data set. In the sequel, we will refer to the data sets as *Holidays1000* and *Copydays100*. To limit the complexity of the experiments, we downscaled all the images to 1600×1200 pixels. We have chosen fairly large images because the amount of keypoints allows us to assess more accurately the performance of the detectors; however, in Sec. 12.4.5 we also consider smaller images.

The idea behind the two data sets is to use the largest one for the verification of the hypotheses underlying the detectors, for parameter tuning (e.g. determining thresholds) and to train the SVM. Once we gathered such information, the three algorithms are tested on the smaller data set.

12.4.3 Verification of the hypotheses behind the detectors

(A) SIFT keypoints lie in proximity of corners

For all the images of the *Holidays1000* data set, we measured the percentage of SIFT keypoints falling within a $d \times d$ neighbourhood of the corners provided by three extractors: Harris [Harris and Stephens, 1988], Shi and Tomasi [Shi and Tomasi, 1994] and FAST (Features from Accelerated Segment Test) [Rosten and Drummon, 2005]. When $d = 0$, we count only the number of keypoints coinciding with corners; as d increases, we also include the keypoints in the proximity of the corner. Clearly, a keypoint is counted only once even if it falls in the neighbourhood of more than one corner. Fig. 12.5 shows the result of this experiment averaged over all the images. The percentage of keypoints coinciding with corners is lower for the Shi and Tomasi and the FAST detectors because they provide less numerous but more robust corners. However, regardless of the algorithm, on average more than 95% of the keypoints of an image are contained in neighbourhoods of side $d = 3$, thus confirming our starting assumption.

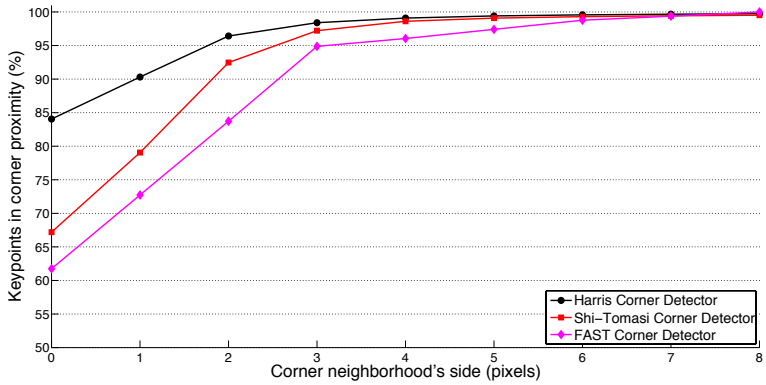


Figure 12.5: *Percentage of keypoints in the $d \times d$ neighbourhood of corners for three different corner extractors.*

(B) Keypoint removal does not affect corners

Fig. 12.6 reports the difference in the number of Harris corners before and after CLBA is carried out on the *Holidays1000* data set (given the similarity, we omit the other algorithms for the sake of brevity). For each image, we express the difference between the number of corners of the authentic and CLBA-forged images in terms of percentage. The reduction in the number of corners in the majority of the images is due to the smoothing operators used by the removal attack; nevertheless, for 97% of the images the difference is bounded in $[+1\%, -3\%]$.

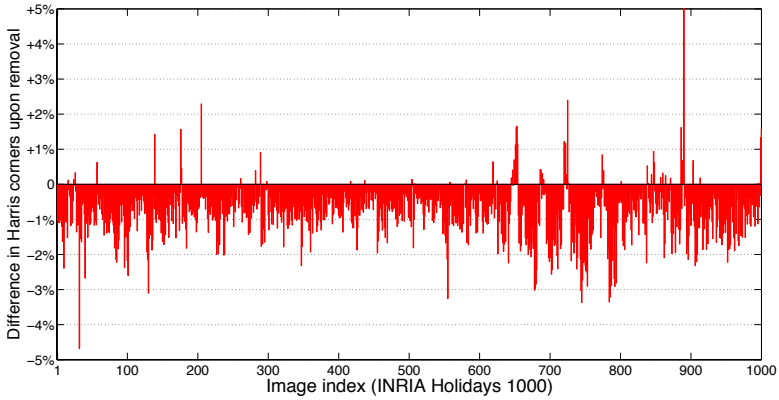


Figure 12.6: *Difference in the number of Harris corners following* CLBA.

(C) Keypoint removal does not affect block classification

The procedure leading to the graphs of Fig. 12.7 is the same as above. This time, however, we plot the differences in number of *low*, *medium* and *high* variance blocks before and after the removal attack. Again, variations are not significant, being confined to the interval $[-1\%, +1\%]$ for all classes.

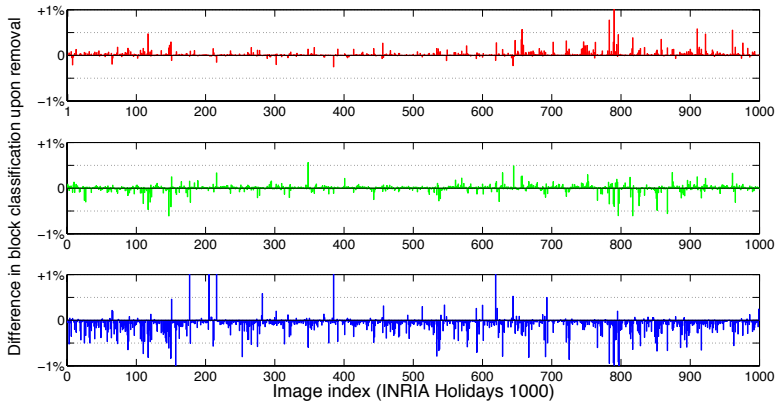


Figure 12.7: *Difference in block-based variance classification following* `CLBA`. *Top: low variance; middle: medium variance; and bottom: high variance.*

12.4.4 Detection of full-frame keypoint removal

The contribution of this section is twofold: first, we analyse and compare the performance of each detector by means of Receiving Operator Characteristics (ROC); then, we derive the thresholds ensuring an acceptable trade-off between the probabilities of detection and false alarm.

(A) Performance of the `KCR` detector

We now evaluate the performance of the `KCR` detector based on the three different corner extractors introduced in Sec. 12.4.3-A, whose $d \times d$ neighbourhood wherein to count the keypoints is set to 3×3.

To assess the discriminative power of the KCR index in Eq. (12.1), we computed the scattergram shown in Fig. 12.8. In practice, the index is calculated for all the images of the *Holidays1000* data set and their forged versions (respectively, blue squares and red circles). The observations sitting exactly on the x-axis correspond to the images with very low KCR, which we set to -4.5 for better readability. More precisely, in Fig. 12.8 we show the scattergram obtained by relying on the FAST corner extractor [Rosten and Drummon, 2005]; we omit the other two graphs because of their extreme similarity to the one being displayed.

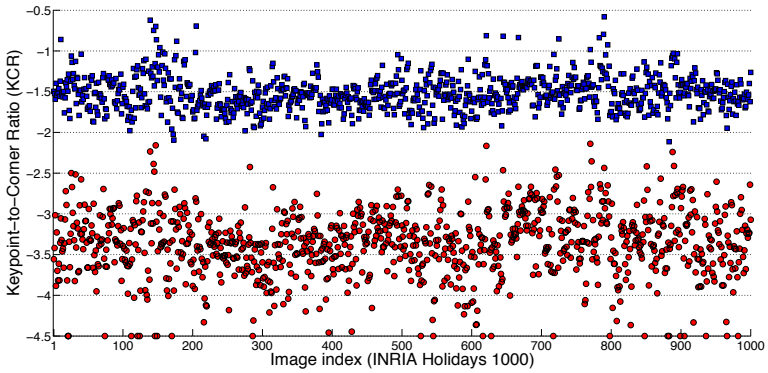


Figure 12.8: *KCR scattergram. Blue squares: authentic images red circles:* `CLBA`*-forged images.*

It can be clearly seen how the indexes cluster into two distinct groups that are effectively separable, as confirmed by the ROC curves of Fig. 12.9. We obtained the curves of each subfigure by varying the threshold separating the clusters in the interval $[0, -5]$ and `CLBA`'s target removal rate in the interval $[10\%, 100\%]$, in order to understand to what extent the forgery is detectable. We carried out the above experiment for each of the corner extractors: Shi and Tomasi (Fig. 12.9 left), Harris (middle) and FAST (right).

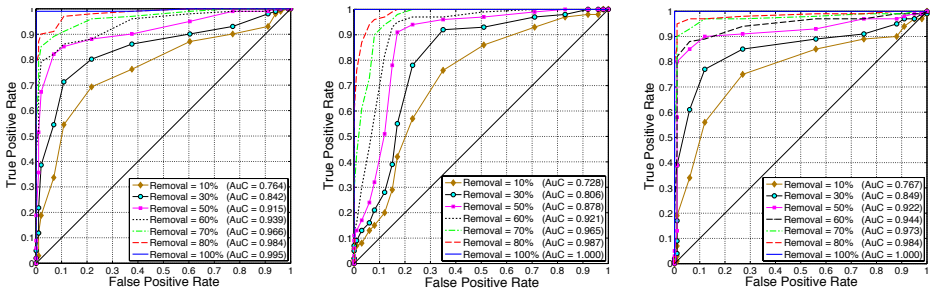


Figure 12.9: *ROCs of* `KCR` *detector depending on removal rate for each corner extractor. Left: Shi and Tomasi; middle: Harris; and right: FAST.*

We can observe two facts. First of all, the more keypoints are left into the image, the harder it is for the detector to separate the clusters; regardless of that, satisfactory results can be obtained even for very low removal rates.

Secondly, the performance of the detector does not depends significantly on the algorithm used to extract the corners, as confirmed by the comparison in terms of Area Under Curve (AUC) and true positive rate for a fixed false alarm probability in Tabs. 12.1–12.2. This is a very important conclusion for the generality of the method, which appear reliable as long as the image content is significant.

	Keypoint Removal Rate						
Corners	**100%**	**80%**	**70%**	**60%**	**50%**	**30%**	**10%**
Shi and Tomasi	0.99	0.99	0.97	0.94	0.92	0.84	0.76
FAST	1	0.98	0.97	0.94	0.92	**0.85**	**0.77**
Harris	1	0.99	0.96	0.92	0.87	0.80	0.73

Table 12.1: *AUC for the* `KCR` *detector based on different corner extractors.*

	Keypoint Removal Rate						
Corners	**100%**	**80%**	**70%**	**60%**	**50%**	**30%**	**10%**
Shi and Tomasi	0.99	0.97	0.91	0.86	0.85	0.71	0.54
FAST	1	0.97	**0.98**	**0.9**	**0.9**	**0.77**	**0.56**
Harris	1	0.97	0.94	0.82	0.51	0.28	0.2

Table 12.2: *True positive rate for false positive rate* 0.1 *for the* `KCR` *detector based on different corner extractors.*

For the rest of the discussion, we will consider only the `KCR` detector based on FAST corners. The reason of this choice is twofold.

- According to Tab. 12.2, for lower keypoint removal rates ($KRR \leqslant 50\%$) and low probabilities of false alarm (0.1) the FAST extractor is significantly more accurate than Harris (+36-49%). This fact is probably due to the higher amount of points provided by Harris, which tend to unnecessarily increase the denominator of Eq. (12.1). While this has no consequences on those images that have been deprived of the majority of keypoints, problems arise when distinction is made harder by lower removal rates.

- The difference in the performance of FAST and Shi and Tomasi extractors is not as noticeable as above. Nevertheless, the accuracy of the KCR based on the former technique is superior by 2-6% for false positive rate 0.1. It is worth noting that, despite its acronym, the FAST extractor is 4 times slower than the Shi and Tomasi extractor. However, its processing time remains quite manageable, as it takes about 35 seconds to process a 1600×1200 image.

(B) Performance of the CHI detector

We validated the CHI detector with the same procedure of KCR. We measured the χ^2 distance of the histogram h_M of each image of the *Holidays1000* data set and its forged counterpart from the authentic medium reference $\widehat{H}_M^{(auth)}$ as in Sec. 12.2, thus producing the scattergram of Fig. 12.10. Again, two clusters corresponding to authentic (green diamonds) and forged (magenta triangles) images can be clearly distinguished. The curves of Fig. 12.11 have been cal-

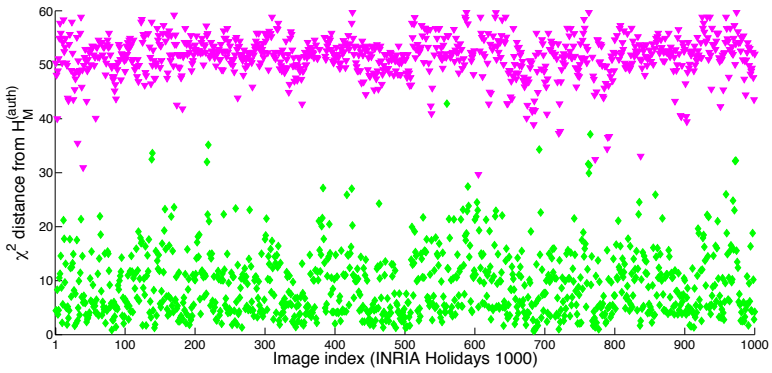


Figure 12.10: CHI *scattergram for the* Holidays1000 *data set. Green diamonds: authentic images; magenta triangles: forged images.*

culated on the *Copydays100* data set by relying on the reference histograms obtained from the *Holidays1000* (Fig. 12.3) and by varying the threshold T_2 in $[0, 80]$ and the target removal rate in $[10\%, 100\%]$. We show two ROCs: the one on the top right is based on the χ^2 distance from $\widehat{H}_M^{(auth)}$ and the one on the bottom left on the χ^2 distance from the manipulated medium reference

$\widehat{H}_M^{(forged)}$; both solutions are equally viable.

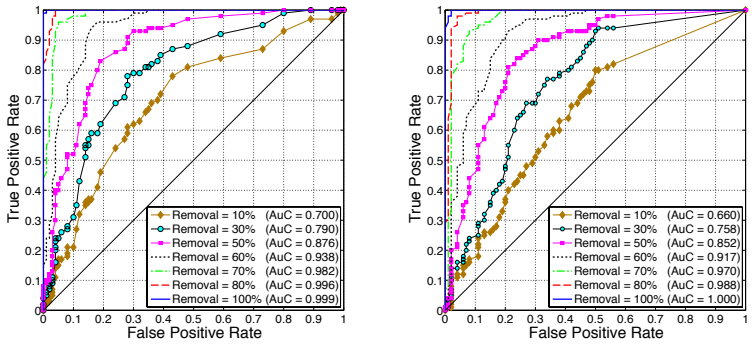


Figure 12.11: *ROC curves of* CHI. *Left:* χ^2 *distance from* $\widehat{H}_M^{(auth)}$; *right:* χ^2 *distance from* $\widehat{H}_M^{(forged)}$.

(C) Performance of the SVM detector

The SVM detector was implemented by relying on the well-established LIBSVM software [Chang and Lin, 2011]. The training stage has been carried out on $2,000$ feature vectors $\mathbf{F}$, i.e. those extracted from the *Holidays1000* data set and its CLBA-forged version. Recall that each feature vector consists of 30 elements, i.e. the 10 bins for *low*, *medium* and *high* variance histograms. The tests have been carried out on the 200 feature vectors coming from the *Copydays100* data set. We used a Radial Basis Function kernel with parameters $C = 8$ and $\gamma = 0.5$ derived from a 5-fold cross-validation on 400 images, which were not considered again for training to prevent over-fitting. The SVM model is probabilistic, i.e. it outputs the probability that the image under analysis is tampered. We calculated each ROC curve of Fig. 12.12 by varying T_3 in $[0, 1]$ and the target removal rate as in the previous tests.

To conclude this set of experiments, in Tab. 12.3 and 12.4 we compare the detectors in terms of AUC and true positive rate (fixed false positive rate of 0.1), as a function of keypoint removal rate; noticeable differences in performance exist only for removal rates below 50%, for which the KCR detector appears to be the most reliable. Concerning higher removal rates,

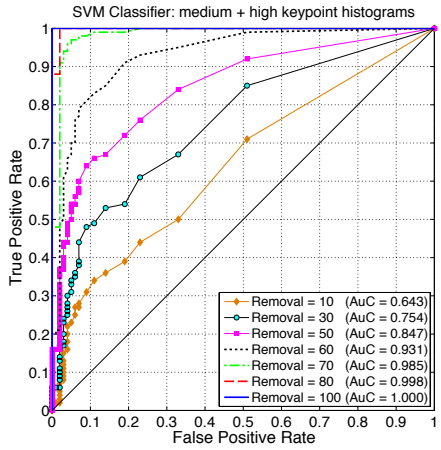


Figure 12.12: *ROC curves of* `SVM` *detector depending on the percentage of removed keypoints.*

the `SVM` detectors appears to be slightly more accurate than `KCR` (+1–5% true positive rate) but the latter detector does not require a learning stage.

	Keypoint Removal Rate						
Detector	**100%**	**80%**	**70%**	**60%**	**50%**	**30%**	**10%**
`KCR`	1	0.98	0.98	**0.94**	**0.92**	**0.85**	**0.77**
`CHI`	0.99	0.99	0.98	0.93	0.87	0.79	0.7
`SVM`	1	0.99	0.98	0.93	0.85	0.75	0.63

Table 12.3: *AUC for the detectors as a function of removal rate.*

12.4.5 Dependence on image size and removal rate

To assign the threshold values T_1, T_2 and T_3, we took into account two factors: the target removal rate and the image size. Both parameters, in fact, impact negatively on the performance of the detectors. Low removal rates cause the forged image to be similar to the authentic one; conversely, reducing image size also reduces keypoints, thus making the authentic image appear as if it were forged.

