Challenging the Security of CBIR Systems

Ewa Kijak, Teddy Furon, Laurent Amsaleg

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Challenging the Security of CBIR Systems

Ewa Kijak — Teddy Furon — Laurent Amsaleg

N° 7153

Décembre 2009

ISSN 0249-6399 ISRN INRIA/RR--7153--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
RENNES - BRETAGNE ATLANTIQUE

Challenging the Security of CBIR Systems

Ewa Kijak , Teddy Furon , Laurent Amsaleg

Thème :
Équipes-Projets Texmex et Temics

Rapport de recherche n° 7153 — Décembre 2009 — 23 pages

Abstract: Content-Based Image Retrieval Systems are now commonly used as a filtering mechanism against the piracy of multimedia contents. Many publications in the last few years have proposed very robust schemes where pirated contents are detected despite severe modifications. But none of these systems have addressed the piracy problem from a *security* perspective. It is now time to check whether they are secure: Can pirates mount violent attacks against CBIRS by carefully studying the technology they use? This paper analyzes the security flaws of the typical technology blocks used in state-of-the-art CBIRS and shows it is possible to delude systems, making them useless in practice.

Key-words: Security, Content-Based Image Retrieval Systems, Robust Content Detection, security analysis

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

Mettre à l'épreuve l'aspect sécurité des systèmes de recherche d'images par le contenu

Résumé : Les systèmes de recherche d'images par le contenu sont maintenant souvent à la base de filtres destinés à contrer le piratage de contenus multimédias. Depuis quelques années, de nombreuses travaux ont proposé des systèmes de reconnaissance très robustes où ces contenus piratés, même sévèrement modifiés, peuvent être détectés. Néanmoins, aucun système n'a envisagé ce problème de piratage du point de vue de la sécurité. Il est temps de le faire : est-ce que les pirates peuvent mettre au point des attaques violentes contre ces systèmes après avoir finement étudié les moindres détails des technologies qu'ils emploient? Cet article analyse les failles de sécurité des briques technologiques typiques utilisées dans les systèmes de reconnaissance de l'état de l'art et montre qu'il est alors possible de les tromper, les rendant inutiles en pratique.

Mots-clés : Sécurité, systèmes de recherche d'images par le contenu

1 Introduction

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

Here are two recent applications of such content recognition that reached the public and are becoming quite popular: anyone can download a small application for Apple's iPhone that does song recognition. It uses low level audio signatures and content-based similarity searches. Also, anyone can now create superb image panoramas with fully automatic geometrical and color corrections. It uses local image descriptor type of content-based techniques.

Overall, CBR systems have so far been used in very cooperative and "friendly" settings where it benefits content providers business, while increasing users digital experience enjoyment.

1.1 From Friendly to Hostile Environments

However, we recently witness another use of this technology. CBR is used to *filter* multimedia contents in order to protect the creation of the few from the piracy of the many. CBR techniques are used to "clean the Internet", stopping the upload of copyrighted material on User Generated Contents (UGC) sharing platforms such as YouTube, or forbidding downloads from P2P networks.

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

Overall, filtering is an application of CBR techniques that is quite different from its primary goal: the environment is now hostile in the sense that filtering restricts users freedom, controlling and/or forbidding distribution of content.

Filtering typically requires to extract low-level signatures from multimedia contents and to query a database containing the material to protect. Alarms suggesting copyright infringements are raised when matches are found. Various *Content-Based Image Retrieval Systems* [12, 20, 11] (CBIRS) enforcing copyright protection for images and videos have been recently proposed. Evaluating their efficiency is so crucial in the real-world that a specific track has been created in the latest TrecVid challenge.

CBIRS in hostile environments are also used as a pre-processing step facilitating the "side informed" watermarking detection. The Achilles' heel of digital

watermarking is its robustness against geometrical attacks (cropping, stretching, change of aspect ratio, etc) and the lack of diversity in the secret keys. Here, by identifying the original version of the content under scrutiny, CBIRS allows the resynchronization of the two versions (ie. inverting the geometrical attack). It could also tell the detector, thanks to a log file, which secret key was used at the watermarking stage. This allows periodical changes of the secret keys. In short, CBIRS compensate some of the flaws of digital watermarking.

In all these new applications, systems no longer magnify cultural richness, but protect the commercial value of contents. Because there are valuable goods to protect, there are serious hackers willing to circumvent the system. Therefore, it is legitimate to carefully investigate the *security* side of content-based retrieval system—this is the goal of this paper, focusing on CBIRS.

1.2 Security is not Robustness

It is necessary to clarify upfront the concept of *security* since robustness is a very similar notion masking what is at stake with security. Mixing security and robustness confused the digital watermarking community in the past [2].

Content identification is deemed *robust* if the system succeeds in recognizing contents despite modifications. The system must offer this feature because contents are often touched: for instance, cropping, stretching, color adjustment and lossy compression are common image modifications when copying contents.