	Keypoint Removal Rate						
Detector	**100%**	**80%**	**70%**	**60%**	**50%**	**30%**	**10%**
KCR	0.99	0.98	0.94	0.87	**0.86**	**0.75**	**0.6**
CHI	1	0.99	0.97	0.78	0.52	0.31	0.21
SVM	1	**1**	**0.99**	**0.92**	0.66	0.48	0.33

Table 12.4: *True positive rate for the detectors for a false positive rate* 0.1 *as a function of removal rate*

In the following experiment we used the *Copydays100* data set to build 4 new sets by progressively downscaling the images to 1600×1200, 1200×900, 800×600 and 400×300. Then, we randomly subdivided each data set in 5 sets of 20 images which have been attacked with CLBA with removal rate $\{50, 60, 70, 80, 100\}$, respectively. From the ROC curves of Fig. 12.13 we can conclude that there is not a strong dependence of the performances of KCR and SVM on the image size, as opposed to CHI, for which the detection is easier on larger images.

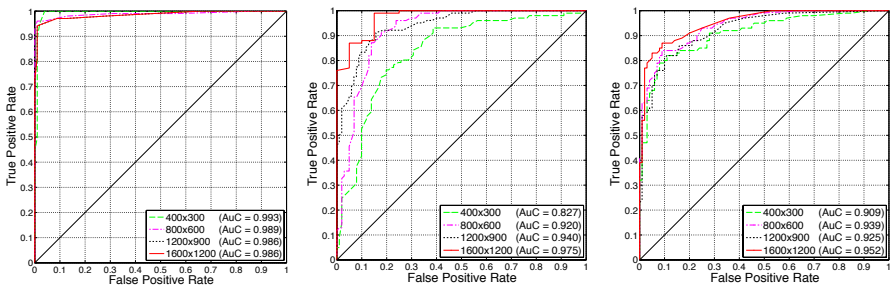


Figure 12.13: *ROC for mixed removal rates depending on image size. From left to right:* KCR, CHI *and* SVM.

To determine the thresholds, we fixed a maximum value of 0.1 for the false positive rate and we gathered the corresponding value for the true positive rate from each curve. Finally, we retrieved the threshold responsible for that specific point in the ROC curves. The results summarised in Tab. 12.5 show that the KCR threshold is stable for the various sizes, whereas the CHI and SVM

thresholds require more tweaking.

	Image size			
Detector	**400 × 300**	**800 × 600**	**1200 × 900**	**1600 × 1200**
KCR	−1.8	−1.9	−1.9	−1.9
CHI	48	32	22	20
SVM	0.41	0.42	0.58	0.67

Table 12.5: *Thresholds vs image size for a probability of false alarm* 0.1

12.4.6 Detection of local keypoint removal

We now take into consideration the case in which keypoints are removed from a specific area of the image. To reveal a local forgery, we apply the detectors in a block-wise fashion: each 32×32 non-overlapping block of the image is processed by analysing the statistics of a larger square region surrounding it. In fact, a statistical analysis of a small block would not be meaningful enough; consequently, on the one hand the analysed region should be large enough, while on the other hand it should be small enough so that its characteristics are representative of the to-be-classified block. Following the experiments of Sec. 12.4.5, we opted for a 600×600 region. In practice, for each block we run the detectors on a 600×600 area, we compute a soft value describing the degree of tampering and we assign it to the inner 32×32 block. Then, we shift by 32 pixels and we repeat the procedure. This procedure provides a soft-valued map roughly localising areas artificially deprived of SIFT keypoints, which is binarised by applying the detector's threshold. The map is finally cleaned by removing those regions whose area is smaller than 2% of the total image area.

Fig. 12.14 displays an example of local removal detection by means of KCR. The keypoints of an authentic image were removed from the framed region (top left), with decreasing removal rate of 100%, 80%, 60%, 40% and 20%. As expected, detection becomes more difficult as the removal rate lowers; nevertheless, even in the case of the lowest removal rate, the detector is still able to correctly localise 39% of the manipulated area.

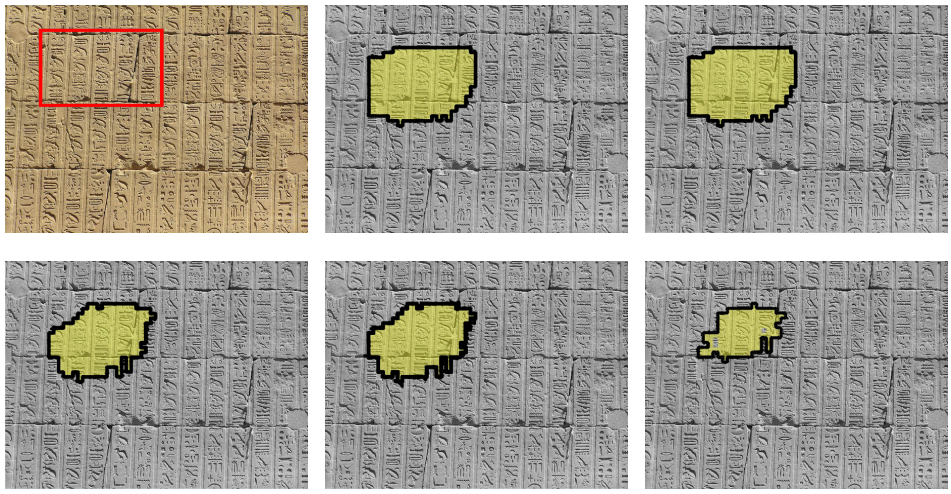


Figure 12.14: *Example of local removal. From top left to bottom right: authentic image;* 100%*,* 80%*,* 60%*,* 40% *and* 20% *removal rate.*

One issue with the windowed approach is that blocks surrounded by regions naturally poor of keypoints, such as in the sky or sea areas, are erroneously considered tampered. Our solution conveniently resorts to the binarised variance map V: first, very low blocks (variance $\leqslant 0.1$) are set to 0; then, morphological flood-fill and area opening are used to remove holes and isolated pixels. The actual tampering map is obtained by binary AND-ing V and the localisation map provided by the removal detector. Two examples of this procedure are shown in Fig. 12.15. A few additional examples are shown in Fig. 12.16, where the leftmost image of each row is the forgery, the middle image is the mask helping localise the manipulated regions and the rightmost image is the output of the `KCR`. In the first image, we hid a fish by duplicating the bottom of the sea; in the second image, we mirrored a bunch of Sistine's Chapel saints on the right and we duplicated them on the left.

Concerning the time complexity of the approach, localisation is computationally intense, since it requires to run the detector on several thousands of blocks; for example, it takes about 9 minutes on a 64 bit OS with 8 GB RAM to process the 1333×2000 image of Fig. 12.14 (top).

(a) (b) (c) (d)

Figure 12.15: *Local detection in presence of authentic regions with few or no keypoints. (a)-(c): keypoints following* `CLBA`*; (b)-(d): masking avoids false positives in the sea and in the sky.*

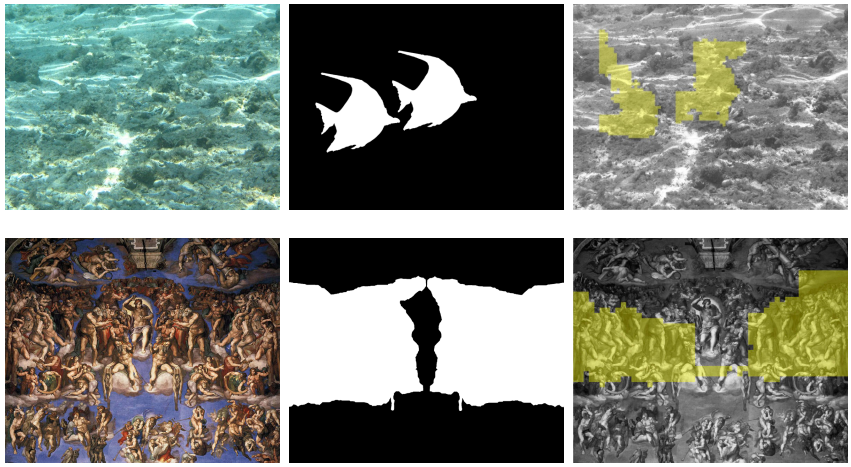


Figure 12.16: *Examples of local removal detection by means of* `KCR`*. First column: forged image; second colum: tampering mask; third column: detection.*

12.5 Additional remarks on `KCR` assumptions

In Sec. 12.1 we assumed that SIFT keypoints lie in proximity of corners and that keypoint removal does not affect significantly corner detection. Based on such assumptions, which we validated experimentally, we were able to devise the most performing of the keypoint removal detectors we proposed in this chapter, i.e. the `KCR` detector. In this section we investigate more in-depth the above two assumptions and, in particular, we discuss the behaviour of the

KCR in those cases where they do not hold.

In general, the first assumption holds regardless of the employed corner extractor, as long as the content of the image under analysis conveys relevant information. However, problems could arise with images characterised by low-variance flat content, such as for example night pictures of the moon, water surfaces, sand or the sky. This kind of content yields few keypoints and few corners even before CLBA is carried out, and thus the *KCR* index of the authentic image may be as low as that of its CLBA-forged version. To verify whether this fact actually affects the performance of KCR, we gathered 25 images representing the sea, the desert and the sky. One image for each category is displayed in Fig. 12.17.

Figure 12.17: *Examples of the images with few keypoints and corners.*

We collected the *KCR* indexes of all images before and after CLBA (100% removal rate) and we calculated the ROC curve of Fig. 12.18. We can clearly see that the KCR does not seem to be particularly influenced by the characteristics of the chosen images. In particular, setting the KCR threshold to its default value $T_1 = -1.9$ ensures true positive rate 0.96 and false positive rate 0.2. Among the pictures of Fig. 12.17, only the rightmost one is not recognised as forged: the *KCR* index, in fact, is -1.12 before CLBA and -1.21 after.

Let us now consider the case where the second assumption is not respected. This could be the case of a careless counterfeiter who resorts to too perceptible attacks or of a wise counterfeiter trying to counter the KCR by artificially raising the indexes of forged images. Recall that the *KCR* index is computed as:

$$KCR = \log_{10}\left(\frac{N_{keypoints}}{N_{corners}}\right) \leqslant T_1. \tag{12.7}$$

The goal of the counterfeiter, then, is to reduce both $N_{keypoints}$ and $N_{corners}$ so

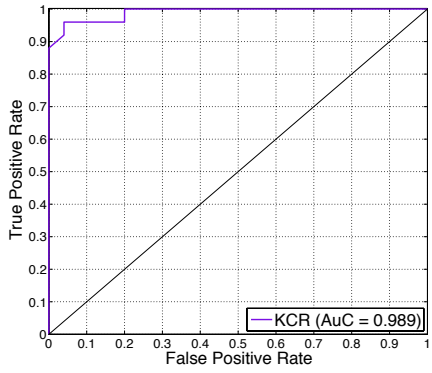


Figure 12.18: *ROC for authentic images with few keypoints and corners.*

to keep their ratio below T_1 throughout the manipulation. In our experiments, we resorted to two attacks manipulating the neighbourhood of each keypoint in such a way to remove the keypoint and as much nearby corners as possible.

- `CLBA20`: the strongest version of `CLBA` based on a support of size 20, which ensures the highest keypoint removal rates for all octaves at the cost of the lowest resulting quality (see Tabs. 10.6–10.7).

- `BlackPatch10`: an attack setting to 0 all the pixels in the 10×10 neighbourhood of each to-be-removed keypoint. Given its extreme degradation of the image, such an attack is obviously not usable in practice but it allows to conceptually simulate an effective strategy to remove corners.

We attacked the first 100 images of the *Holidays1000* data set with the above attacks; as shown in Fig. 12.19, both of them considerably reduced the percentage of corners, given that the average reduction is respectively -14.1% for `CLBA20` and -25% for `BlackPatch10`. Both these values are far superior to the corner reduction caused by the `CLBA` with support 10 (i.e. -3%) employed throughout the chapter. Regardless of this fact, the attacks do not succeed in keeping the *KCR* indexes above T_1 nor in raising them significantly, as proved by the scattergrams in Fig. 12.20. Note that the indexes corresponding to the `BlackPatch10` attack are slightly higher than those of `CLBA 20`.

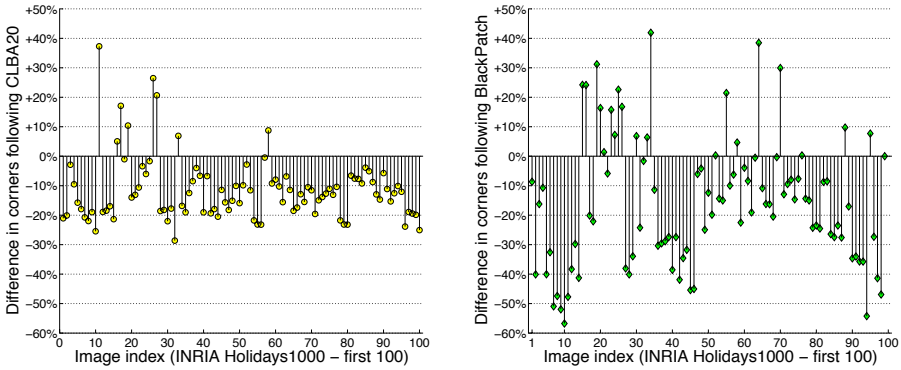


Figure 12.19: *Difference in the number of FAST corners (%) following* `CLBA20` *(yellow circles, left) and* `BlackPatch10` *(green diamonds, right).*

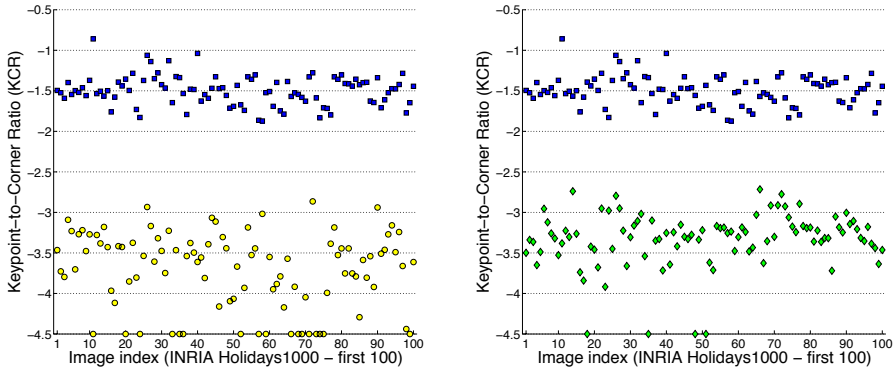


Figure 12.20: *KCR scattergram following* `CLBA20` *(yellow circles, left) and* `BlackPatch10` *(green diamonds, right). Blue square markers correspond to authentic images.*

In conclusion, deleting corners seems to be an unlikely strategy to impair the `KCR` detector due to problems of perceptibility and effectiveness. The percentage of corners that need to be deleted to keep the *KCR* index of forged images above the authenticity threshold is too high and consequently the quality of the image following the attack is seriously compromised. Instead, we

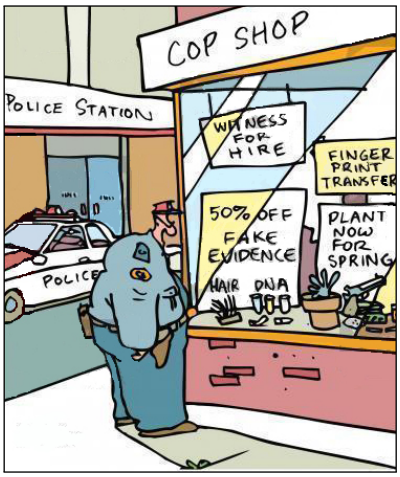
will see in the next chapter that a sound strategy to artificially raise the *KCR* index consists in inserting spurious SIFT keypoints (not matching with the originals and not interfering with the preceding removal) into a `CLBA`-forged image.

12.6 Concluding remarks

In this chapter we discussed three different forensic detectors revealing global and local keypoint removal attacks and we experimentally validated them. We observed that there exists a trade-off between the capability of detection and the removal rate of the attacks: the less keypoints are removed from an image, the more difficult the detection. As a consequence, if the manipulation has to be undetectable, the adversary is either forced to remove only a limited amount of keypoints or to devise new counter-forensic methods against the removal detectors. Such is the aim of the next chapter, where we will address the dual problem of keypoint removal, i.e. keypoint insertion. Our goal is to repopulate an image with fake keypoints that: i) are detected by the SIFT DoG; ii) cause the failure of removal detectors; and iii) have different properties than authentic keypoints to preserve the results attained by the removal stage.

Chapter 13

SIFT keypoint injection



From the previous chapter it emerged that under some conditions the detection of keypoint removal is possible. In this chapter the counterfeiter attempts to bypass the newly devised forensic detectors by forging false keypoints to repopulate previously depleted areas. Firstly, the reasons why keypoints should be re-inserted (or "injected") are explained in Secs. 13.1 and 13.2, where a general framework for this task is also proposed. Then, a number of original techniques are presented in Sec. 13.3, combined into a new and more efficient injection attack in Sec. 13.4 and compared against each other on a large data set in Sec. 13.5. In the same section we also try to counter the keypoint removal detectors of Chapter 12 by means of the most performing injection attack. Sec. 13.7 concludes the chapter by considering again the SIFT-based copy-move detection scenario, where we apply all the tools developed so far.

13.1 Motivations

We know from Chapter 12 that keypoint removal leaves traces of its application into a manipulated image. Such traces consist in the anomalous distribution of keypoints in semantically relevant image regions or in proximity of certain points of interest (corners), near which keypoints should be found but are instead absent. Based on such footprints, we were able to devise keypoint removal detectors exposing the work of the counterfeiter.

Arguably, the most straightforward strategy to bypass keypoint removal detection consists in introducing *fake* keypoints in suitably chosen positions. If enough of such keypoints are reintroduced, then the discriminative power of the forensic tools of Chapter 12 may be compromised. Obviously, to avoid invalidating what has been accomplished with the removal stage, some constraints must be satisfied: first of all, new keypoints should pass all SIFT checks and thus be detectable; secondly, they should not have the same descriptors of their authentic counterparts.

So far the re-introduction (or "*injection*") of SIFT keypoints did not attract the same attention that has been dedicated to their removal. In early works, injection has been seen more as a side effect of removal rather than as a potentially useful tool. Consequently, although briefly analysed in the CBIR context in [Do et al., 2010a], the topic has not been systematically studied yet: in this chapter we push the research towards this direction. More specifically, the chapter further develops the ideas underlying the preliminary studies in [Amerini et al., 2013c]. With respect to such a work, we compare existing and newly devised techniques against each other on a large data set of images. Moreover, we combine the most effective among such methods to obtain a new attack outperforming each single technique and we assess its negative impact on the performance of keypoint removal detectors. Finally, we apply all the tools to a practical image forensic scenario of SIFT-based copy-move forgery detection, where we let the counterfeiter resort to keypoint removal and injection to create copy-move forgeries that successfully elude SIFT-based detectors but are in turn exposed by the keypoint removal detectors.

13.2 Keypoint injection

We evaluated five possible attacks to introduce fake keypoints. Three of them in particular are based on adaptive image enhancement algorithms already known in the literature but never used before in the context of SIFT countering. The rationale behind repurposing locally adaptive image enhancement for keypoint injection is the following. It is known that smoothing techniques perform well in removal since they reduce image details; therefore, we may argue that enhancement techniques, which in turn exalt details, should conversely introduce new keypoints. Furthermore, because traditional techniques applied to the whole image (e.g. sharpening, global contrast enhancement) generate visually unpleasant images, more sophisticated solutions should be employed, so that the resulting quality is comparable to that of the authentic image. The experimental results obtained later in this chapter confirm both intuitions. In addition to the above tools, we also considered the dual versions of keypoint removal methods already known in literature, i.e. Gaussian `Smoothing` and `FMD` [Do et al., 2010a]. Regardless of the approach, one of the advantages brought by the above methods is that it is possible to combine them to achieve better results in terms of injected keypoints, as we prove by devising a sixth method, called Classification-Based Injection (`CLBI`).

13.2.1 Injection framework

The framework we will use to evaluate the capability of injection is the same for all the methods. Therefore, for the sake of clarity we introduce its working principles here in accordance with the block diagram of Fig. 13.1.

The injection attack is applied to an image I_{rem} whose keypoints have been removed by means of a certain attack, for example $I_{rem} = \texttt{CLBA}(I)$. Unlike removal, the first stage of keypoint injection is not iterative nor local, since I_{rem} is processed full-frame with the chosen injection algorithm, producing the injected image I_{inj}. Although I_{inj} already contains new keypoints, two facts suggest avoiding to use it as it is: the overall visual quality of the image is degraded by full-frame processing and some injected keypoints may match with their homologues in the authentic image I.

The first problem is tackled with by extracting the $M \times M$ neighbourhood

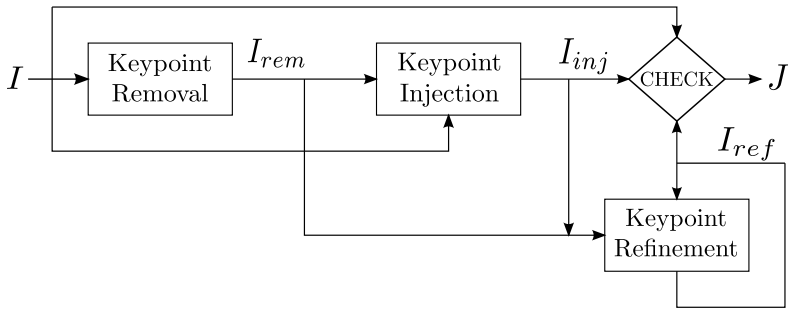


Figure 13.1: *Work-flow of the injection framework.*

of each new keypoint and inserting it back into I_{rem}. Artefacts along the edges of the pasted region can be hidden by means of a linear combination of I and I_{rem} in analogy with how `CLBA` works in Eq. (10.8).

The second problem requires to find and suppress those new keypoints of I_{inj} that are *correctly* matching with their homologues in I. In practice, a match is considered correct if the distance between old and new descriptors falls below a certain threshold. When one of such matches is found, the keypoint of I_{inj} is discarded by restoring the $M \times M$ region of I_{rem} preceding the injection. The idea is to primarily avoid a SIFT match at the expense of the loss of an injected keypoint. Ideally, the resulting image J has a plausible amount of forged keypoints, that are not distinguishable from authentic keypoints. Interestingly, it is not so crucial to check whether a fake keypoint is or is not in the same spatial position of an authentic one before removal, as long as their descriptors do not match.

13.3 Injection algorithms

The attacks used to forge fake keypoints rely on five different algorithms, which are briefly described in this section.

13.3.1 Contrast Limited Adaptive Histogram Equalisation

Global contrast enhancement techniques assume that the distribution of gray scale pixel values is uniform over all the areas of an image. When this assumption does not hold, performances of global methods are poor and the enhanced images are visually unpleasant. `CLAHE`'s (Contrast Limited Adaptive Histogram Equalisation) [Zuiderveld, 1994] solution to this problem is twofold: it adapts to the local properties of the regions of an image and it limits the contrast differences across them. In a nutshell, the algorithm proceeds as follows (see [Zuiderveld, 1994] for details). First, the $M \times N$ grayscale image I is divided into non-overlapping tiles and the histogram of each tile is computed. Then, a clipping limit β for the contrast enhancement is obtained:

$$\beta = \frac{MN}{L}\left(1 + \frac{\alpha}{100}(s_{max} - 1)\right), \tag{13.1}$$

where L is the number of histogram bins, $\alpha \geqslant 0$ is the *clipping factor* and s_{max} is the slope of the transfer function mapping the contrast from its input value to its output value; if $s_{max} = 1$, then no enhancement is performed, while larger values (usually up to 4) will result into more visible enhancements. Next, each histogram is clipped in such a way that its height is limited by β. At this point, it is necessary to remap the clipped values to the entire intensity range, that is to re-normalise the histogram of the processed image to its original area. This task can be carried out in several ways, the most common of which consists in redistributing the clipped pixels uniformly in all the bins of the histogram of the whole image.

13.3.2 Brightness Preserving Fuzzy Histogram Equalisation

`BPDFHE` (Brightness Preserving Dynamic Fuzzy Histogram Equalisation) is a method to enhance the contrast of an image while preserving its mean brightness, and thus the perceived subjective quality [Sheet et al., 2010]. Similarly

to other contrast enhancement techniques, it proposes to divide the image histogram into segments,[1] which are then independently equalised. Partitioning, however, is not performed on the normal histogram but rather on its fuzzy counterpart, whereby a pixel may belong to some degree to more than one of the bins in accordance with a fuzzy membership function. Such a histogram, in fact, is generally smoother and has no missing levels or abrupt fluctuations, thus allowing a more accurate segmentation. The pixel membership functions can be designed in different fashions depending on the application. In their paper, Sheet *et al.* use Triangular and Gaussian membership functions. The algorithm proceeds as follows:

1. the fuzzy histogram $\widetilde{H}(k)$, $k = [0, L]$ is computed by assigning to each bin k the number of pixels whose value is *around* k, in accordance with the chosen membership function;

2. the local maxima $\{m_1, m_2, \ldots, m_n\}$ are computed and used to define the segments of the histogram: $S = \{\, [0, m_1 - 1], [m_1, m_2 - 1], \ldots, [m_n, L] \,\}$;

3. each segment is equalised by means of a technique depending on the number of pixels belonging to the partition;

4. in order to cope with the alterations that may have been introduced, the resulting brightness is normalised to match the original brightness.

13.3.3 Anisotropic Diffusion

Anisotropic Diffusion (`2D-AD`) is a method to enhance images by preserving the perceptual quality of semantically relevant parts like straight lines, edges and geometric shapes [Perona and Malik, 1990]. In principle, it is a generalisation of the scale-space transform whereby an image I is iteratively convolved with a nonlinear smoothing filter, which is adapted to the local content to generate progressively more blurred versions of I (theoretical details are left to [Weickert, 1999] and [Weickert and Scharr, 2002]). In other words, the key-idea behind `2D-AD` is to keep image structures intact and smooth only the area around them.

[1] The idea is not dissimilar from the one behind the classification method on which the `CLBA` attack is based.

The filter model allowing anisotropic diffusion is derived from well-known operators used to extract image details. Let $I_\sigma = I * G_\sigma$ be the convolution of an image I with a Gaussian kernel ($\sigma > 0$); then, the gradient ∇I_σ can be employed to highlight structures like the edges of I. Since the gradient does not always perform satisfactorily, a more effective operator is derived from it.

Let $J(\nabla I_\sigma) = \nabla I_\sigma \nabla I_\sigma^T$; then, $J_\rho(\nabla I_\sigma) = J(\nabla I_\sigma) * G_\rho$, that is the convolution with a Gaussian kernel ($\rho > \sigma$), is called *tensor operator* and it can be used to effectively highlight flow-like, T-shaped or Y-shaped structures [Weickert, 1999]. The eigenvectors $\{w_1, w_2\}$ of J_ρ give indications on local orientations and the corresponding eigenvalues (μ_1, μ_2) on the local contrast along these directions. The 2×2 *diffusion tensor* D performing anisotropic diffusion is created in such a way that its eigenvectors are the same of J_ρ and its eigenvalues (λ_1, λ_2) are:

$$\begin{aligned} \lambda_1 &= c_1 \\ \lambda_2 &= \begin{cases} c_1 & \text{if } \mu_1 = \mu_2 \\ c_1 + (1 - c_1)\exp\left[\dfrac{-c_2}{(\mu_1 - \mu_2)^2}\right] & \text{otherwise,} \end{cases} \end{aligned} \tag{13.2}$$

where $c_1 \in (0, 1)$ and $c_2 > 0$. The elements of D are derived from (λ_1, λ_2). The result is an operator that can steer the enhancement according to the direction of flow-like structures in the image. By resorting to D, it is possible to efficiently compute blurred versions of $I(x, t)$ as numerical solutions of Eq. (13.3), where $t \geqslant 0$ is called *diffusion time*:

$$\frac{\partial I}{\partial t} = \nabla \cdot (D \nabla I). \tag{13.3}$$

In practice, the algorithm proceeds as follows: given $I = I(x, 0)$, first $J_\rho(\nabla I_\sigma)$ is computed and D is derived with Eq. (13.2); next, $I(x, 1)$ is obtained with Eq. (13.3). Starting from $I(x, 1)$, the process is repeated until a specified number of iterations t_{max} is reached.

13.3.4 Gaussian Smoothing

Even though Gaussian `Smoothing` is primarily used to remove keypoints, it can also inject keypoints if the filter's standard deviation σ_f is large enough. To highlight this particular behaviour and to determine a suitable value for σ_f, we ran a test that is similar to the one we set up in Sec. 10.3.4 for keypoint removal: we collected 100 random `CLBA`-forged images from the UCID data set and we injected them with Gaussian `Smoothing` with increasing σ_f in the interval $[1, 3]$, then we computed the percentage of injected keypoints with respect to the number of authentic ones.[2] Fig. 13.2 shows the average percentage as a function of σ_f; based on this trend, we set $\sigma_f = 2$ since the small gain granted by higher values does not justify additional quality loss.

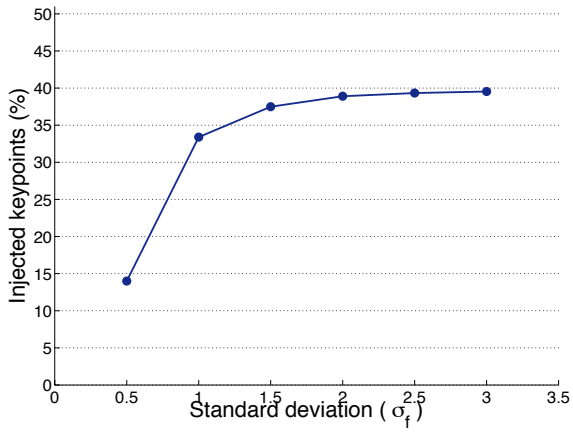


Figure 13.2: *Parameter tuning of the* `Smoothing` *attack (injection): average percentage of injected keypoints as a function of standard deviation* σ_f.

[2] Since we are evaluating the raw injection power of the algorithm, we do not worry about whether the new keypoints match or not with their original homologues. When we apply match refinement, in fact, less keypoints are effectively injected (see Sec. 13.5.2, Fig. 13.5).

13.3.5 Forging with Minimum Distortion

Forging with Minimum Distortion (`FMD`) is the symmetrical version of the `RMD` attack used to remove keypoints in [Do et al., 2010a]. Given a location (x, y) and a scale σ at which a keypoint should be introduced, this time the patch ϵ that needs to be added to the neighbourhood of (x, y) is computed by solving the following problem:

$$\epsilon = \underset{\epsilon: D'(\mathbf{x}) = D(\mathbf{x}) - \delta}{\operatorname{argmin}} \frac{1}{2}||\epsilon||^2. \tag{13.4}$$

The parameter δ still controls the intensity of the attack: $D(x, y, \sigma)$ is raised by $|\delta|$ in such a way that the altered value $D'(x, y, \sigma)$ in now greater than the contrast threshold. The advantage of the `FMD` with respect to the use of the rest of the injection methods is that it allows to choose the coordinates at which a keypoint should be injected. Consequently, with the `FMD` we can define suitable locations wherein to inject the keypoints, while with the rest of the tools keypoints are injected randomly and the appropriateness of their positions verified afterwards. On the other hand, the `FMD` strongly affects the quality of the forgery and thus it should be used sparingly.

13.4 Classification-based injection attack

During the injection stage we need to pay attention to the perceptibility of the result, especially because now we are working on images that have been already manipulated by removing keypoints at the expense of visual quality. Expectedly, the number of injected fake keypoints is inversely proportional to the quality of the forged image and thus once again we have to find a balance between effectiveness and imperceptibility. To do so, we chose to follow the same approach of the `CLBA`, that is a classification-driven procedure exploiting the advantages of each single algorithm while limiting its weaknesses. For this reason we refer to it as Classification-Based Injection (`CLBI`).

The new attack relies on `FMD` and on three image enhancers, i.e. Gaussian `Smoothing`, `2D-AD` and `BPDFHE`. Intuitively, `CLBI` forges keypoints with `FMD` where they *must* be found (i.e. in proximity of corners and edges) and with the three enhancers where they *should* be found (i.e. in textured non-flat

regions). Therefore, the classification task basically consists in distinguishing between regions containing edges and corners and the rest of the image.

The general scheme of the proposed attack is the same of Fig. 13.1. `CLBI` takes as input an image I_{rem} without keypoints and produces the injected image J in four steps as shown by the colored blocks in Fig. 13.3: i) classification of I_{rem}'s regions (red); ii) `FMD` injection (green); iii) contrast-enhancement injection (orange); iv) refinement of matches (cyan).

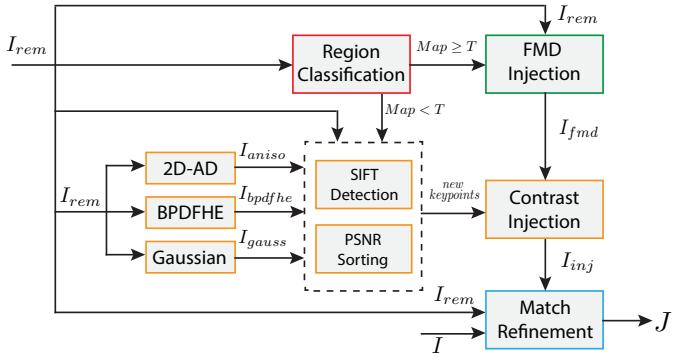


Figure 13.3: `CLBI` *injection attack (best viewed in color).*

13.4.1 Classification of image regions

To distinguish image regions according to the structures they contain, we chose the tensor operator [Weickert and Scharr, 2002] of I_{rem}, which produces a map of the same size of the image highlighting edges and flow-like, T-shaped and Y-shaped structures. By normalising the output of the operator in $[0, 1]$, we obtain the injection map Map wherein each element (x, y) quantifies the saliency of the corresponding pixel. The idea is to use the most salient points such that $Map(x, y) \geqslant 0.5$ as preferred injection locations for the `FMD`, while the remaining less descriptive points (whose score is however not null) are left for the image enhancers:

$$\text{if } \begin{cases} 0.5 \leqslant Map(x,y) \leqslant 1, & (x,y) \text{ salient} \\ 0 < Map(x,y) < 0.5, & (x,y) \text{ not salient.} \end{cases} \tag{13.5}$$

13.4.2 `FMD` injection

The coordinates of all the pixels belonging to the salient regions as well as the input image I_{rem} are fed to the first injection block, where the `FMD` attempts to introduce a keypoint in each location. The large amount of candidate locations is a direct consequence of `FMD`'s characteristics: the artefacts it introduces are visually unacceptable if the attack is too intense, thus forcing us to set its strength δ to the minimum (that is 2); by doing so, however, the attack can not ensure a positive outcome and for this reason we repeat the injection for all the locations; following each attempt, we run a SIFT check to verify whether a new keypoint has been introduced. If this is the case, we do not further alter its 8×8 neighbourhood to avoid undoing the forgery. This stage produces a first version I_{fmd} of the injected image.