Security is different in many points. A pirate is of course operating with the malicious purpose of deluding the system. He doesn't use classical image processing tools such as the ones provided by a photo editor software. He mounts his own attack, a process dedicated to delude a particular content identification technique. He doesn't blindly lead an attack, instead he first observes and accumulates knowledge about the details of the CBIR techniques and then focuses attacks on very specific parts of the system where flaws have been identified. Security attacks have in general bigger success rates thanks to the knowledge stolen by the pirates. They can invent much trickier scenarios than forging content escaping from recognition. For instance, the pirates might create dummy contents rising so many false alarms that they pull down the reliability of the filter.

1.3 Contributions

While the cryptographic community has been investigating the security of systems for decades, almost no publication address this issue in the computer-vision community. This paper sheds a security-oriented light on the typical components building a complete CBIRS. However, we are not pirates. Our goal is to warn the community that security is of utmost importance when diving this technology into already existing hostile environments. This paper makes three major contributions:

Threat Analysis for CBIRS

The first contribution is a presentation of the crucial elements to consider in order to assess the security of CBIRS. This is based on the typical threat analysis frameworks from digital watermarking and cryptography, adapted to the specific

context of CBIRS. Analysing the security of a system asks to follow a rigorous methodology aiming at decoupling the security threats on the whole system from the security flaws of the core technologies under scrutiny.

Attacking the Core Technologies of CBIRS

The second contribution is a discussion about the building blocks forming CBIRS from a security perspective. We survey the major state-of-the-art core technology blocks found today in CBIRS and present an initial set of potential security flaws for each block.

Proofs of Concept

This paper experimentally shows, through 3 illustrations, that it is possible, for a pirate well aware of the specific techniques used in particular systems, to pull down the recognition capabilities. This is, however, initial work, and we believe that many other configurations are much harder to break. Our goal is to show that challenging the security of CBIRS is feasible in practice, and therefore this deserves strong and continuous effort from the research community; an effort that we instigate in this paper.

This paper is structured as follows. Section 2 provides three motivational scenarios where the security of CBIRS plays a central role and where pirates challenge systems in various ways. Section 3 describes the key concepts of security applied to CBIRS and presents the resulting threat analysis framework. Section 4 links the various components building a complete CBIRS to potential security flaws. Section 5 describes three experiments showing it is possible to delude existing techniques. Section 6 concludes.

2 Examples Challenging Security

This section gives three examples where pirates challenge the security of CBIRS—their goal for hijacking valuable contents differs as they play a different role in the system.

2.1 Content Concealment

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

2.2 Abnormally Frequent Identifications

Thanks to photo portals, users can now buy beautiful pictures on-line. Once the picture of interest found, they can receive a high-quality printed poster in their surface mail box. The typical process starts with checking a first page of thumbnails, and, with the help of keywords, relevance feedback and/or visual similarity searches, iterative refinements eventually isolate the picture to purchase. Once bought, various people (the photographer, portal keepers, ...) get payed. As pirates tweak HTML pages to get ranked higher in textual search engines (this is known as "black hat SEO"—Search Engine Optimization), a pirate can tweak the visual contents of images such that they always get (artificially) ranked high in the result list of similarity searches, or, in contrast, tweak other images such that their ranks get lowered. In this case, the pirate can be someone working for the portal, and who receives dirty money by secretly favoring a particular photographer or photo agency. Note similar effect can be produced by a pirate spying the communication channels and inserting well chosen contents on-the-fly in the image result list sent back to the user.

2.3 Denial of Service

Many UGC platforms include a content filter and advertise a lot about this to increase the confidence level of right holders. That filter scans whatever users try to upload and rejects the material detected as illegal. A pirate, or many pirates, in the quite realistic case of a collusion, can bombard one particular platform with material that will trigger infringement alarms.

That material can simply be forged from ripped copies of the latest blockbuster (or any other contents known to match with a significant part of the signatures in the database, which movies are protected is probably known). In addition to the costly detection phase, processing many alarms can produce a denial of service as the system is overloaded. This is particularly true if the material to reject is very difficult to detect, therefore consuming many resources at the server due to extensive analysis, but clear enough to be eventually recognized. Here, producing the material with the distortion level maximizing the time and energy spent for its recognition is the key. If a counter measure is to reduce the severity of the filter, with the hope of fewer alarms, then the door gets opened for uploading illegal material.

3 Threat Analysis for CBIRS

This section gives a brief overview of the main concepts underpinning the theory of security of systems. It adapts the threat analysis frameworks from digital watermarking and cryptography to the special case of CBIRS.

It is very important to clarify who are the actors playing a role in the system, what are the assets to protect, what are the goals of the pirates, what type of knowledge on the system's internal they can get, and whether they can get access to some of its building blocks found off-the-shelf (or elsewhere). Even if the exact same core technology was used in all systems, the conclusions of the security analyses would differ depending on the nature of the application, on the target chosen by the pirate, on the level of details he has on the specific techniques used in the system and whether he has any role inside the system

itself (as in the photo portal example above). There are at least the following four important classes of assumptions that need to be clarified, detailed next.

3.1 Trust Model

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

1. **The image right-holder** who is entitled to upload his Works or their low-level signatures in the database.
2. **The image server** where a collection of signatures (for example computed from uploaded images) is indexed in a database used for building the answers of content-based queries.
3. **The querying user** who comes up with a particular image to search for.
4. **The client software** which processes the query image, connects to the server, send requests, receives and processes answers before displaying them.