Algorithm 3: *Pseudo-code of the first stage of injection.*

```
I_fmd ← I_rem;
Map ← computeMAP(I_fmd);
foreach Map(x, y) ⩾ 0.5 do
    I_fmd ← FMD(I_fmd, x, y);
end
```

13.4.3 Contrast-enhancement injection

A copy of the input image I_{rem} is fed to each enhancer to produce the three partially injected images referred to as I_{aniso}, I_{bpdfhe} and I_{gauss} in Fig. 13.3. All the 8×8 neighbourhoods of the injected keypoints of these images are extracted and ordered according to their local PSNR with respect to I_{rem}. If more than one enhancer has created a keypoint in the same location, we keep only the one with highest PSNR. The goal of this procedure is to generate as much keypoints of acceptable quality as possible to repopulate the more uniform (but still significant) regions of I_{fmd}, which were not touched by the previous stage of the attack; therefore, keypoints passing this selection are pasted with their neighbourhoods into I_{fmd} only if $0 < Map(x, y) < 0.5$, producing the injected image I_{inj}.

Algorithm 4: *Pseudo-code of the second stage of injection.*

$I_{aniso} \leftarrow$ `2D-AD`(I_{rem});
$I_{bpdfhe} \leftarrow$ `BPDFHE`(I_{rem});
$I_{gauss} \leftarrow$ `smoothing`(I_{rem});
keypoints $\leftarrow$ `calculateSIFT`$(I_{aniso}, I_{bpdfhe}, I_{gauss})$;
keypoints $\leftarrow$ `PSNRsort`(keypoints, I_{rem});
foreach *keypoint* $= (x_{kp}, y_{kp})$ **in** keypoints **do**
  if $0 < Map(x_{kp}, y_{kp}) < 0.5$ **then**
    $I_{inj} \leftarrow$ `replaceNeighbourhood`$(I_{inj}, x_{kp}, y_{kp})$;
  end
end

13.4.4 Match refinement

Finally, we check that the keypoints of I_{inj} do not match with their counterparts in the authentic image. We consider a match correct if the distance between the old and new descriptors falls below a certain threshold (8 in our case). This final stage addresses two types of matches at the same time, i.e. those accidentally created by the injection procedure and those left because of non-perfect removal. We suppress the former by restoring the corresponding neighbourhood from I_{rem}. If such neighbourhood already contains keypoints, then they are suppressed by applying the Removal with Minimum Distortion (`RMD`) attack [Do et al., 2010a]; in fact, `RMD` is very effective when its strength δ is increased at least to 3 (against a default of 2), although this tends to introduce salt and pepper noise.

The output of this stage is the final refined image J. It is worth noting that, despite all the controls, J may still have some correct matches, mainly due to two factors: the changes in pixel values while pasting overlapping neighbourhoods and the extreme robustness of some keypoints even to the strengthened `RMD`. The presence of such matches, however, is not a major drawback for several reasons. First of all, their number is very small if compared to that of the authentic ones (see Sec. 13.5.2 and Fig. 13.6 in particular); secondly, whether surviving matches constitute a problem or not ultimately depends on the countered application.

13.5 Experimental validation of keypoint injection

13.5.1 Experimental setup

The injection capability is measured with the Keypoint Injection Rate (KIR) in a symmetrical manner with respect to removal:

$$KIR = \left(\frac{new\ keypoints\ following\ injection}{keypoints\ before\ removal} \right) \times 100. \tag{13.6}$$

A second important metric is the number of fake keypoints correctly matching with the authentic ones (the less the better, obviously). We consider a SIFT match correct if it links two keypoints located in the same spatial position or, at most, within a 8×8 neighbourhood.

In order to keep the complexity of the experiments under control,[3] we considered smaller images, i.e. those composing the UCID data set. The performance of keypoint removal detection on this data set (Fig. 13.4) are in line with those we obtained with the downscaled INRIA Holidays images. Concerning the experimental setup, keypoints are detected by means of VLFeat (edge and peak thresholds set to 10 and 4); matching is performed with nearest neighbour (threshold fixed to 0.8), as suggested in [Lowe, 2004] and with k-d trees [Bentley, 1975]. Given the similarity of the results, we discuss only the former matching strategy. The parameters for the removal stage are the same of Sec. 10.4.2; the size of the injection support is set to $M = 8$. The specific parameters of each technique have been set to the values indicated in the reference papers.

13.5.2 Effectiveness of keypoint injection

All attacks are evaluated on the UCID data set, whose $1,338$ images have been previously attacked with `CLBA`. Fig. 13.5 shows the KIR histogram envelopes for the six attacks. The performance of most of the single methods are quite similar, with Gaussian `Smoothing` (average $KIR = 27.9\%$), `2D-AD` (23.4%)

[3] On average, our Matlab implementation takes up to 180 seconds on a 400×400 image, due to the several thousands of iterations required by the `FMD`, each of which also includes SIFT detection (64 bit OS, 8 GB RAM).

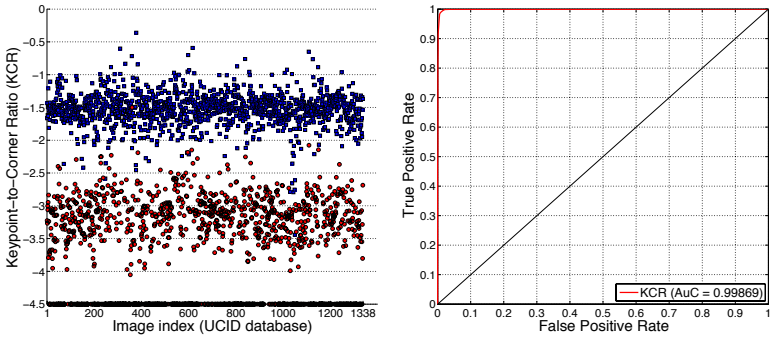


Figure 13.4: `KCR` *scattergram and ROC on the UCID data set.*

and `FMD` (17%) being superior to `CLAHE` (14%) and `BPDFHE` (either membership function, 11%). Two observations are in order here. First of all, the methods allow to introduce a larger number of keypoints with non-default parameters, if one is willing to trade-off with a reduced image quality. Secondly, attacks should be used in combinations, as confirmed by `CLBI`, which guarantees the best results ($KIR = 49.7\%$).

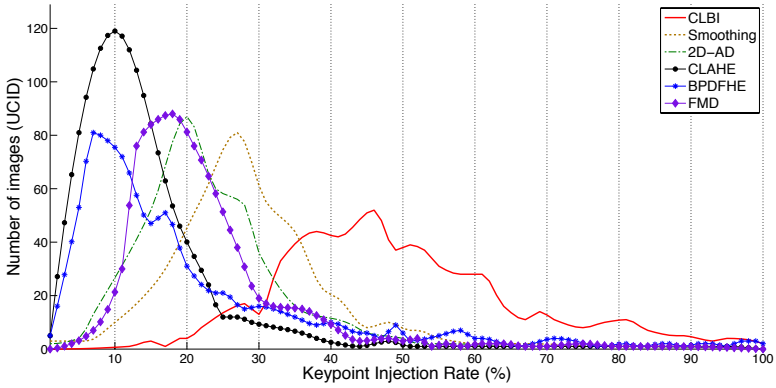


Figure 13.5: *KIR envelopes for the* `CLBI` *and class-unaware attacks.*

In Fig. 13.6 we show the cumulative distribution (with superposed normalised histogram) of the matches between authentic and forged images after removal (dash-dotted blue line) and after injection (solid red line). Following

refinement, the matches left because of non-perfect removal and those introduced accidentally during injection are effectively reduced, to the point that 61% of the images have less than 3 matches (versus 39% before injection); furthermore, the images without matches increased from 11% to 25% of the data set. Such results are satisfactory, especially if we consider that prior to the attack images have on average 232 matches. False positive matches are rarely introduced by `CLBI` (on average less than one per image), hinting that more specialised algorithms working on descriptors are required for the task.

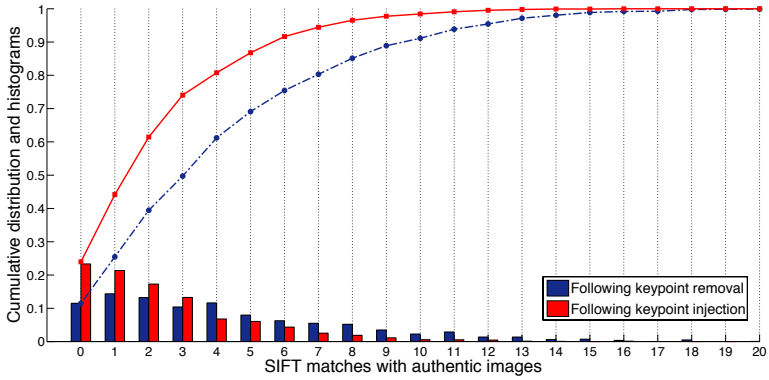


Figure 13.6: *Cumulative distribution (with superposed normalised histogram) of correct matches following removal and following injection.*

In Tab. 13.1 we give some results about the quality loss caused by removal and injection in terms of PSNR and SSIM, evaluated both globally (averaged on all the data set) and locally (averaged on all the attacked neighbourhoods). Both the visual quality with respect to the authentic image and to the outcome of removal are not significantly worsened by injection.

	Global quality		**Local quality**	
Stage	PSNR	SSIM	PSNR	SSIM
After removal	32.74	0.976	28.88	0.894
After injection	30.98	0.97	27.29	0.839

Table 13.1: *Quality of removal-injection: average PSNR (db) and SSIM on entire images and on the attacked neighbourhoods*

In conclusion, the proposed injection attack allows to reach a good compromise between percentage of removed keypoints, percentage of injected keypoints, correct matches accidentally left or introduced and perceptual quality of the forged image.

13.5.3 Examples of keypoint injection

A detailed example of the removal-injection procedure is given in Fig. 13.7. From left to right, the top row shows the keypoints of authentic (blue squares), `CLBA`-forged (perfect removal) and injected (red circles) images. The bottom row shows authentic image's matches with `CLBA`-forged (left) and injected (right) images. The following three aspects are worth attention.

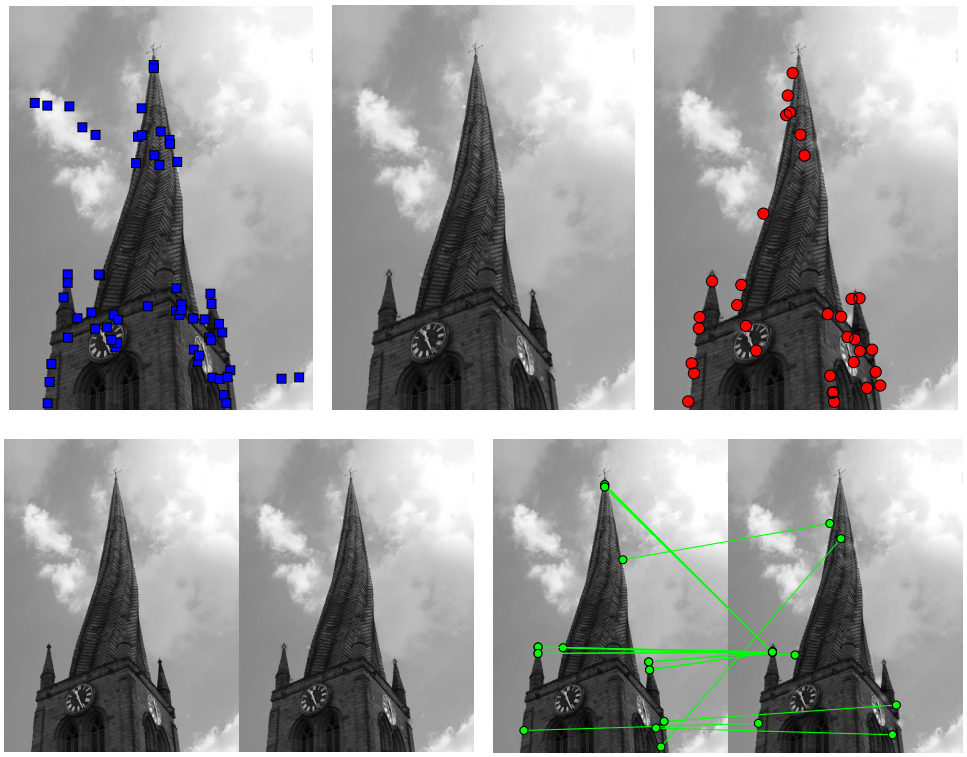


Figure 13.7: *Example of removal-injection procedure (i).*

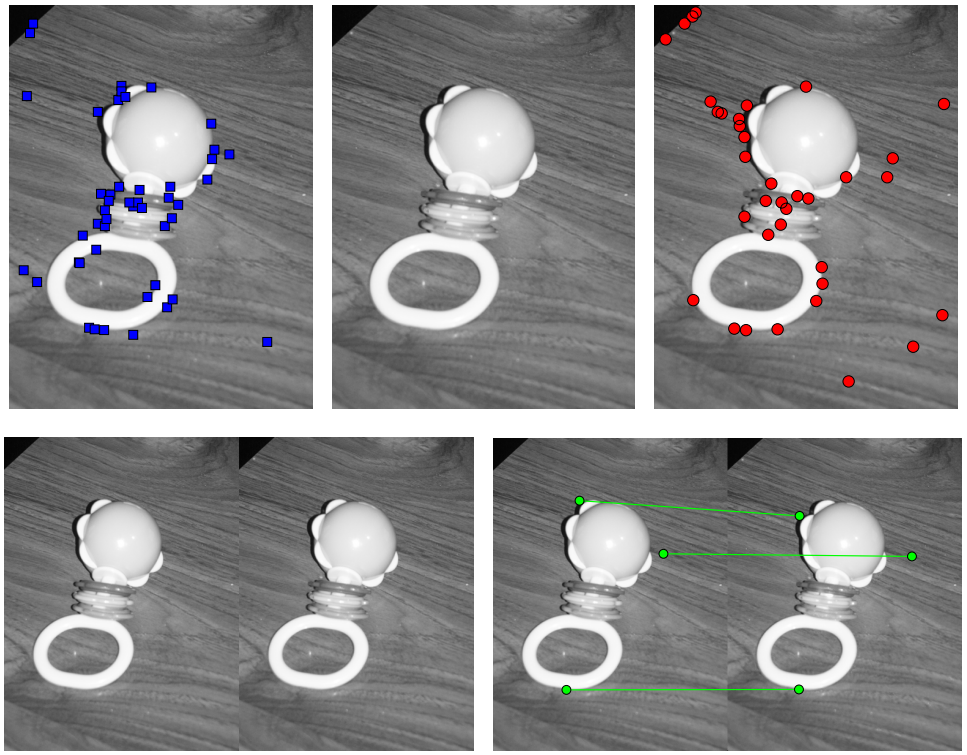


Figure 13.8: *Example of removal-injection procedure (ii).*

- **Concerning keypoints**. The CLBA-forged image does not contain any keypoint, which is quite suspicious per se being the image content very textured, as clearly evidenced by the authentic distribution of keypoints. On the contrary, the injected image looks more natural and the absence of keypoints over the clouds, which is an almost flat area, is not so strange. The original number of keypoints was 73, the injected amount is 35 ($KIR = 48\%$).

- **Concerning matches**. Obviously, the CLBA-forged image does not produce any match. Although the injected image has 12 matches (Fig. 13.7 bottom-right), only one of them is correct[4], i.e. that on the left spire.

[4]Recall that a match is deemed correct if the fake keypoint falls inside the 8×8 neigh-

This is a very interesting outcome, confirming that injection can support removal not only by concealing keypoint removal traces but also by further misleading any analysis that is based on matching.

- **Concerning quality**. The injection procedure does not deteriorate excessively the manipulated image. With respect to the authentic image, the following quality metrics are obtained: average patch PSNR of 32.8 dB; average patch SSIM of 0.9671; full frame PSNR of 38.9 dB; full-frame SSIM of 0.9941. In terms of full-frame PSNR, the injection causes a small loss of 2.5 dB with respect to the `CLBA`-forged image.

Similar conclusions can be drawn for the images of Fig. 13.8. The authentic image (top left) contains 53 keypoints; following a perfect removal (top center and bottom left), 41 keypoints were injected (top right, *KIR* 77%). The injection procedure generated only one match between the authentic and the removal-injection forgery (bottom right).

13.5.4 Time complexity of keypoint injection

In general, injection attacks are not significantly demanding in terms of computational resource, as the average execution times over the UCID data set in Fig. 13.9 show. Since full-frame attacks do not permit to decide the spatial location of the injected keypoints and thus do not require any particular iteration or check, their time complexity is very low and never exceeds 15 seconds. On the contrary, `FMD` and consequently `CLBI` relying on it, need about 60 and 90 seconds respectively to produce the injected image. This behaviour can be explained with the rather high amount of iterations (several thousands) required by the `FMD` to attempt the injection in all the candidate locations and to verify its outcome by means of a SIFT check. All tests have been performed with the same configuration used for keypoint removal, i.e. on MathWorks Matlab on a desktop configuration with 2 GHz dual-core processor, 4 GB RAM, 32 bit OS.

bourhood of its authentic homologue.

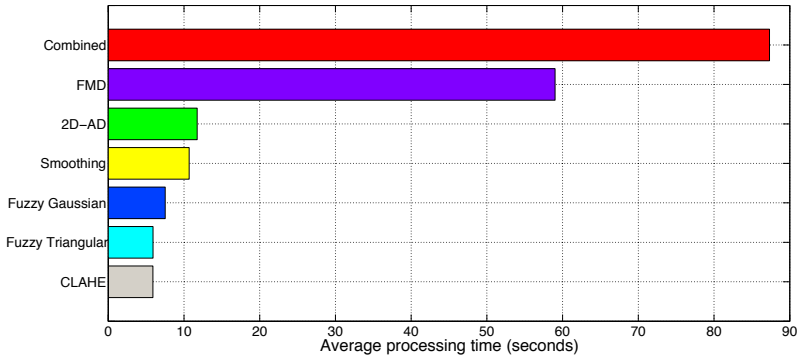


Figure 13.9: *Average processing time depending on the injection algorithm.*

13.6 Impact of CLBI on keypoint removal detection

We used 300 images to assess the effect of injection on keypoint removal the detectors: 100 randomly drawn from the UCID data set; the corresponding 100 CLBA-forgeries (effective removal rate $\geqslant 90\%$); the same 100 images after CLBI injection. We ran each of the detectors on all the images, we collected the detection scores and we organised them into scattergrams like in Secs. 12.1–12.3. In Fig. 13.10 we show the scattergrams relative to the KCR (top), CHI (middle) and SVM (bottom) detectors, with the blue squares representing authentic images, red circles images following keypoint removal and green triangles images following keypoint removal-injection.