The trust model states which actors honestly play open cards and which actors want to delude the system. There are a priory many possible trust models as any of the four above-mentioned actors, or even worse, a collusion of several of them, can be suspected of dishonesty. The most classical scenario is when the user is the pirate forging illegal copies to conceal from the system, all other actors being trusted. An other trust model is, for instance, a right-holder modifying his contents in order to increase the recognition rate because he receives incentives whenever the query is deemed to be a copy of his Works (see section 2.2).

3.2 Goals

The goals of the pirates might also be very different from an application to the other. In the context of CBIRS, we can easily identify two main goals that are (this list is non-exhaustive, however):

1. **Producing false negatives**. The pirate manipulates multimedia material that the CBIRS fails to detect. This encompasses two situations. Either the query is manipulated (section 2.1), either some documents in the database are manipulated: when the server is not trusted, artificially deleting some relevant matches from the result list before sending it back to the client indeed produces some false negatives (see section 2.2).
2. **Producing false positives**. The pirate manipulates multimedia material such that it will always be detected by the system (even truly innocuous contents). This also encompasses two situations: the pirate produces problematic queries triggering a denial of service (Section 2.3), or the manipulated data belongs to the database to artificially multiply matches (Section 2.2).

3.3 Knowledge

The assumption on what the pirate knows from the system is crucial as any security system fundamentally depends on keeping *some* things secret.

3.3.1 Kerckhoffs' Principle

Generally speaking, there are two broad approaches to keep things secret: the first approach maintains security of the system through obscurity. The second approach maintains security through the use of cryptographic-based techniques defining a secret key [9].

Security through obscurity relies on the fact that pirates may have very hard time to precisely know what are the algorithms used in the system, what are the implementation details, what are the parameters and what can be their values. In other words, security of systems through obscurity assumes the pirates are unlikely to find the security flaws due to the great complexity of the system they are attacking.

It has been demonstrated that solely relying on obscurity is not reliable. Pirates look for any piece of information in publications, patents, standards or by social engineering. It is impossible to empirically assess how difficult it is to disclose information about a system.

The second drawback of security through obscurity is the high cost of changing the algorithm if it gets disclosed. Designers need to re-implement another obscure algorithm, likely way different, with heavy testing phases and a high burden for deploying the algorithm on sites. Furthermore, this takes time during which the system remains insecure.

These problems have been reported since a long time and Kerckhoffs came up with several design principles still applicable today [9]. His best known principle says that the system must remain secure even if everything about the system, except a secret key, is public knowledge. A secret key is a piece of information that determines the output of a particular algorithm: the same algorithm does not produces the same output if it is fed with the same data but with different keys. This guarantees the security of the system.

As it is much easier to protect a small piece of information (the secret key) than a complete system by obscurity, secure systems using secret keys are much more reliable. A secret key is one or more very large random number. Finding its value by an exhaustive search is almost impossible as the key space is very large. If the secret key is discovered, then forging another key is easy and fast. Real-world secure systems typically include secret keys and elements of obscurity.

Knowing the algorithm and/or the secret key allows to define two neighboring concepts coming from the digital watermarking community: for a given secure system, the *worst case attack* is the most efficient attack when the pirate knows the algorithm but not the secret key [24]. Efficiency is typically measured by the loss of quality versus the increase in probability of watermarking decoding errors. A *security hack* is different in the sense that the pirate first takes time to analyze observations in order to estimate the missing information such as the secret key, and then leads an attack based on this stolen knowledge [2].

3.3.2 Kerckhoffs and CBIRS

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

Secret-based CBIRS Very few CBIRS use a secret key. Some use the secret key to generate a private selection of parts of the contents [10] or of parts of its features. Another approach defines a secret transform which extracts some private features [15]. Both [8, 23] define a secret quantifier used to quantize the extracted features.

In this context, the worst case attack is the best process in terms of probability of successfully changing the robust hash, and quality of the manipulated content. On the other hand, a security hack is possible by observing pairs of an image and its robust hash. This attack framework can be entitled KIA for *Known Image Attack*, a terminology coming from the cryptanalysis: in the Known Plaintext Attack, the adversary observes pairs of plain and cipher texts. The assessment evaluates the security level as the number of pairs needed to disclose the secret key with a given accuracy. This is also known as the unicity distance [16]. Then, a second step is to mount the attack itself which takes full advantage of the disclosed key to forge pirated images.

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

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

3.4 Access to Oracles

In cryptanalysis and digital watermarking security assessments, there is a class of attack named *oracle attacks*. The pirate has an unlimited access to a piece of software that is part of the system under scrutiny—this piece is said to be an oracle. It may even be a sealed black box process. What matters is the output and the total freedom to run that software on any arbitrary, yet well chosen, input. With CBIRS, one oracle could be the software calculating local descriptors, or the filtering system itself giving a binary decision. There are two families of oracle attacks: the *Chosen Image Attack* (CIA) [3] and the *Closest Point Attack* (CPA) [4].

Chosen Image Attack In this case, the pirate aims at disclosing a secret by challenging the oracle. The difference with the KIA (see 3.3.2) is that the pirate is allowed to choose the images sequentially. For secret-based CBIRS, the pirate iteratively creates an image, observes its robust hash and gain knowledge about the secret key with the minimum number of calls to the oracle. For non secret-based CBIRS, the pirate can tests whether a given image feature, such as the chrominance channels, play any role in the search.

Closest Point Attack Here, the pirate is not interested in disclosing any secret. He has a content which is filtered out by the system, and his goal is to forge the least degraded copy of this content which will not be blocked by the filter. In the content space, the content is a point belonging to the acceptance region of the filter. A very degraded copy is another point lying outside this region. The question is about the fastest iterative process increasing the quality of the pirated copy. For instance, as a toy example, within few oracle trials and the help of a dichotomy search, the pirate easily finds the content which lies at the boundary of the acceptance region.

Interesting results are known in the watermarking community: there exist algorithms which do need any assumption about the shape of the acceptance region, or in other words, about the technique used by the filter [4]. The increase in quality is very fast at the beginning (ie. in the first hundreds of trials) and then it is going very slowly requiring millions of trials to get any substantial quality gain [6].

4 Security of CBIRS

CBIRS can all be decomposed into a small set of building blocks and this section will try to emphasize where, in each block, pirates are likely to concentrate their attacks. This section discusses security issues for the communication channels between the client and the server, and, more importantly, provides extensive details on the potential security flaws at the server, where the core technology is.

At the server, the key elements of a CBIRS are:

- **The image description technique**. It extracts low-level descriptions from the images. Descriptions are typically high-dimensional vectors compared using a specific metric (often L_2).
- **The indexing and retrieval techniques**. These two elements are very hard to decouple as they are designed jointly to provide the maximum performance. The indexing technique inserts the descriptions in a particular data structure. The retrieval technique navigates very efficiently in that structure to retrieve the k-nearest neighbors or to do an ε-range search for each low-level query descriptor.
- **Typical optimizations**. Many CBIRS use various optimizations to improve their response time and/or the quality of their result. Note these optimizations are optional and are not all implemented in systems.

We now go through each item, describe the typical state-of-art technology used as implementation, and discuss the resulting security flaws.

4.1 Communication Channels

A pirate can eavesdrop on or modify the communications between the remote client and the server. Therefore, this channel must be secure. Modern cryptography solves the problem with entity authentication and session key agreement protocols. Although this works for trust models fearing message interception on the line, it is useless whenever the pirate is the user. The pirate can spy or modify the data just before entering the secure channel, analyzing the volatile memory of the computer. He might also reverse engineer the object code of the client software into understandable source code to learn elements of the technique. Secure coding and obfuscation code are the defense walls, but they rely on adhoc techniques providing no proven security.

4.2 Image Description

CBIRS typically represent images by one or more high-dimensional vectors describing some low-level visual features of images. A large variety of description schemes exist, based on colors, textures, edges or any combination. Some schemes are based on image transforms like Fourier, wavelets, DCT. When a pirate decides to attack that part of the system, its goal is to produce specific modifications of the images entirely driven by the deep understanding of the properties of the description scheme, such that matches will subsequently fail. We therefore briefly survey the two main families of descriptors and highlight potential attacks.

When the description algorithm yields a single descriptor from the entire image, the descriptor is referred to as *global descriptor*. Usually these descriptors are quantized for the purpose of compactness and robustness, producing a compact and discriminative *robust hash*. These techniques are fast thanks to the low complexity of the algorithms used, but are especially sensitive to cropping and logo insertion. However they are designed to be robust to some global image transformations affecting the feature used for description. Knowing the algorithm used to compute a descriptor, the pirate creates a dedicated transformation maximizing the the perturbation of the descriptor, while minimizing the visual artefacts. For example, the strategy for attacking will be different whether colors or edges are considered for the description. Also, since a quantization process is typically in the loop, playing with the values such that attacked descriptors fall into the wrong bins (with respect to the non attacked image) is an option.

When multiple descriptors are obtained from a single image, each vector describing different image sub-region, the descriptors are called *local descriptors*. Local descriptors are less sensitive to artefacts such as background variations, clutter, crops. These descriptors are particularly interesting for partial querying (only a part of image is of interest, e.g. the case of seeking for a logo). Local image descriptors are usually obtained using a two step approach. During the first step, sub-regions of images (called support regions) are extracted. These regions can be defined by covariant regions, or by the neighborhood of interest points [19, 17]. During the second step, high-dimensional descriptors are extracted using the contents of the support regions. One of the most popular descriptor is the SIFT descriptor [14] where regions are key points, but other approaches exist [1, 18].

That two phase extraction process gives several handles to pirates for attacking local description schemes: either perturbing the region extraction, or the descriptor computation, or both. With the descriptor computation, the knowledge of support regions is important as they give the exact location where to apply image modifications when attacking. For example, a local description scheme constructing an histogram of orientation gradients can be impacted when artificially inserting small edges in the support regions.

With region detection, a pirate may want to artificially add new regions, delete or move detected regions by applying local well-chosen modifications. This affects strongly the detection as even a small change in the support region may dramatically change the descriptor. Artificially triggering the detection of new regions by patching the images in turn create some new descriptors. Therefore, the resulting cost of the retrieval process also increases, leading to potential denials of service attacks (Section 2.3). In addition, the new descriptors are unlikely to correctly match with indexed descriptors at search time, impacting the accuracy of results. In contrast, patching images to delete regions prevent the system from recognizing some particular objects in images, like for example advertisements, reducing royalties. This can be done by modifying the visual neighborhood of the object to hide, such that no regions of interest can be detected.