The capability of separating the classes of authentic and CLBA-forged images is in line with the results obtained with different data sets but the scores of injected images tend to scatter and mix with those of the other two classes. This phenomenon is noticeable especially for SVM and CHI detectors, while KCR proves once again to be the most robust tool, as green triangles still appear separable from blue squares despite their proximity (see Fig. 13.10). This observation is corroborated by the data of Tab. 13.2 showing the confusion matrices for each of the detectors when the thresholds suggested in Sec. 12.4.5 for small images are used ($T_1 = -1.9$, $T_2 = 48$ and $T_3 = 0.41$).

Authentic and CLBA-forged images are correctly labelled with average detection accuracies of 92.5%, 93.5% and 100%, whereas the accuracy in la-

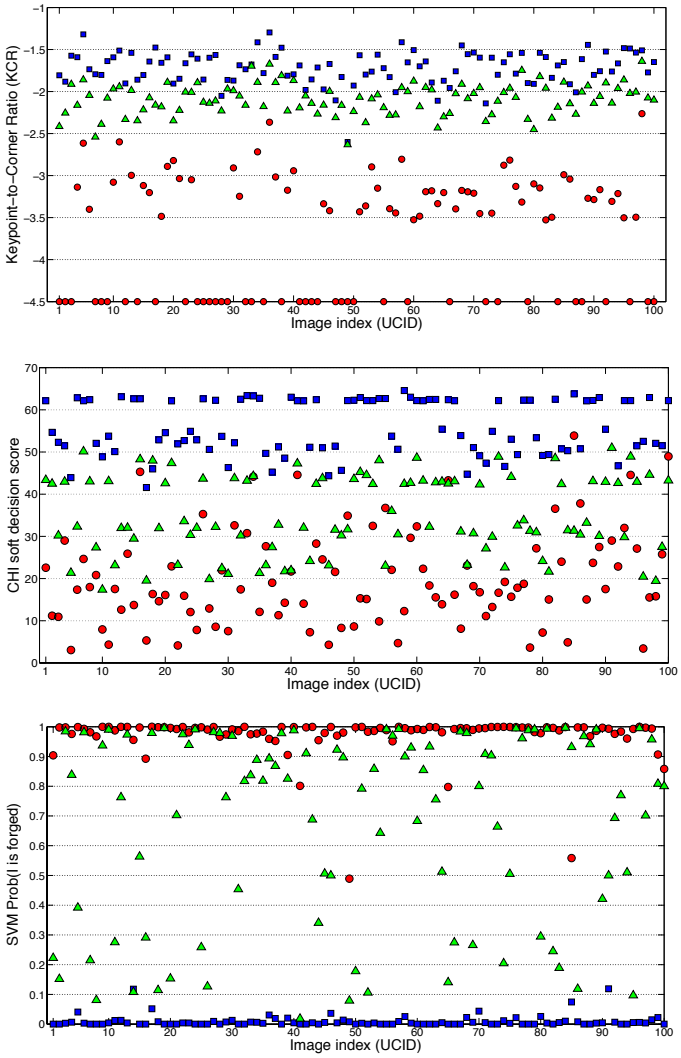


Figure 13.10: `KCR` *(top),* `CHI` *(middle) and* `SVM` *(bottom) scattergrams; blue squares: originals; red circles:* `CLBA`*; green triangles:* `CLBI`*.*

belling injected images as forgeries is 84%, 2% and 26% respectively; the consequences of keypoint injection are severe on `CHI` and `SVM`, for which slightly better results can be obtained, at the expense of the capability of identifying authentic images, by tweaking T_2 and T_3.

	KCR		CHI		SVM	
	Authentic	Forged	Authentic	Forged	Authentic	Forged
Authentic	**0.85**	0.15	**0.98**	0.02	**1**	0
CLBA	0	**1**	0.11	**0.89**	0	**1**
CLBI	0.16	**0.84**	0.98	**0.02**	0.74	**0.26**

Table 13.2: *Confusion matrices for the removal detectors (thresholds set to* -1.9*,* 48 *and* 0.41*).*

13.6.1 Three-class SVM detector

Based on the above observations, we improved the SVM detector by reformulating the underlying classification as a 3-class problem, in such a way to discriminate between authentic, CLBA-forged and injected images. In doing so, not only we were able to recognise a forged image but also to identify the manipulation it has undergone.

We have used $1,200$ UCID images of each class to build the probabilistic SVM model (200 for 5-fold cross-validation, $1,000$ for training with $C = 32$ and $\gamma = 0.125$) and the remaining 138 images per class to test it. In Tab. 13.3 we report the confusion matrix we obtained by assigning each test image to the class corresponding to the maximum of the output probability.

	Authentic	CLBA	CLBI
Authentic	**0.83**	0.02	0.15
CLBA	0.01	**0.94**	0.04
CLBI	0.15	0.05	**0.8**

Table 13.3: *Detection accuracy for the* 3*-class SVM detector*

The discriminative power of the 3-class SVM in presence of keypoint injection is now comparable to that of the KCR, with authentic and injected images misclassified in the 15% of the cases. We believe that such a behaviour is at least in part due to the nature of the UCID data set, whose relatively low amount of keypoints in images makes distinctions harder, and that it should subside as the image size grows.

13.7 Application to copy-move detection

As a final test, we investigate the interplay between all the tools described so far in the context of the detection of copy-move forgeries; this is, in fact, a typical image forensic problem that can be effectively tackled with by relying on SIFT features, whose robustness and distinctiveness allow to reliably match cloned areas. In Chapter 11 we saw that the goal of copy-move counter-forensics is to create a forgery that is undetectable by SIFT-based techniques; a clever counterfeiter can attain such goal by first applying keypoint removal to disable the targeted algorithm and then keypoint injection to hide the traces of removal exploited by removal detectors.[5] It goes without saying that all the manipulations are carried out locally on one (or more) of the cloned areas and leave the rest of the image unaltered.

13.7.1 Evaluation procedure and employed detectors

We considered again only two copy-moved areas, the source A_1 and its clone A_2, both containing a fair amount of keypoints because otherwise concealing the forgery would be trivial. We collected 10 images ranging from 1600×1200 to 3000×2000 pixels (some belonging to the data set used in [Christlein et al., 2012]) and we created 10 realistic forgeries by duplicating one region of size in the order of 300×400 pixels and variable shape (see for example Figs. 13.12 and 13.11 (a)–(b)).

We chose the copy-move forgery detector (CMFD) in [Amerini et al., 2013a], which improves the one by the same authors in [Amerini et al., 2011]. In a nutshell, following SIFT feature detection and hierarchical clustering, the algorithm in [Amerini et al., 2011] considers an image as forged if at least 3 matches are found within pairs of image regions. The major drawback with this approach is that requiring only 3 matches may lead to a large number of false positives, especially in images with many keypoints and repeated texture patterns such as walls. To overcome this problem, a new clustering technique was devised in [Amerini et al., 2013a] allowing to estimate the affine transfor-

[5] We do not include block-based detection in this analysis; however, due to its lack of robustness to geometric manipulations, it can be disabled as in Sec. 11.4.6 by cascading `CLBA` and simple geometric attacks like those in [Nguyen and Katzenbeisser, 2011].

mation between two sets of matched points. By extending such transformation to the image regions underlying matching sets, it becomes possible to localise the tampered areas. According to the improved detector, an image is tampered if at least one affine transformation is found among pairs of clusters. We detected keypoint removal by means of the `KCR` detector which, based on the experiments, proved to be the most robust to variations of image size, keypoint removal rate and injection.

Since several hundreds of keypoints at time may match across the cloned areas, the strategy adopted in Chapter 11 consisting in deleting half of the matches in each region, did not prove effective; consequently, we chose the second less perceptible solution: we let the counterfeiter remove all the keypoints in A_2 and those matches that are robust enough to survive the attack on A_2 have their members in A_1 removed instead. Then, A_2 is repopulated by means of `CLBI`.

13.7.2 Experimental analysis

Following each stage of the manipulation, the performances of the CMFD and the `KCR` detector are measured at image level to evaluate their capability of detecting tampered images and at pixel level to assess their accuracy in localising forged regions. For the first level, we simply consider the fraction of tampered images that are correctly identified. The metrics we chose for the second level are precision p and recall r:

$$p = \frac{Tp}{Tp + Fp} \qquad r = \frac{Tp}{Tp + Fn}. \tag{13.7}$$

When Eq. (13.7) are applied on a pixel basis, Tp is the number of forged pixels that are correctly identified; Fp is the number of authentic pixels erroneously labelled as forged; and Fn is the number of forged pixels erroneously labelled as authentic. Hence, precision is the fraction of pixels identified as tampered that are truly tampered and recall (or true positive rate) is the fraction of tampered pixels that are correctly classified as such. Precision and recall are conveniently combined by considering their harmonic mean, called F_1-score:

$$F_1 = 2\,\frac{p \cdot r}{p + r}. \tag{13.8}$$

(A) Authenticity verification

In our implementation, an image is forged according to the CMFD if there is at least one affine transformation linking A_1 to A_2, and according to the KCR detector if there is at least one tampered region whose area is $\geqslant 2\%$ of the image area. In Tab. 13.4, for each test image we report the number of keypoints of A_2, matches between A_1 and A_2 and the binary decision on image authenticity according to the CMFD in the plain copy-move, following removal and following removal-injection. Regardless of the underlying criterion (at least either 3 matches as in [Amerini et al., 2011] or 1 affine transformation as in [Amerini et al., 2013a]), the CMFD reveals 100% of the plain forgeries (Tab. 13.4, fourth column) but is always disabled by the keypoint manipulations (seventh and tenth columns). A similar analysis is carried out for the KCR detector in Tab. 13.5; obviously, the removal detector is incapable of discriminating until keypoints are actually altered (Tab. 13.5, third column); in the other cases, it recognises 100% of the counter-forensically treated copy-moves (fifth and seventh columns), since the regions it detects have on average an area which is 12.3% of the total image area after removal and 9.5% after removal-injection.

	Plain forgery			**After removal**			**After injection**		
Img	keypoints	matches	forged	keypoints	matches	forged	keypoints	matches	forged
I_1	157	130	1	1	1	0	85	0	0
I_2	1023	868	1	5	0	0	321	1	0
I_3	284	214	1	6	1	0	126	0	0
I_4	516	417	1	1	0	0	128	2	0
I_5	182	88	1	1	0	0	44	0	0
I_6	782	579	1	11	1	0	208	0	0
I_7	1211	929	1	2	0	0	184	2	0
I_8	1221	989	1	28	0	0	367	0	0
I_9	597	458	1	10	0	0	263	0	0
I_{10}	306	152	1	0	0	0	87	0	0

Table 13.4: *Keypoints in A_2, matches between A_1 and A_2 and binary decision on authenticity according to the CMFD following each stage of the forgery.*

	Plain forgery		**After removal**		**After injection**	
Img	area	forged	area	forged	area	forged
I_1	0%	0	7.8%	1	5.9%	1
I_2	0%	0	28.2%	1	25.1%	1
I_3	0%	0	9.1%	1	5.8%	1
I_4	0%	0	7.9%	1	5.7%	1
I_5	0%	0	9.2%	1	7.6%	1
I_6	0%	0	15.7%	1	14.3%	1
I_7	0%	0	7.7%	1	5.3%	1
I_8	0%	0	15.5%	1	11.4%	1
I_9	0%	0	11.5%	1	6.7%	1
I_{10}	0%	0	10.2%	1	6.9%	1
Avg.	**0**%		**12.3**%		**9.5**%	

Table 13.5: *Percentage of total image area detected as tampered by* KCR *and binary decision on authenticity following each stage of the forgery.*

(B) Tampering localisation

Tab. 13.6 shows the localisation accuracy of the CMFD and the KCR in terms of average p, r and F_1-score (computed over all the images' pixels) following each stage of the forgery and confirms the inadequateness of the CMFD in presence of keypoint manipulations. The repercussion of injection on KCR's performance is twofold: on the one hand, the probability of detecting forged pixels (i.e. r) is lowered by 18%, even though the reduction is insufficient to hide keypoint removal; on the other hand, the false positives caused by the sliding window approach on the borders of the tampered areas are also reduced, hence explaining the higher p.

Figs. 13.12 (I_2) and 13.11 (I_3), wherein the exit sign was hidden by replicating a portion of the wall and a cookie was duplicated, exemplify well the data of Tabs. 13.4–13.6 and the capability of injection in hiding the traces of keypoint removal to a visual investigation. The distributions of Figs. 13.12–13.11 (e) can still raise the suspicion of a keen observer but those of Figs. 13.12–13.11 (g) certainly require specialised tools. It is also worth noting that the CMFD allows to localise both the cloned areas whilst the KCR detector, due to the

	Plain copy-move			After removal			After injection		
Detector	p	r	F_1	p	r	F_1	p	r	F_1
CMFD	0.84	0.67	0.75	0	0	0	0	0	0
KCR	0	0	0	0.6	0.85	0.7	0.74	0.67	0.7

Table 13.6: *Average precision, recall and F_1-score for the CFMD and the* `KCR` *detector following each stage of the forgery.*

attacker's strategy, can identify only the one that has been tampered with. Two additional examples of localisation are shown in Fig. 13.13: the left column corresponds to the CMFD output on the plain copy-moves and the right column to the `KCR` output on the `CLBA`-attacked forgeries (on which the CMFD fails).

In conclusion, we can say that in the examined scenario the adversary struggles in evading copy-move detection if the forensic analyst resorts to the combination of the two above categories of detectors, e.g. by OR-ing their binary outputs on image authenticity or the tampering maps.

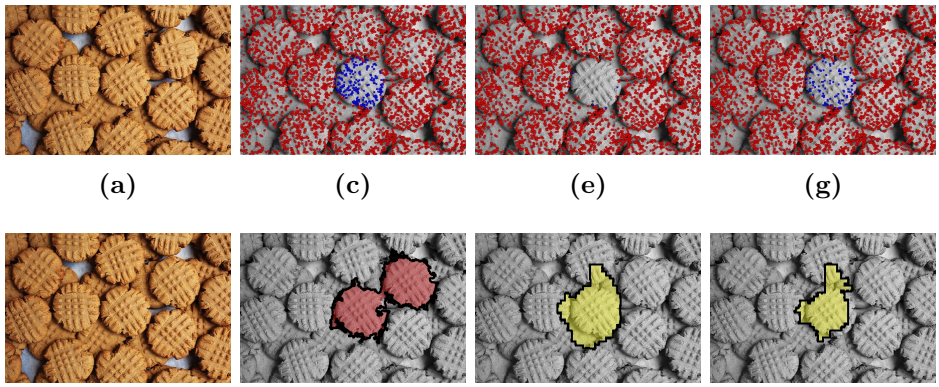


Figure 13.11: *Copy-moved image I_3. (a) Authentic; (b) copy-move; (c) keypoints of (a) (square markers indicate the keypoints of the cloned area); (d) output of the CMFD; (e) keypoints following removal; (f) output of KCR on (e); (g) keypoints following injection; and (h) output of KCR on (g).*

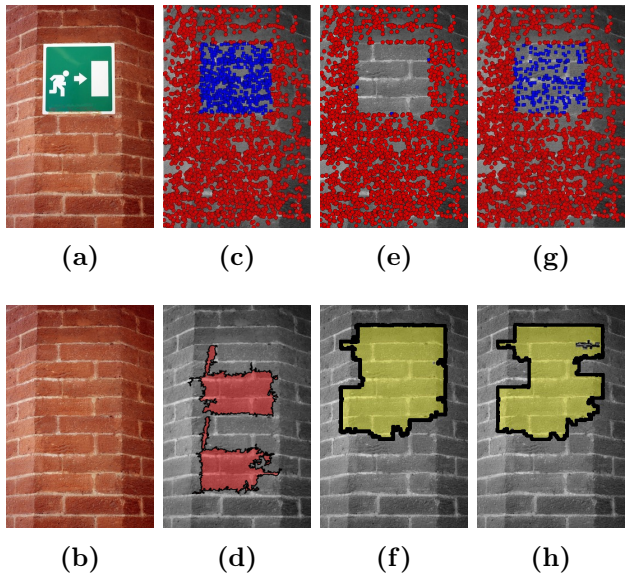


Figure 13.12: *Copy-moved image I_2. (a) Authentic; (b) copy-move forgery; (c) keypoints of (a) (square markers indicate the keypoints of the cloned area); (d) output of the CMFD; (e) keypoints following removal; (f) output of* KCR *on (e); (g) keypoints following injection; and (h) output of* KCR *on (g).*

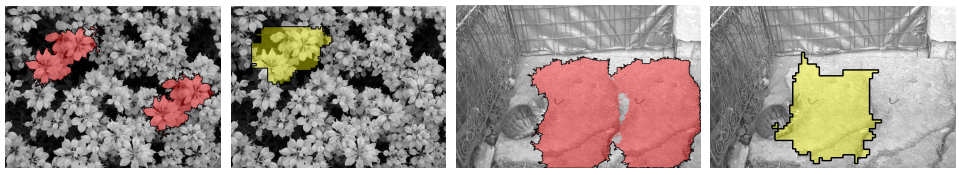


Figure 13.13: *Examples of localisation. (a)-(c): CMFD output highlighting copy-moved regions; (b)-(d):* KCR *output following removal.*

13.8 Concluding remarks

In this chapter we studied the injection of fake SIFT keypoints in an image whose authentic keypoints have been previously deleted. So far, no systematic study on this topic was carried out. Our interest stemmed from the consideration that an image with *too few* keypoints is per se a clue of counterfeit, which can be used to reveal the removal attack by means, for example, of the forensic detectors discussed in Chapter 12. As a consequence, the adversary must improve the imperceptibility of the removal attack by introducing new keypoints that are detectable by SIFT but do not have any correspondence with the authentic ones and do not interfere with the results attained with removal.