4.3 Indexing and Retrieval

Without any loss of generality, all indexing and retrieval schemes take advantage of some form of partitioning of the data collection into cells. At query time, instead of analysing the whole descriptor collection, confining the search to few cells provides efficiency [21].

Depending on the partitioning technique, cells may overlap in space, or not. At run time, the query point is used to quickly determine a list of cells. Cells are taken in a certain order, and the contents of each cell is analysed, i.e., real distances between the vectors in the current cell and the query vector are computed, which in turn may update the current query result set. At one point, the system decides that the current result set will not be improved by analysing more cells or vectors [13, 7]. The search is then stopped. Modern indexing and retrieval schemes are performing *approximate* searches, where result quality is traded for reduced query execution time.

Overall, the distribution of the feature vectors in space impacts the major indexing decisions. One decision is where to create the partition borders to have as few frontier problems as possible since data points close in space may get separated by a partition border and assigned to two different cells. To compensate for that problem, several (neighboring) cells are analysed during the processing of one query, which directly impacts its cost. Overlap between cells has similar consequences. In addition, the cost of query processing is also impacted by the number of data points in each analysed cell, and therefore, deciding on ways to control the cardinality of cells is key.

Given these typical technology traits, pirates can envision two grand angles for attacking the "database" part of CBIRS. Since modern scheme are all approximate, one angle is to tweak the data collection such that the approximations made by the system have a more severe impact on the accuracy of the result. The other angle of attack is to increase the cost of queries by forcing extra

processing as the search has more difficulties to find query results and therefore needs to go deeper in the data collection. By significantly increasing that cost, the system may quickly become overloaded and unresponsive, therefore unable to play its role.

There are various possibilities for generating these attacks but they all boil down to modifying the data collection such that specific phenomenons arise. Of course, this can only happen if the pirate has the possibility to upload some material in the CBIRS, for example by inserting images in the collection as in the case of a picture-sharing site with content-based facilities. Directly uploading pirated descriptors is also possible: in a copyright enforcement setting, some content providers refuse to give their images to the party in charge of the protection, but rather compute descriptors home and then deliver them for insertion into the reference collection. Forging pirated descriptors is easy, anywhere along that chain.

Impacting the accuracy and/or the complexity of the indexing and search techniques is possible by artificially exaggerating the likelihood with which the system will run into frontier problems or will have to face severe skews in the cells' cardinality. If the pirate knows about the location of partition borders, he can try to produce descriptors that are slightly shifted in space, such that they get assigned to different partitions, effectively separating near neighbors. This reduces the quality of the result (with [22], being assigned to the incorrect visual word is problematic).

Another option is when the pirate tweaks the data collection such that the cells are abnormally under- or over-filled. Near empty cells generate high overhead since many cells are needed to cover a decent neighborhood. Full cells dramatically increase the cost of the analysis since distances to very many points must be computed. This phenomenon may happen "naturally" due to the normal distribution of data points in high-dimensional spaces, and this is one drawback usually objected to the LSH [5] where certain hash buckets are particularly full. By contracting and dilating the (entire or subsets of the) data collection in space, the pirate can concentrate the whole database in few cells, making the system ineffective in practice.

Simpler tricks can be harmful. Inserting outlier data points in the collection may be very destructive. For some partitioning techniques, outliers greatly increase the overlap between cells, which in turn increases severely the cost of the retrievals. In the case of local description schemes, producing images with very many descriptors used either at index construction time or at query time greatly increase costs. Last, if the CBIR is dynamic, then attacking the insertion policy is an option. For example, some tree-based approaches are concerned with maintaining a balanced data structure for guaranteeing efficiency. Forcing the inserts into a single leaf triggers complex reorganizations, much more expensive than what it is traditionally assumed, when inserts are distributed over the whole index.

When the level of penetration of the pirate into the system is high (Section 2.2), then tweaking the distribution is very easy as any weird value can be inserted in the index even if they do not belong to any real picture, even if these values could not possibly be calculated over real data – which CBIRS is today concerned with integrity constraints?

4.4 Typical Optimizations

To either improve the quality of the results eventually returned to the user and/or to reduce query response times, most CBIRS use various optimizations. We briefly review here three major tricks found in existing systems, and try to show how pirates can divert their effects. We first start by presenting two optimizations aiming at reducing the response times of queries.

To reduce the cost of distance calculations, various approaches apply principal component analysis (PCA) to the data before indexing [7]. PCA captures the global shape of the data cloud, from which a new low-dimensional projection basis is determined. It is therefore possible to attack PCA-based CBIRS by inserting strong outliers artificially distorting the data cloud.