Five injection tools have been separately analysed and then combined into a single, more effective injection attack. Experimental results show that injection is feasible without causing a successive detection at SIFT matching level. Moreover, sometimes false matches can be introduced to further mislead matching-based analysis. Visual quality is preserved both with respect to the original image and, particularly, in comparison with the image quality achieved after keypoint removal. By resorting to the proposed injection attack, it was possible to reduce the discriminative power of the keypoint removal detectors (dramatically, for two out of three). However, when applied to counter SIFT-based copy-move forgery detection, removal and injection were not able to bypass the most robust among the keypoint removal detectors.

Chapter 14

Conclusion

"Now this is not the end.
It is not even the beginning of the end.
But it is, perhaps, the end of the beginning."

Winston Churchill

IN THIS thesis we focused on three different image forensic problems: the study of the history and the relationships between near-duplicate images; the fusion of multiple detection scores produced by a set of tools evaluating the authenticity of an image; and the study of weaknesses of SIFT-based image forensic algorithms.

14.1 Summary

Our interest for the first problem stemmed from a simple observation. When we query any web engine for an image, several almost identical versions of the queried image are usually retrieved; most of such images differ in colors, size, contrast and so on. Clearly, there must be some links (or *dependencies*) between them, which are rarely documented by the users who downloaded, modified and redistributed the images. Though the current state-of-the-art of Image Forensics permits to gather very interesting information about the *history* of an image, the majority of instruments developed so far focus on the analysis of single images. In several applications, however, the investigation the relationships between a group of images, may be of similar or even greater importance (e.g. copyright infringement, clustering, retrieval, news and opinion tracking). We tried to explore such relationships by casting the problem into a theoretical framework. Since we could not rely on the visual content

of identical images to understand whether a digital image has been produced by starting from another image representing the same scene, we proposed a content-independent footprint characterising the process that produced the images. Based on such a footprint, which is in principle similar to image noise, we considered two images as dependent if some form of similarity exists between their content-independent components.

We also implemented a system, called *Image Dependency Explorer*, putting the framework in practice. The system estimates the best transformation leading from an image A to an image B as the composition of a subset of the most common processing tools: color manipulations, contrast and brightness adjustments, geometric transformation and compression. The analysis is applied in a pair-wise fashion to the data set of duplicates and a graph describing their relationships is produced. The tests we conducted on synthetic data sets demonstrated the reliability of the relationships when the ground truth is known; tests on real Web data sets demonstrated the plausibility of the relationships when the ground truth is unknown. In this latter case we resorted to alternative ways to evaluate the performance of the system, such as visual investigation and the introduction of small subsets of duplicates with known ground truth working in a similar way to the litmus paper in chemistry.

Concerning the second topic addressed in the thesis, we proposed a theoretical framework for the cooperation of forensic tools by means of the fusion of their decision scores. Generally, each forensic technique deals with the detection of a typical footprint left by a single processing tool under specific settings. Forensic techniques, however, like any other realistic process or system, are never perfect and their measurements are usually affected by uncertainty, ambiguity or impreciseness. Another obstacle to overcome when judging the integrity of a given image is that most of the times a tampered image is not the result of the application of a single processing tool. Since rarely we know beforehand the kind of manipulation the image has undergone, the application of a single footprint detection technique may not be enough, thus requiring the parallel use of more than one technique. A problem with the use of several tools looking for different footprints is that each tool provides an output describing the degree of presence of the specific footprint it is looking for. Even

when using more than one tool, we are interested in obtaining a single global answer allowing to decide whether the image under analysis is authentic or not. Obtaining such a global answer is not a trivial task, as outputs may not only be inaccurate but also heterogeneous.

We tackled with both the above problems by means of a *decision fusion* framework based on Fuzzy Theory, because of its capability to address problems whose mathematical or statistical models are hard to define. Our framework resorts to the experience and the knowledge of the forensic analyst to mimic her behaviour. An analyst, would first tweak the tools at her disposal by gathering as much information as possible (e.g. which ones are the most trustworthy, on what kind of images they work, how they interact with each other), thus tackling with the uncertainty problem. Then, she would run all the tools on the image under analysis and exploit the previously gathered knowledge to make a final decision, thus tackling with the fusion problem. The proposed framework translating into practice such rationale brings several advantages: it outperforms canonical approaches; it is easily scalable and allows incremental addition of new forensic tools without requiring new training; it does not require mathematical models; it addresses the fusion problem in a sound yet intuitive way. We validated the system by fusing five different forensic detectors revealing cut & paste manipulation, with superior results with respect to those guaranteed by traditional techniques.

The final part of the thesis is dedicated to Counter-forensics, that is the exploitation of the weaknesses of forensic algorithms to understand the limits of the existing tools not only in presence of "innocent" processing but also in presence of a malicious adversary interested in fooling a certain forensic analysis. We focused on a class of algorithms not impaired so far, i.e. SIFT-based detectors. Our first contribution is a new method to remove SIFT keypoints, called Classification-based Attack (`CLBA`). Such algorithm is based on the concept of keypoint classification preceding the attack itself; identifying classes of keypoints with different properties, in fact, allows to choose the attack that fits the most to such properties. Multiple iterations of the classification and attack procedure allow the removal of practically all the first-octave keypoints and, to some extent, of a satisfactory amount of more robust, higher-octave

keypoints. By using this solution, we were able to outperform state-of-the-art class-unaware keypoint removal attacks in terms of effectiveness and impact on the counterfeit's quality. Furthermore, we used the CLBA to impair SIFT-based copy-move detection, which represents one of the most robust and reliable approaches to reveal this common way to tamper with an image. In this scenario, the counterfeiter can create a forensically undetectable copy-move by deleting those keypoints whose matches across cloned regions have been revealed by the targeted SIFT-based copy-move detector. The results we obtained show that with CLBA the adversary always succeeds in bypassing the targeted forensic algorithms, thus representing a serious threat to SIFT-based detection in general.

To tackle with this problem, we devised three forensic detectors for the identification of images whose SIFT keypoints have been artificially removed. The proposed algorithms scan image regions with sufficiently high variance in search of suspect inconsistencies in the number and in the distribution of SIFT keypoints. By relying on such algorithms, the forensic analyst can decide on the authenticity of the image as a whole or localise tampered regions within the image by means of a sliding window approach.

We then provided the adversary with new tools to bypass keypoint removal detection by re-introducing fake keypoints into the attacked image regions. We combined the most performing of such tools into a new method, called Classification-based Injection (CLBI), based on the same principle of CLBA, i.e. a classification-driven procedure exploiting the advantages of each single injection algorithm while limiting its weaknesses. By relying on such an attack, we assessed the robustness of keypoint removal detectors in the presence of an adversary and we demonstrated that in the copy-move detection scenario the forensic analyst is not defeated.

14.2 Open issues

The work of this thesis could benefit from further research. Regarding the first part, future works should begin with a theoretically sound formulation of the hypothesis testing problem lying at the heart of the Dependency Explorer Framework, in such a way to dismiss heuristic aspects like the empirical thresholding of pairwise correlations. From a more practical point of view, it would be interesting to push forward the integration between the proposed system and the existing forensic tools for source and forgery detection. Knowing the source of the images composing the set of near-duplicates would help ruling out implausible relationships (e.g. similar images generated by different camera brands), thus ensuring the construction of a more accurate dependency graph at a reduced computational cost. Knowing that an image has been tampered with would have a number of interesting consequences, such as: determining with more accuracy parent-child relationships; in the case of image splicing, lifting the hypothesis of a single parent for each image; tracking the history of the tampered regions by means of a dedicated dependency subgraph.

Concerning the second part of the thesis, it is of paramount importance to extend the theoretical fusion framework to the most complex case in which the suspicious tampered region is not known a priori. As a matter of fact, forensic detectors such as those we considered in the implementation of our fuzzy fusion framework can be used to localise tampered regions of an image, e.g. by means of a sliding window approach. In this case, however, a potential issue would arise if each tool produces different tampering maps or different localisations. Therefore, it could be necessary to somehow fuse the spatial maps to tackle with this new form of uncertainty and reach a consensus on the suspicious region. Once the region has been located, the fuzzy fusion can proceed as we described in this thesis, by relying on the detection scores of each tool in the suspicious region.

Finally, among the other possible extensions, we mention the following: to implement a wider set of forensic tools working on different manipulations in order to better exploit the capability of the system to deal with heterogeneous outputs; to compare the fuzzy method against other soft decision approaches, like Bayesian and Neural Networks, on extended real-world datasets.

As for the third part, much work can still be done. The removal of SIFT keypoints should be expanded to encompass multiple octaves at the same time. Even though the majority of the results of the proposed attack were obtained on the keypoints detected at the first SIFT octave, our preliminary experiments show that the trend is similar for the remaining octaves and we have already proposed a variation of the original algorithm taking into account multiple octaves. Nevertheless, its effectiveness on higher octaves should be investigated in-depth so that an acceptable trade-off between removal rate and visual quality of the forgery is attained.

It could also be useful to adapt the `CLBA` in such a way to make possible the removal of keypoints detected by other techniques like, for example, SURF (Speeded Up Robust Features) [Bay et al., 2008] and MSER (Maximally Stable External Regions) [Matas et al., 2004]. As a matter of fact, while removing SIFT keypoints, `CLBA` can already reduce the population of SURF keypoints and MSER regions as shown in Fig. 14.1. We obtained the data in Fig. 14.1 as follows. We ran a version of the `CLBA` consisting only of two attacks, i.e. Gaussian `Smoothing` and `Collage`[1] attacks on 400 UCID images; we did not consider the `RMD` attack because it is specifically tailored to the SIFT algorithms and thus can not be applied as it is to other keypoint detectors. Then, as usual, we calculated the envelope of the *KRR* histograms.

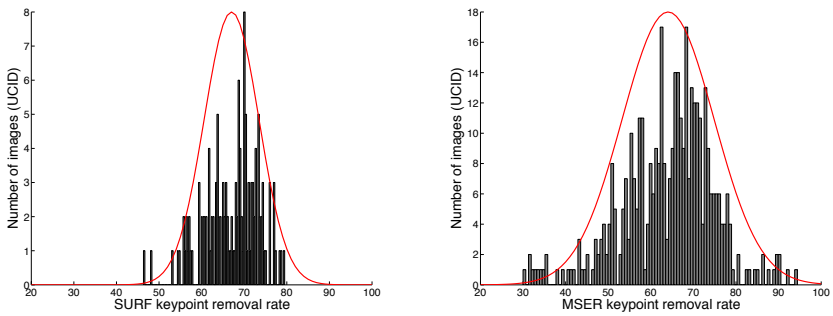


Figure 14.1: *KRR attained with* `CLBA` *on SURF and MSER keypoints.*

[1] Obviously, we have collected two databases consisting of image patches without SURF keypoints and MSER regions.

It turns out that on average this simplified version of `CLBA` is capable of deleting 66% of the SURF keypoints and 62% of the MSER regions. These exploratory results are promising and suggest that `CLBA` may be also able to counter applications based on these two keypoint detectors.

In a long-term perspective, it would also be very interesting to consider another type of attack, i.e. altering SIFT descriptors; such an attack could be used to impair matching-based applications by hiding matches without deleting the corresponding keypoints or by introducing false positive matches.

Finally, concerning keypoint injection, we are working on the re-introduction of the color information into the forged image. To do so, we work in the YCbCr color space: first we perform the removal-injection procedure on the luminance channel of the authentic image; then, we reconstruct the color forgery by means of the attacked luminance and of the authentic color channels Cb and Cr. Since color conversions may accidentally introduce correctly matching keypoints, the proposed attack must be tuned to cope with such side effect.

Bibliography

Abdel-Hakim, A. E. and Farag, A. A. (2006). CSIFT: a SIFT descriptor with color invariant characteristics. In *Proc. of the IEEE Int. Conference on Computer Vision and Pattern Recognition (CVPR06)*, volume 2, pages 1978–1983. IEEE.

Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., Del Tongo, L., and Serra, G. (2013a). Copy-move forgery detection and localization by means of robust clustering with j-linkage. *Signal Processing: Image Communication*.

Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., and Serra, G. (2011). A SIFT-based forensic method for copy move attack detection and transformation recovery. *IEEE Trans. on Information Forensics and Security*, 6(3):1099 –1110.

Amerini, I., Barni, M., Caldelli, R., and Costanzo, A. (2013b). Counter-forensics of SIFT-based copy-move detection by means of keypoint classification. *EURASIP Journal on Image and Video Processing*, 2013(1):18.

Amerini, I., Barni, M., Caldelli, R., and Costanzo, A. (2013c). SIFT keypoint removal and injection for countering matching-based Image Forensics. In *The 1st ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM.

Amerini, I., Battisti, F., Caldelli, R., Carli, M., and Costanzo, A. (2014). Exploiting perceptual quality issues in countering SIFT-based forensic methods. In *[Submitted to] the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP14)*. IEEE.

Arganda-Carreras, I., Sorzano, C. O. S., Marabini, R., Carazo, J. M., de Solorzano, C. O., and Kybic, J. (2006). Consistent and elastic registration of histological sections using vector-spline regularization. In *Computer Vision Approaches to*

Medical Image Analysis, volume 4241 of *Lecture Notes in Computer Science*, pages 85–95. Springer Berlin / Heidelberg.

Attneave, F. (1954). Some informational aspects of visual perception. *Psychological review*, 61(3):183–193.

Avidan, S. and Shamir, A. (2007). Seam carving for content-aware image resizing. In *ACM Trans. on Graphics (TOG07)*, volume 26, page 10. ACM.

Barni, M. (2012). A game theoretic approach to source identification with known statistics. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP12)*, pages 1745–1748. IEEE.

Barni, M. and Bartolini, F. (2004). *Watermarking systems engineering: enabling digital assets security and other applications*. CRC Press.

Barni, M. and Costanzo, A. (2012a). Dealing with uncertainty in image forensics: A fuzzy approach. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP12)*, pages 1753–1756. IEEE.

Barni, M. and Costanzo, A. (2012b). A fuzzy approach to deal with uncertainty in image forensics. *Signal Processing: Image Communication*.

Barni, M., Costanzo, A., and Sabatini, L. (2010). Identification of cut & paste tampering by means of double-JPEG detection and image segmentation. In *Proc. of the IEEE Int. Symposium on Circuits and Systems (ISCAS10)*, pages 1687–1690. IEEE.

Barni, M., Fontani, M., and Tondi, B. (2012). A universal technique to hide traces of histogram-based image manipulations. In *Proc. of 17th ACM Workshop on Multimedia and Security*, pages 97–104. ACM.

Barni, M. and Tondi, B. (2012). Optimum forensic and counter-forensic strategies for source identification with training data. In *Proc. of the IEEE Int. Workshop on Information Forensics and Security (WIFS12)*, pages 199–204. IEEE.

Barni, M. and Tondi, B. (2013). The source identification game: An information-theoretic perspective. *IEEE Trans. on Information Forensics and Security*, 8(3):450–463.

Battiato, S., Bruna, A. R., Messina, G., and Puglisi, G. (2010). *Image processing for embedded devices*. Bentham Science Publishers.

Battiato, S., Farinella, G. M., Puglisi, G., and Ravi, D. (2013). Aligning codebooks for near duplicate image detection. *Multimedia Tools and Applications*, pages 1–24.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

Bayram, S., Sencar, H., Memon, N., and Avcibas, I. (2005). Source camera identification based on cfa interpolation. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP05)*, volume 3, pages III–69. IEEE.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18:509–517.

Bentley, J. L. (1980). Multidimensional divide-and-conquer. *Communications of the ACM*, 23(4):214–229.

Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. (2000). Image inpainting. In *Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co.

Bianchi, T., De Rosa, A., and Piva, A. (2011). Improved dct coefficient analysis for forgery localization in jpeg images. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP11)*, pages 2444–2447. IEEE.

Bianchi, T. and Piva, A. (2011). Detection of non-aligned double jpeg compression with estimation of primary compression parameters. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP11)*, pages 1929–1932. IEEE.

Böhme, R. and Kirchner, M. (2012). Counter-forensics: attacking image forensics. In Sencar, H. T. and Memon, N., editors, *Digital Image Forensics (Chapter 10)*. Springer, New York.

Bosch, A., Zisserman, A., and Munoz, X. (2006). Scene classification via pLSA. In *Proc. of the European Conference on Computer Vision (ECCV06)*, pages 517–530. Springer.

Bosch, A., Zisserman, A., and Muoz, X. (2007). Image classification using random forests and ferns. In *Proc. of the IEEE 11th Int. Conference on Computer Vision (ICCV07)*, pages 1–8. IEEE.

Brown, M. and Lowe, D. G. (2002). Invariant features from interest point groups. In *British Machine Vision Conference*, volume 21, pages 656–665.

Burghouts, G. J. and Geusebroek, J.-M. (2009). Performance evaluation of local colour invariants. *Computer Vision and Image Understanding*, 113(1):48–62.

Burt, P. J. (1981). Fast filter transform for image processing. *Computer graphics and image processing*, 16(1):20–51.

Caldelli, R., Amerini, I., Ballan, L., Serra, G., Barni, M., and Costanzo, A. (2012). On the effectiveness of local warping against SIFT-based copy-move detection. In *Proc. of the Int. Symposium on Communications, Control and Signal Processing (ISCCSP12)*, Roma, Italy.