Some CBIRS have been specifically designed to handle local descriptors. In this case, query time is often high because hundreds or even thousands of descriptors computed on the query image are used to probe the database. In [12], special stop-rules have been designed to abort as soon as possible the search process, trying to use only a small subset of the descriptors in the query. In a copyright enforcement scenario, stop-rules assume that matching images receive a lot of votes, while unrelated images receive few random votes, roughly distributed over the whole data collection. Therefore, after having processed a fixed number of query descriptors, it is possible to evaluate the likelihood that a particular image from the collection will receive more votes if more query descriptors are used. If that likelihood is below a threshold for all the candidate database images, then the system terminates query processing and returns a no-match flag. A dual stop-rule aborts the processing if one particular image rapidly collects many votes. In [12], as few as 20 descriptors were found sufficient to find matches, and about 100 descriptors must be used to decide no-match. A possible attack is to try to add to a pirated copy a bunch of non-matching descriptors that will be used first in the querying process. In this case, stop-rule may incorrectly decide no-match, as none of the descriptors describing the true contents to protect are used, the search being stopped before getting to them.

In addition to these mechanisms, CBIRS may include optimizations for filtering false positives. One typical filter relies on checking the geometrical consistency between the query image and the candidate images. Many approaches rely on Hough transforms, which do a good job for matching affine transforms. Two quasi-identical images differing by non affine modifications, however, fail to match. It is therefore possible to attack such schemes by deforming a portion of an image in a non-affine way. While the non-deformed part will match thanks to local descriptors, however, the matching image gets eventually rejected due to the geometrical filter.

5 Experiments

Through three independent experiments, we show in this section that it is possible to delude existing techniques. Here, the working assumptions are the following. The pirate is a dishonest user aiming at producing false negatives, i.e., concealing copies from the CBIRS, and he has a deep knowledge about the techniques used. In addition, systems under scrutiny do not use any secret key. The following experiments each focus on one different part of CBIRS. We first

attack the detection of keypoints, the a global description scheme and finally retrieval part.

5.1 Challenging Keypoint Detection

This subsection deals with a system based on the SIFT local descriptors [14] and focuses only on the keypoint detection, not on descriptor computation, looking for the worst case attack deluding this part.

Keypoint detection relies on local extrema of the Difference-of-Gaussian function $D(x, y, \sigma)$, calculated by:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) \otimes I(x, y) \\ &= \Delta G_\sigma(x, y) \otimes I(x, y), \end{aligned} \tag{1}$$

where $\otimes$ is the 2D convolution operator, and $G(x, y, \sigma)$ is the kernel of the variable-scale Gaussian low-pass filter.

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

- $D(\mathbf{x})$ is a local extrema over a neighborhood of $\mathbf{x}$,
- a sustainable contrast is present, i.e., $|D(\mathbf{x})| > C$ where C is a threshold hard-coded in the algorithm,
- the keypoint is not located on an edge, which can be detected by $\mathrm{tr}(\mathbf{H})^2 / \det(\mathbf{H}) < \tau$, with $\mathbf{H}$ the 2x2 Hessian matrix of $D(\mathbf{x})$.

This gives three opportunities for the pirate to remove a keypoint. On the other hand, any $\mathbf{x}$ satisfying two out of these three conditions can be tweaked to become a new keypoint. Another option is to move a keypoint from (x, y, σ) to $(x+u, y+v, \sigma)$, in order to change the local descriptor afterwards. This very preliminary work is not evaluating on the global system the impact of erasing, forging or moving keypoints, but it simply shows it is feasible.

Knowing the keypoint detection process, we are looking for the minimal local distortion to apply on an image for erasing or forging keypoints. This is done by applying a patch ϵ in the neighborhood of each keypoint (x, y, σ). Since the Difference of Gaussian kernel at scale σ has a limited support $\mathcal{S}_\sigma$ in the spatial domain, ϵ defined over $(x, y) + \mathcal{S}_\sigma$ modify the quantity $D(x, y, \sigma)$ of a given amount δ. In other word, for (u, v) in the neighborhood of (x, y), the image is modified in $I'(u, v) = I(u, v) + \epsilon(u, v)$ so that $D'(\mathbf{x}) = D(\mathbf{x}) + \delta$. The patch should be of minimal Euclidean norm to reduce the perceptual degradation. This obviously resorts to an optimization under constraint:

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

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

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

where $t_{(x,y)}$ is the 2D translation operator of a shift (x, y). This patch succeeds in controlling $D(\mathbf{x})$, however it also modifies the values in the neighborhood,

δ^+	total	removed	unchanged	new	PSNR
0.01	190	124	159	31	49.4
0.02	193	181	102	91	38.2
0.03	197	214	69	128	34.9
0.06	205	264	19	186	30.6

Table 1: Removal of keypoints, varying δ^+.

such as $D(x+u, y+v, \sigma)$ with $(u,v) \in \{-2^{(h-2)}, 0, 2^{(h-2)}\}^2$ where h is the octave of the scale. We can force the patch to not modify the $D(x+u, y+v, \sigma)$ values in the neighborhood previously defined. This add constraints requiring the patch to be orthogonal to $t_{(x+u,y+v)}(\Delta G_\sigma)$. Again, minimizing that convex function under affine constraints is solved by the Karush Kuhn Tucker theorem. Define $\mathcal{H} = \text{Span}\{t_{(u,v)}(\Delta G_\sigma) | (u,v) \in \{-2^{(h-2)}, 0, 2^{(h-2)}\}^2 - (0,0)\}$, then the optimal patch is:

$$\epsilon = \frac{\delta}{\|\Delta G_\sigma - p_{\mathcal{H}}(\Delta G_\sigma)\|^2} t_{(x,y)}(\Delta G_\sigma - p_{\mathcal{H}}(\Delta G_\sigma)), \tag{3}$$

$p_{\mathcal{H}}(\Delta G_\sigma)$ being the projection of ΔG_σ on $\mathcal{H}$. Note that the norm of the patch is $|\delta| / \|\Delta G_\sigma - p_{\mathcal{H}}(\Delta G_\sigma)\|$. Therefore, adding constraints increases the norm of the patch, producing more visible artefacts.

Let C be the fixed contrast threshold. In a first experiment, we erase keypoints in the subset $\mathcal{E}_{\delta^+} = \{\mathbf{x} : C < |D(\mathbf{x})| < C + \delta^+\}$, which means that we decrease $|D(\mathbf{x})|$ by an amount $|\delta|$ such that its new value is below the threshold C: $|\delta| = |D(\mathbf{x})| - C$. The threshold C sets the robustness against noise addition as only very salient points are selected. Note this helps the pirate because the higher C, the lower $|\delta|$ can be.

Erasing keypoints is performed on the image "Einstein" which has originally 283 keypoints. Here $C = 0.02$. Table 1 summarises the results when comparing keypoints between the attacked and the original images. It shows, for varying δ^+ values, the total number of detected keypoints in the patched image, the number of removed keypoints and the number of unchanged keypoints. As expected, when δ^+ grows, because we tackle a bigger subset $\mathcal{E}_{\delta^+}$, more keypoints are removed and less keypoints remain unchanged between the two pictures. The total number of keypoints, however, is not following the same trend because new keypoints are unintentionally created due to the distortion of the patch addition. At PSNR= 34.9 dB, the image is very degraded whereas 25% of the keypoints are still matching with the original. The most visible artefacts are caused by the removal of keypoints at higher scales, mostly because the patches have a big size and a strong amplitude.

In a second experiment, we test the complimentary strategy forging new keypoints, using the same original image. We address the local extrema in the subset $\mathcal{F}_{\delta^-} = \{\mathbf{x} : C - \delta^- < |D(\mathbf{x})| < C\}$. In this case, adding patches strengthen the contrast in the neighborhood of keypoints. This reduces the gap between the absolute values of the first and second eigenvalues, such that the third condition above (on the Hessian matrix) gets verified most of the times. Results are shown Table 2. It shows that most keypoints existing in the original image are found again in the attacked version as the unchanged number does

δ^-	total	unchanged	new	PSNR
0.003	387	280	107	57.8
0.005	467	279	188	49.0
0.007	549	278	271	45.7
0.01	698	282	416	41.6
0.02	945	272	673	36.5

Table 2: Forgery of keypoints, varying δ^-.

not vary significantly. The new keypoints are easy to add, especially in the first octave, such that the artefacts look like salt and pepper noise.

By combining the two strategies, we are able to seriously decrease the ratio of keypoints matching between the two images over their total number. However, there is always a small set of keypoints whose removal causes too severe distortions. Two major lessons can be drawn from this experiment. First, there seems to be a trade-off between robustness against classical attacks and security (quantified here by the visibility of the hacks). Second, and more importantly, the pirate cannot entirely delude a CBIRS with this unique strategy.

5.2 Challenging a Global Description

Here, we investigate one possible attack against a global description scheme. Many schemes rely on a quantization of the features for robust hash creation. Quantization not only provides a compact representation of the extracted features into a binary hash, but also sets the trade-off between robustness and diversity. A big quantization step improves the robustness because a feature of the distorted copy is still quantized in the same bin as its undistorted version, but it also lowers the entropy of the hash, and hence the likelihood of collisions. The key assumption we make in this security analysis is that the pirate knows about the location of the quantization cells.

It is clear that, for a given content, some features are more prone to be attacked than others: the pirate should start by moving the features lying near the quantization frontiers outside their cells in order to minimize the Euclidean distance between the pirated and the original images.

To illustrate an attack on an existing quantization process, we consider the well-known Scalable Color global Descriptor (SCD) from MPEG-7. The SCD creation process on one image starts with a uniform quantization of the HSV space, with 16 levels for H, 4 levels for S as well as for V: $\hat{h}(i,j) = Q_{16}(H(i,j))$, $\hat{s}(i,j) = Q_4(S(i,j))$, and $\hat{v}(i,j) = Q_4(V(i,j))$. This leads to a 256 bins histogram: $p(\hat{h},\hat{s},\hat{v})$. The histogram values are then non-uniformly quantized in a 4-bit representation to achieve more efficient encoding: $\hat{p}(\hat{h},\hat{s},\hat{v}) = Q_{16}(p(\hat{h},\hat{s},\hat{v}))$. This gives higher significance to the small values, while important ones may be truncated. Finally, the histogram is encoded using a Haar transform. SCDs are compared using the L_1 norm.