Caldelli, R., Amerini, I., Picchioni, F., and Innocenti, M. (2010). Fast image clustering of unknown source images. In *IEEE Int. Workshop on Information Forensics and Security (WIFS10)*, pages 1–5. IEEE.

Cao, G., Zhao, Y., Ni, R., and Tian, H. (2010a). Anti-forensics of contrast enhancement in digital images. In *Proc. of the 12th ACM Workshop on Multimedia and Security*, pages 25–34. ACM.

Cao, G., Zhao, Y., Ni, R., Yu, L., and Tian, H. (2010b). Forensic detection of median filtering in digital images. In *Proc. of the IEEE Int. Conference on Multimedia and Expo (ICME10)*, pages 89–94. IEEE.

Celiktutan, O., Avcibas, I., Sankur, B., Ayerden, N., and Capar, C. (2006). Source cell-phone identification. In *Proc. of the IEEE 14th Signal Processing and Communications Applications*, pages 1–3. IEEE.

Celiktutan, O., Sankur, B., and Avcibas, I. (2008). Blind identification of source cell-phone model. *IEEE Trans. on Information Forensics and Security*, 3(3):553–566.

Chandramouli, R., Kharrazi, M., and Memon, N. (2004). Image steganography and steganalysis: Concepts and practice. In *Digital Watermarking*, pages 35–49. Springer.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Trans. on Intelligent Systems and Technology*, 2:27:1–27:27.

Chang, J.-H., Fan, K.-C., and Chang, Y.-L. (2002). Multi-modal gray-level histogram modeling and decomposition. *Image and Vision Computing*, 20(3):203–216.

Chatzis, V., Bors, A. G., and Pitas, I. (1999). Multimodal decision-level fusion for person authentication. *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans*, 29(6):674–680.

Chen, C. and Shi, Y. Q. (2008). JPEG image steganalysis utilizing both intrablock and interblock correlations. In *Proc. of the IEEE Int. Symposium on Circuits and Systems (ISCAS08)*, pages 3029–3032. IEEE.

Chen, D., Odobez, J.-M., and Bourlard, H. (2004). Text detection and recognition in images and video frames. *Pattern Recognition*, 37(3):595–608.

Chen, M., Fridrich, J., Goljan, M., and Lukás, J. (2008). Determining image origin and integrity using sensor noise. *IEEE Trans. on Information Forensics and Security*, 3(1):74–90.

Chen, W., Shi, Y. Q., and Xuan, G. (2007). Identifying computer graphics using hsv color model and statistical moments of characteristic functions. In *Proc. of the IEEE Int. Conference on Multimedia and Expo (ICME07)*, pages 1123–1126. IEEE.

Chetty, G. and Singh, M. (2010). Nonintrusive image tamper detection based on fuzzy fusion. *Int. Journal of Computer Science and Network Security*, 10:86–90.

Christlein, V., Riess, C., Jordan, J., Riess, C., and Angelopoulou, E. (2012). An evaluation of popular copy-move forgery detection approaches. *IEEE Trans. on Information Forensics and Security*, 7(6):1841 –1854.

Chum, O., Philbin, J., Isard, M., and Zisserman, A. (2007). Scalable near identical image and shot detection. In *Proc. of the 6th ACM Int. Conference on Image and Video Retrieval*, pages 549–556. ACM.

Chunhui, F., Zhengquan, X., and Xinghui, Z. (2012). An anti-forensic algorithm of JPEG double compression based forgery detection. In *Int. Symposium on Information Science and Engineering (ISISE12)*, pages 159–164. IEEE.

Conotter, V., Boato, G., and Farid, H. (2010). Detecting photo manipulation on signs and billboards. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP2010)*, pages 1741–1744. IEEE.

Cover, T. M. and Thomas, J. A. (Second Ed., 2006). *Elements of Information Theory*. John Wiley & Sons, New Jersey.

Cozzolino, D., Gargiulo, F., Sansone, C., and Verdoliva, L. (2013). Multiple classifier systems for image forgery detection. In *Proc. of the IEEE 17th Int. Conference on Image Analysis and Processing (ICIAP13)*.

Criminisi, A., Perez, P., and Toyama, K. (2003). Object removal by exemplar-based inpainting. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR03)*, volume 2, pages II–721. IEEE.

Crowley, J. L. and Parker, A. C. (1984). A representation for shape based on peaks and ridges in the difference of low-pass transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, (2):156–170.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Proc. of the European Conference on Computer Vision (ECCV04)*, volume 1, pages 1–22.

D'Angelo, A. and Barni, M. (2008). A structural method for quality evaluation of desynchronization attacks in image watermarking. In *Proc. of the IEEE 10th Workshop on Multimedia Signal Processing*, pages 754–759.

Das, S., Darsan, G., L, S., and Devan, D. (2012). Blind detection method for video inpainting forgery. *Int. Journal of Computer Applications*, 60(11):33–37.

De Boor, C. (1978). *A practical guide to splines*, volume 27. Springer-Verlag New York.

De Rosa, A., Uccheddu, F., Costanzo, A., Piva, A., and Barni, M. (2010). Exploring image dependencies: a new challenge in image forensics. In *IS&T/SPIE Electronic Imaging*, pages 75410X–75410X. Int. Society for Optics and Photonics.

Dehnie, S., Sencar, T., and Memon, N. (2006). Digital image forensics for identifying computer generated and digital camera images. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP06)*, pages 2313–2316. IEEE.

Delgado, M., Gómez-Skarmeta, A., and Martin, F. (1997). A fuzzy clustering-based rapid prototyping for fuzzy rule-based modeling. *IEEE Trans. on Fuzzy Systems*, 5(2):223–233.

Dias, Z., Rocha, A., and Goldenstein, S. (2012). Image phylogeny by minimal spanning trees. *IEEE Trans. on Information Forensics and Security*, 7(2):774–788.

Dirik, A. E., Sencar, H. T., and Memon, N. (2008). Digital single lens reflex camera identification from traces of sensor dust. *IEEE Trans. on Information Forensics and Security*, 3(3):539–552.

Do, T.-T., Kijak, E., Amsaleg, L., and Furon, T. (2012). Enlarging hacker's toolbox: deluding image recognition by attacking keypoint orientations. In *Proc. of the*

IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP12), pages 1817–1820. IEEE.

Do, T.-T., Kijak, E., Furon, T., and Amsaleg, L. (2010a). Deluding image recognition in SIFT-based CBIR systems. In *Proc. of the 2nd ACM Workshop on Multimedia in Forensics, Security and Intelligence*, pages 7–12, New York, NY, USA. ACM.

Do, T.-T., Kijak, E., Furon, T., and Amsaleg, L. (2010b). Understanding the security and robustness of SIFT. In *Proc. of the 18th ACM Int. Conference on Multimedia*, pages 1195–1198, New York, NY, USA. ACM.

Domingos, P. and Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130.

Fan, W., Wang, K., Cayre, F., and Xiong, Z. (2013a). JPEG anti-forensics using non-parametric DCT quantization noise estimation and natural image statistics. In *The 1st ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM.

Fan, W., Wang, K., Cayre, F., and Xiong, Z. (2013b). A variational approach to JPEG anti-forensics. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP13)*. IEEE.

Fan, Z. and de Queiroz, R. (2000). Maximum likelihood estimation of jpeg quantization table in the identification of bitmap compression history. In *Proc. of the Int. Conference on Image Processing (ICIP00)*, volume 1, pages 948–951. IEEE.

Fan, Z. and de Queiroz, R. L. (2003). Identification of bitmap compression history: JPEG detection and quantizer estimation. *IEEE Trans. on Image Processing*, 12(2):230–235.

Fang, Y., Dirik, A. E., Sun, X., and Memon, N. (2009). Source class identification for DSLR and compact cameras. In *Proc. of the IEEE Int. Workshop on Multimedia Signal Processing*, pages 1–5. IEEE.

Farid, H. (2008). Digital image ballistics from jpeg quantization: A followup study. *Department of Computer Science, Dartmouth College*.

Farid, H. (2009a). Exposing digital forgeries from JPEG ghosts. *IEEE Trans. on Information Forensics and Security*, 4:154–160.

Farid, H. (2009b). Image forgery detection. *IEEE Signal Processing Magazine*, 26(2):16–25.

Fillion, C. and Sharma, G. (2010). Detecting content adaptive scaling of images for forensic applications. In *IS&T/SPIE Electronic Imaging*, pages 75410Z–75410Z. Int. Society for Optics and Photonics.

Fontani, M. and Barni, M. (2012). Hiding traces of median filtering in digital images. In *Proc. of the 20th European Signal Processing Conference (EUSIPCO12)*, pages 1239–1243. IEEE.

Fontani, M., Bianchi, T., De Rosa, A., Piva, A., and Barni, M. (2011). A dempster-shafer framework for decision fusion in image forensics. In *Proc. of the IEEE Int. Workshop on Information Forensics and Security (WIFS11)*, pages 1–6. IEEE.

Fontani, M., Bianchi, T., De Rosa, A., Piva, A., and Barni, M. (2013). A framework for decision fusion in image forensics based on dempster-shafer theory of evidence. *IEEE Trans. on Information Forensics and Security*.

Fontani, M., Costanzo, A., Barni, M., Bianchi, T., De Rosa, A., and Piva, A. (2012). Two decision fusion frameworks for image forensics. In *Annual meeting of the Telecommunications and Information Technology Group (GTTI)*.

Foo, J. J. and Sinha, R. (2007). Pruning SIFT for scalable near-duplicate image matching. In *Proc. of the 18th Conference on Australasian database*, pages 63–71. Australian Computer Society, Inc.

Foo, J. J., Zobel, J., and Sinha, R. (2007a). Clustering near-duplicate images in large collections. In *Proc. of the Int. Workshop on Multimedia Information Retrieval*, pages 21–30. ACM.

Foo, J. J., Zobel, J., Sinha, R., and Tahaghoghi, S. M. (2007b). Detection of near-duplicate images for web search. In *Proc. of the 6th ACM Int. Conference on Image and Video Retrieval*, pages 557–564. ACM.

Fridrich, A., Soukal, B., and Lukáš, A. (2003a). Detection of copy-move forgery in digital images. In *Proc. of Digital Forensic Research Workshop*. Citeseer.

Fridrich, J., Goljan, M., and Hogea, D. (2003b). Steganalysis of jpeg images: Breaking the f5 algorithm. In *Information Hiding*, pages 310–323. Springer.

Fridrich, J., Goljan, M., and Memon, N. D. (2000). Further attacks on Yeung-Mintzer fragile watermarking scheme. In *Security, Steganography, and Watermarking of Multimedia Contents*, volume 3971, pages 428–437.

Geusebroek, J.-M., Van Den Boomgaard, R., Smeulders, A. W., and Dev, A. (2000). Color and scale: the spatial structure of color images. In *Proc. of the European Conference on Computer Vision (ECCV00)*, pages 331–341. Springer.

Gloe, T., Borowka, K., and Winkler, A. (2009). Feature-based camera model identification works in practice. In *Information Hiding*, pages 262–276. Springer.

Gloe, T., Franz, E., and Winkler, A. (2007a). Forensics for flatbed scanners. *Proc. of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX*, 6505:1I–1J.

Gloe, T., Kirchner, M., Winkler, A., and Böhme, R. (2007b). Can we trust digital image forensics? In *Proc. of the 15th ACM Int. Conference on Multimedia*, pages 78–86. ACM.

Goljan, M., Fridrich, J., and Chen, M. (2010). Sensor noise camera identification: Countering counter-forensics. In *IS&T/SPIE Electronic Imaging*, pages 75410S–75410S. Int. Society for Optics and Photonics.

Gou, H., Swaminathan, A., and Wu, M. (2007). Robust scanner identification based on noise features. *Proc. of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX*, 6505:0S–0T.

Hall, D., De Verdière, V. C., and Crowley, J. L. (2000). Object recognition using coloured receptive fields. In *Proc. of the European Conference on Computer Vision (ECCV00)*, pages 164–177. Springer.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision Conference*, volume 15, page 50. Manchester, UK.

Hess, R. (2010). An open-source SIFTLibrary. In *Proc. of the ACM Int. Conference on Multimedia*, pages 1493–1496. ACM.

Hsu, C.-C., Hung, T.-Y., Lin, C.-W., and Hsu, C.-T. (2008). Video forgery detection using correlation of noise residue. In *Proc. of the IEEE 10th Workshop on Multimedia Signal Processing*, pages 170–174. IEEE.

Hsu, C.-Y., Lu, C.-S., and Pei, S.-C. (2009). Secure and robust SIFT. In *Proc. of the 17th ACM Int. Conference on Multimedia*, MM '09, pages 637–640, New York, NY, USA. ACM.

Huang, H., Guo, W., and Zhang, Y. (2008). Detection of copy-move forgery in digital images using SIFT algorithm. In *Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA08)*, volume 2, pages 272–276.

Jaimes, A. (2003). *Conceptual structures and computational methods for indexing and organization of visual information*. PhD thesis, Columbia University.

Jain, A., Nandakumar, K., and Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern recognition*, 38(12):2270–2285.

Jegou, H., Douze, M., and Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search. In *Proc. of the European Conference on Computer Vision (ECCV08)*, pages 304–317. Springer.

Johnson, M. K. and Farid, H. (2005). Exposing digital forgeries by detecting inconsistencies in lighting. In *Proc. of the 7th ACM Workshop on Multimedia and Security*, pages 1–10. ACM.

Johnson, M. K. and Farid, H. (2006). Exposing digital forgeries through chromatic aberration. In *Proc. of the 8th ACM Workshop on Multimedia and Security*, pages 48–55. ACM.

Johnson, M. K. and Farid, H. (2007). Exposing digital forgeries in complex lighting environments. *IEEE Trans. onInformation Forensics and Security*, 2(3):450–461.

Johnson, M. K. and Farid, H. (2008). Detecting photographic composites of people. In *Digital Watermarking*, pages 19–33. Springer.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer, second edition.

Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR04)*, volume 2, pages II–506. IEEE.

Ke, Y., Sukthankar, R., and Huston, L. (2004). An efficient parts-based near-duplicate and sub-image retrieval system. In *Proc. of the 12th annual ACM Int. Conference on Multimedia*, pages 869–876. ACM.

Keimel, C., Habigt, J., Horch, C., and Diepold, K. (2012). Qualitycrowd – a framework for crowd-based quality evaluation. In *Proc. of the IEEE Picture Coding Symposium (PCS12)*, pages 245–248. IEEE.

Kennedy, L. and Chang, S.-F. (2008). Internet image archaeology: automatically tracing the manipulation history of photographs on the web. In *Proc. of the 16th ACM Int. Conference on Multimedia*, pages 349–358. ACM.

Khanna, N. and Delp, E. J. (2010). Intrinsic signatures for scanned documents forensics: effect of font shape and size. In *Proc. of IEEE Int. Symposium on Circuits and Systems (ISCAS10)*, pages 3060–3063. IEEE.

Khanna, N., Mikkilineni, A. K., Chiu, G. T., Allebach, J. P., and Delp, E. J. (2007a). Forensic classification of imaging sensor types. In *Proc. of the SPIE Int. Conference on Security, Steganography, and Watermarking of Multimedia Contents IX*, volume 6505, page 65050U.

Khanna, N., Mikkilineni, A. K., Chiu, G. T., Allebach, J. P., Delp, E. J., et al. (2007b). Scanner identification using sensor pattern noise. In *Proceedings of the SPIE Int. Conference on Security, Steganography, and Watermarking of Multimedia Contents IX*, volume 6505, page 65051K.

Kharrazi, M., Sencar, H. T., and Memon, N. (2004). Blind source camera identification. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP04)*, volume 1, pages 709–712. IEEE.

Kharrazi, M., Sencar, H. T., and Memon, N. (2006). Improving steganalysis by fusion techniques: A case study with image steganography. In *Trans. on Data Hiding and Multimedia Security*, pages 123–137.

Kirchner, M. (2008). Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *Proc. of the 10th ACM Workshop on Multimedia and Security*, pages 11–20. ACM.

Kirchner, M. (2011). Notes on Digital Image Forensics and Counter-Forensics. Technical report, IT Security Research Group.

Kirchner, M. and Böhme, R. (2007). Tamper hiding: Defeating image forensics. In *Information Hiding*, pages 326–341. Springer.

Kirchner, M. and Bohme, R. (2008). Hiding traces of resampling in digital images. *IEEE Trans. on Information Forensics and Security*, 3(4):582–592.

Kirchner, M. and Böhme, R. (2009). Synthesis of color filter array pattern in digital images. In *IS&T/SPIE Electronic Imaging*, pages 72540K–72540K. Int. Society for Optics and Photonics.

Kirchner, M. and Fridrich, J. (2010). On detection of median filtering in digital images. In *IS&T/SPIE Electronic Imaging*, pages 754110–754110. Int. Society for Optics and Photonics.

Kittler, J., Hatef, M., Duin, R. P., and Matas, J. (1998). On combining classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(3):226–239.

Kuncheva, L. I. (2007). Combining pattern classifiers: Methods and algorithms. *IEEE Trans. on Neural Networks*, 18(3):964–964.

Lai, S. and Böhme, R. (2011). Countering counter-forensics: the case of JPEG compression. In *Information Hiding*, pages 285–298. Springer.

Lam, L. and Suen, S. (1997). Application of majority voting to pattern recognition: an analysis of its behavior and performance. *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans*, 27(5):553–568.

Leekwijck, W. and Kerre, E. (1999). Defuzzification: criteria and classification. *Fuzzy sets and systems*, 108(2):159–178.

Li, C.-T. (2010). Source camera identification using enhanced sensor pattern noise. *IEEE Trans. on Information Forensics and Security*, 5(2):280–287.

Li, H., Luo, W., and Huang, J. (2012). Countering anti-JPEG compression forensics. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP12)*, pages 241–244. IEEE.

Lin, X., Li, C.-T., and Hu, Y. (2013). Exposing image forgery through the detection of contrast enhancement. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP13)*. IEEE.

Lin, Z. C., He, J. F., Tang, X., and Tang, C. K. (2009). Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis. *Pattern Recognition*, 42:2492–2501.

Lindeberg, T. (1994). Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2):225–270.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *Int. journal of computer vision*, 30(2):79–116.

Lopes, A. P., de Avila, S. E., Peixoto, A. N., Oliveira, R. S., and Araújo, A. d. A. (2009). A bag-of-features approach based on hue-SIFT descriptor for nude detection. In *Proc. of the 17th European Signal Processing Conference*, pages 1552–1556. Citeseer.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proc. of the 7th IEEE Int. Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. journal of computer vision*, 60(2):91–110.

Lu, C.-S. and Hsu, C.-Y. (2012). Constraint-optimized keypoint inhibition/insertion attack: security threat to scale-space image feature extraction. In *Proc. of the 20th ACM Int. Conference on Multimedia*, pages 629–638. ACM.

Lu, Y. (1996). Knowledge integration in a multiple classifier system. *Applied Intelligence*, 6(2):75–86.

Lukas, J., Fridrich, J., and Goljan, M. (2006). Digital camera identification from sensor pattern noise. *IEEE Trans. on Information Forensics and Security*, 1(2):205–214.

Luo, W., Huang, J., and Qiu, G. (2010). JPEG error analysis and its applications to digital image forensics. *IEEE Trans. on Information Forensics and Security*, 5(3):480–491.

Luo, W., Qu, Z., Huang, J., and Qiu, G. (2007). A novel method for detecting cropped and recompressed image block. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP07)*, volume 2, pages II–217. IEEE.

Lv, X. and Wang, Z. (2012). Perceptual image hashing based on shape contexts and local feature points. *IEEE Trans. on Information Forensics and Security*, 7(3):1081–1093.

Mahdian, B. and Saic, S. (2008). Blind authentication using periodic properties of interpolation. *IEEE Trans. on Information Forensics and Security*, 3(3):529–538.

Mahy, M., Van Eycken, L., and Oosterlinck, A. (1994). Evaluation of uniform color spaces developed after the adoption of CIELAB and CIELUV. *Color research and application*, 19(2):105–121.

Mamdani, E. and Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *Int. Journal of Man-Machine Studies*, 7(1):1–13.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767.

McKay, C., Swaminathan, A., Gou, H., and Wu, M. (2008). Image acquisition forensics: Forensic analysis to identify imaging source. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP08)*, pages 1657–1660. IEEE.

Mihçak, K. M., Kozintsev, I., and Ramchandran, K. (1999). Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE Int. Conference on*, volume 6, pages 3253–3256. IEEE.

Mikolajczyk, K. (2002). *Detection of local features invariant to affines transformations*. PhD thesis, Grenoble.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630.

Milani, S., Tagliasacchi, M., and Tubaro, S. (2012). Discriminating multiple JPEG compression using first digit features. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP12)*, pages 2253–2256. IEEE.

Milani, S., Tagliasacchi, M., and Tubaro, S. (2013). Antiforensics attacks to benford's law for the detection of double compressed images. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP13)*. IEEE.

Ng, T.-T., Chang, S.-F., Hsu, J., Xie, L., and Tsui, M.-P. (2005). Physics-motivated features for distinguishing photographic images and computer graphics. In *Proc. of the 13th annual ACM Int. Conference on Multimedia*, pages 239–248. ACM.

Nguyen, H. C. and Katzenbeisser, S. (2011). Security of copy-move forgery detection techniques. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP11)*, pages 1864–1867.

Pan, X. and Lyu, S. (2010). Region duplication detection using image feature matching. *IEEE Trans. on Information Forensics and Security*, 5(4):857–867.

Park, U., Pankanti, S., and Jain, A. (2008). Fingerprint verification using SIFT features. In *SPIE*, volume 6944, page 69440K.

Pele, O. and Werman, M. (2010). The quadratic-chi histogram distance family. In *Proc. of the European Conference on Computer Vision (ECCV10)*, pages 749–762. Springer.

Perona, P. and Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):629–639.

Piva, A. (2013). An overview on image forensics. *ISRN Signal Processing*, 2013.

Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advance in large margin classifiers*, pages 61–74. MIT Press.

Popescu, A. C. and Farid, H. (2004). Exposing digital forgeries by detecting duplicated image regions. *Dept. Computer Science, Dartmouth College, Tech. Rep. TR2004-515*.

Popescu, A. C. and Farid, H. (2005). Exposing digital forgeries in color filter array interpolated images. *IEEE Trans. on Signal Processing*, 53(10):3948–3959.

Qian, Z. and Zhang, X. (2012). Combating anti-forensics of JPEG compression. *Int. Journal of Computer Science Issues (IJCSI)*, 9(3):454–459.

Rajeev Ramanath, A., Snyder, B. W., and Hinks, C. D. (2002). Image comparison measure for digital still color cameras. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP02)*, volume 1, pages I–629. IEEE.

Rao, Q., , Li, H., Luo, W., and Huang, J. (2013). Anti-forensics of the triangle test by random fingerprint copy-attack. In *Proc. of the Computational Visual Media Conference (CVM13)*.

Redi, J. A., Taktak, W., and Dugelay, J.-L. (2011). Digital image forensics: a booklet for beginners. *Multimedia Tools and Applications*, 51(1):133–162.

Reinhard, E., Adhikhmin, M., Gooch, B., and Shirley, P. (2001). Color transfer between images. *IEEE Computer Graphics and Applications*, 21(5):34–41.

Rosten, E. and Drummon, T. (2005). Fusing points and lines for high performance tracking. In *Proc. of the IEEE Int. Conference on Computer Vision (ICCV05)*, pages 1508–1511.

San Choi, K., Lam, E. Y., and Wong, K. K. (2006). Automatic source camera identification using the intrinsic lens radial distortion. *Optics express*, 14:11551–11565.

Sarkar, A., Nataraj, L., and Manjunath, B. (2009). Detection of seam carving and localization of seam insertions in digital images. In *Proc. of the 11th ACM Workshop on Multimedia and security*, pages 107–116. ACM.

Schaefer, G. and Stich, M. (2003). UCID: an uncompressed color image database. In *Electronic Imaging 2004*, pages 472–480. Int. Society for Optics and Photonics.

Schiele, B. and Crowley, J. L. (2000). Recognition without correspondence using multidimensional receptive field histograms. *Int. Journal of Computer Vision*, 36(1):31–50.

Schoen, R. and Yau, S.-T. (1994). *Lectures on differential geometry*, volume 2. Int. press Cambridge.

Sencar, H. T. and Memon, N. D. (2013). *Digital Image Forensics: There is More to a Picture Than Meets the Eye*. Springer.

Shafer, G. (1976). *A mathematical theory of evidence*, volume 1. Princeton university press.

Sheet, D., Garud, H., Suveer, A., Mahadevappa, M., and Chatterjee, J. (2010). Brightness preserving dynamic fuzzy histogram equalization. *IEEE Trans. on Consumer Electronics*, 56(4):2475–2480.

Shi, J. and Tomasi, C. (1994). Good features to track. In *Proc. of the IEEE Int. Conference on Computer Vision and Pattern Recognition (CVPR94)*, pages 593–600. IEEE.

Shih, T. K. and Chang, R.-C. (2005). Digital inpainting-survey and multilayer image inpainting algorithms. In *Proc of the IEEE Int. Conference on Information Technology and Applications (ICITA05)*, volume 1, pages 15–24. IEEE.

Sivanandam, S., Sumathi, S., and Deepa, S. (2007). *Introduction to fuzzy logic using MATLAB*. Springer Verlag.

Sorzano, C. O. S., Thevenaz, P., and Unser, M. (2005). Elastic registration of biological images using vector-spline regularization. *IEEE Trans. on Biomedical Engineering*, 52(4):652–663.

Stamm, M. and Liu, K. R. (2008). Blind forensics of contrast enhancement in digital images. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP08)*, pages 3112–3115. IEEE.

Stamm, M. C., Lin, W. S., and Liu, K. (2012). Forensics vs. anti-forensics: A decision and game theoretic framework. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP12)*, pages 1749–1752. IEEE.

Stamm, M. C. and Liu, K. R. (2010). Wavelet-based image compression anti-forensics. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP10)*, pages 1737–1740. IEEE.

Stamm, M. C., Tjoa, S. K., Lin, W. S., and Liu, K. R. (2010a). Anti-forensics of JPEG compression. In *Proc. of the IEEE Int. Conference on Acoustics Speech and Signal Processing (ICASSP10)*, pages 1694–1697. IEEE.

Stamm, M. C., Tjoa, S. K., Lin, W. S., and Liu, K. R. (2010b). Undetectable image tampering through JPEG compression anti-forensics. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP10)*, pages 2109–2112. IEEE.

Sugeno, M. (1974). Theory of fuzzy integrals and its applications (phd thesis). *Tokyo Institute of Technology*.

Sugeno, M. (1985). *Industrial Applications of Fuzzy Control*. Elsevier Science Inc., New York, NY, USA.

Swain, M. J. and Ballard, D. H. (1991). Color indexing. *Int. journal of computer vision*, 7(1):11–32.

Tai, Y.-W., Jia, J., and Tang, C.-K. (2005). Local color transfer via probabilistic segmentation by expectation-maximization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR05)*, volume 1, pages 747–754. IEEE.

Tanaka, K. (1997). Mechanisms of visual object recognition: monkey and human studies. *Current opinion in neurobiology*, 7(4):523–529.

Terano, T., Asai, K., and Sugeno, M. (1992). *Fuzzy Systems Theory and its Applications*. Academic Press Boston.

Tuytelaars, T. and Mikolajczyk, K. (2008). Local invariant feature detectors: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280.

Valenzise, G., Nobile, V., Tagliasacchi, M., and Tubaro, S. (2011a). Countering JPEG anti-forensics. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP11)*, pages 1949–1952. IEEE.

Valenzise, G., Tagliasacchi, M., and Tubaro, S. (2011b). The cost of JPEG compression anti-forensics. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP11)*, pages 1884–1887. IEEE.

Valenzise, G., Tagliasacchi, M., and Tubaro, S. (2013). Revealing the traces of JPEG compression anti-forensics. *IEEE Trans. on Information Forensics and Security*, 8(2):335–349.

Van, L. T., Emmanuel, S., and Kankanhalli, M. S. (2007). Identifying source cell phone using chromatic aberration. In *Proc. of the IEEE Int. Conference on Multimedia and Expo (ICME07)*, pages 883–886. IEEE.

Van De Sande, K. E., Gevers, T., and Snoek, C. G. (2010). Evaluating color descriptors for object and scene recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596.

Van De Weijer, J. and Schmid, C. (2006). Coloring local feature extraction. In *Proc. of the European Conference on Computer Vision (ECCV06)*, pages 334–348. Springer.

Vedaldi, A. (2007). An open implementation of the SIFT detector and descriptor. Technical Report 070012, UCLA CSD.

Vedaldi, A. and Fulkerson, B. (2010). VLFeat: an open and portable library of computer vision algorithms. In *Proc. of the ACM Int. Conference on Multimedia*, pages 1469–1472. ACM.

W, L., J, H., and G, Q. (2006). Robust detection of region duplication forgery in digital image. In *Proc. of the IEEE Int. Conference on Pattern Recognition (ICPR06)*. IEEE.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Processing*, 13(4):600–612.

Weickert, J. (1999). Coherence-enhancing diffusion filtering. *Int. Journal of Computer Vision*, 31(2):111–127.

Weickert, J. and Scharr, H. (2002). A scheme for coherence-enhancing diffusion filtering with optimized rotation invariance. *Journal of Visual Communication and Image Representation*, 13(1):103–118.

Weinzaepfel, P., Jégou, H., and Pérez, P. (2011). Reconstructing an image from its local descriptors. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR11)*, pages 337–344. IEEE.

Welsh, T., Ashikhmin, M., and Mueller, K. (2002). Transferring color to greyscale images. *ACM Trans. on Graphics*, 21(3):277–280.

Wu, Q., Sun, S., Zhu, W., and Li, G. (2009). Identification of inpainted images and natural images for digital forensics. *Journal of Electronics (China)*, 26(3):341–345.

Wyszecki, G. and Stiles, W. S. (1982). *Color science*. Wiley New York.

Xu, G., Shi, Y. Q., and Su, W. (2009). Camera brand and model identification using moments of 1-D and 2-D characteristic functions. In *Proc. of the IEEE Int. Conference on Image Processing (ICIP09)*, pages 2917–2920. IEEE.

Xu, L., Krzyzak, A., and Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man and Cybernetics*, 22(3):418–435.

Yao, H., Wang, S., Zhao, Y., and Zhang, X. (2012). Detecting image forgery using perspective constraints. *IEEE Signal Processing Letters*, 19(3):123–126.

Yerushalmy, I. and Hel-Or, H. (2011). Digital image forgery detection based on lens and sensor aberration. *Int. journal of computer vision*, 92(1):71–91.

Yu, G. and Morel, J.-M. (2011). ASIFT: An algorithm for fully affine invariant comparison. *Image Processing On Line*, 2011.

Yuan, H.-D. (2011). Blind forensics of median filtering in digital images. *IEEE Trans. on Information Forensics and Security*, 6(4):1335–1345.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8:338–353.

Zadeh, L. A. (1973). Outline of a new approach to the analysis of complex systems and decision. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-3:28–44.

Zeng, X. and Keane, J. (2006). Learning for hierarchical fuzzy systems based on the gradient-descent method. In *Proc. of IEEE Int. Conference on Fuzzy Systems*, pages 92–99.

Zhang, D.-Q. and Chang, S.-F. (2004). Detecting image near-duplicate by stochastic attributed relational graph matching with learning. In *Proc. of the 12th annual ACM Int. Conference on Multimedia*, pages 877–884. ACM.

Zhang, W., Cao, X., Zhang, J., Zhu, J., and Wang, P. (2009). Detecting photographic composites using shadows. In *Proc. of the IEEE Int. Conference on Multimedia and Expo (ICME09)*, pages 1042–1045. IEEE.

Zhu, J., Hoi, S. C., Lyu, M. R., and Yan, S. (2008). Near-duplicate keyframe retrieval by nonrigid image matching. In *Proc. of the 16th ACM Int. Conference on Multimedia*, pages 41–50. ACM.

Zitová, B. and Flusser, J. (2003). Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000.

Zuiderveld, K. (1994). Contrast limited adaptive histogram equalization. In *Graphics gems IV*, pages 474–485. Academic Press Professional.

Index

Author's information

Andrea Costanzo was born in Florence (Italy) on August 28th 1982. He qualified from Liceo Classico E.S. Piccolomini of Siena, where he gave his unconditional love to Italian and French literature, to ancient Greek and Latin, to History, Philosophy and History of the Arts. He proudly ignored stone-hearted Mathematics, Geometry and Physics, at least as much as they ignored him. Hardly flattening scores were effortlessly achieved, such as "*4 out of 10*", "*No, seriously?*", "*. . . . . .*" and "*Esteemed Mr Costanzo, is your son joking with me?*". Misled by those who once he believed friends, blinded by fabled easy profit, he enrolled at the faculty of Engineering of the University of Siena. Mind set on task, he dreadfully abandoned a brilliant football career, at least according to his mother. He regretted both choices for years but through cunning and bribery he graduated *cum laude* in Telecommunications in 2008, with a thesis on something technical. In 2009, he became a Ph.D. student in Something More Technical; his first task consisted on the visual discrimination between the few images viable for papers and nakednesses of all sorts inefficiently gathered online by absent-minded research partners. The overcrowded laboratory undermined the secrecy of his mission. His research interests are focused on ~~Digital Image Forensics and Counter-forensics~~ acid rants against the entire world in general. He participated actively to projects funded by the European Community, during whose general meetings he experienced the heights of Viennese pastry and the depths of Cola & After Eight. He mastered the usage of infinite `while` loops to artificially delay experimental results so that he could level up his World of Warcraft's Horde shaman, paladin, druid and rogue (no computers were harmed in the process). He was visiting student of his own laboratory because the idea of travelling disturbs him very much. Despite his ambitious dreams, no one kneels when he enters a bank unmasked and unarmed.

It's all so easy with Photoshop. With imaging software so widely available, the manipulation of digital images is not anymore a matter for experts only. While there is little harm besides gossip in retouching an unwanted belly or an incipient baldness, the simplicity of counterfeiting is a serious issue when it is exploited to convey social, political or military messages. It is not surprising, then, that restoring the credibility of digital content has become a task of paramount importance. *Digital Image Forensics* is a science allowing to gather information on the history of an image in such a way that its veracity can be evaluated, based on the principle that any manipulation leaves more or less subtle traces.

We contribute to the image forensics' mission by addressing three open issues. First, we analyse the history of groups of near-duplicate images to reveal their parent-child relationships, thus opening new scenarios for copyright enforcement, news tracking or clustering. Secondly, we make possible the cooperation of heterogeneous image forensic detectors by fusing their decision scores in such a way to deal with uncertainty, incompatibility and noise typically affecting the analysis. Finally, we study strengths and weaknesses of the forensic algorithms based the Scale Invariant Feature Transform; we devise new attacks bypassing the forensic analysis and we discuss possible counter-measures.

UNIVERSITÀ DI SIENA
1240

The Ph.D. School of Information Engineering of the University of Siena is a school aiming at educating scholars in a number of fields of research in the Information Engineering area. The Ph.D. School of Information Engineering is part of the Santa Chiara High School of the University of Siena. A Scientific Committee of external experts recognized Ph.D. Schools belonging to Santa Chiara as excellent, according to their degree of internationalization, their research and educational activities.