The attack principle is the following: the pirate is allowed to change a given percentage of pixels color values, in order to deeply modify the descriptor, while minimizing the visual distortion. Knowing the quantization step, we can determine, for each bin $(\hat{h},\hat{s},\hat{v})$, the minimum number of pixels $\delta_{\hat{h},\hat{s},\hat{v}}$ to move in

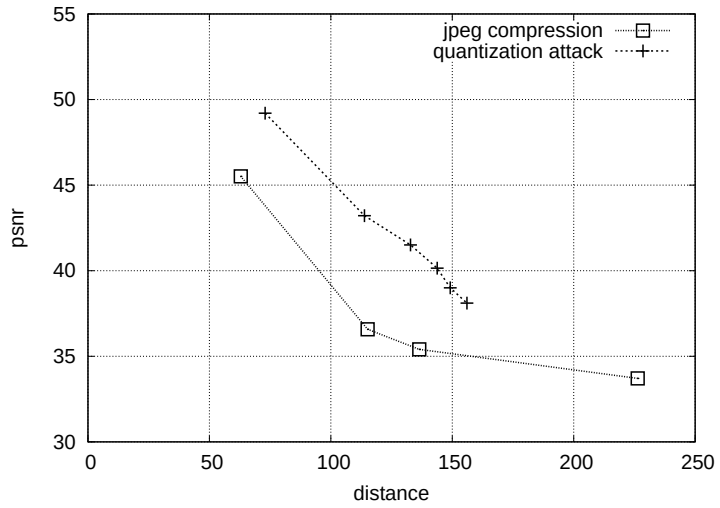

Figure 1: Attacking SCD quantization process.

order to change the quantized value $\hat{p}(\hat{h},\hat{s},\hat{v})$. Each bin $(\hat{h},\hat{s},\hat{v})$ is ranked in increasing $\delta_{\hat{h},\hat{s},\hat{v}}$ order and iteratively proceed until the allowed amount of changed pixel is reached. The severity of the attack depends on this last parameter.

In a bin $(\hat{h},\hat{s},\hat{v})$, the $\delta_{\hat{h},\hat{s},\hat{v}}$ pixels to modify are chosen among those which $H(i,j), S(i,j), V(i,j)$ values are the nearest from quantization frontiers. To ensure coherence, a pixel (i,j) can not be modified if (i) it has already been modified in a previous iteration, (ii) its new quantized value $(\hat{h}',\hat{s}',\hat{v}')$ corresponds to a bin already processed.

We estimate the attack efficiency through the distance between the SCD of the original and attacked images. The goal here is to produce an image with that distance greater than any other distance between the original image and quasi-copies forged using a standard typical transformation. In this example, we compare our "informed" attack to what gives jpeg compressions, as SCD are sensitive to this type of modifications. The pnsr value is used to measure the introduced distortion for all modified images.

We apply that quantization hack, as well as jpeg compression, to a set of 100 images randomly taken from Flickr, so that they present a great content diversity. We vary the attacks severity through, respectively, the percentage of modified pixels, and the quality factor of jpeg compression. For each attacked image, we compute its distance and psnr to each corresponding original image. Figure 1 compares the average distance versus the average pnsr computed over the image set for different jpeg compression and quantization hacks. The quality factor of jpeg compression is set to 20, 40, 60 and 80. In the case of quantization hack, the percentage of modified pixels takes value among 1%, 5%, 10%, 15%, 20%, and 25%. As expected, the distance increases and the psnr decreases as the attack severity is more important. Figure 1 shows that for a given distance, the psnr of attacked image is greater than for the jpeg-modified. This means the dedicated modification less degrades the image than a jpeg-compression for a equivalent change in descriptor. Conversely, for a given psnr, the average distance between the original and attacked images is larger than to the jpeg-

Figure 2: The original and the modified images.

modified images. This means the attacked images are further away in the feature space, offering opportunities for reducing the recognition quality of the system.

5.3 Challenging Retrieval

To demonstrate the feasibility of attacking the "database" part of CBIRS, we show it is possible to conceal quasi-copies by exploiting the knowledge a pirate can have of the internals of the retrieval process. This toy example follows the guidelines presented in the "Stop Rule" part of the Section 4.4. Therefore, the pirate knows that (i) at search time, query descriptors are processed sequentially, as they are stored in the query file given to the system and (ii) a stop rule mechanism exist. This example uses a real CBIRS [12] indexing 1,127,918 high-resolution images randomly downloaded from Flickr. Each image is described using the local descriptors by Lowe [14], resulting in a total of 1,222,920,344 SIFT descriptors. That system was chosen as it handles gracefully pretty large descriptor collections.

On top of this system, we simulated the stop rule by evaluating during the search, and after having processed a varying percentage of the query descriptors (20%, 30%, 40% and 50% here), how confident can we be that a match has been found, that no match is likely to be found even with further processing, or that more query descriptors need to be processed to eventually push the decision one way or the other. That confidence is one minus the probability for the image from the database with the current highest score to have that score by chance. Evaluating that probability gives a value to a decision flag, respectively "found", "failed" or "cont.", impacting the continuation of the search process.

To challenge the system, the pirate takes one particular image that is in the database (we therefore know there must be a match) and produces a quasi-copy by inserting a small box of text at the top. The original and the modified image are shown Figure 2. Analysing the SIFT of both images shows that the original image has 511 descriptors, the modified has 640 descriptors, and that all the 129 new descriptors in the modified image are around that text box. The pirate then sorts the SIFT descriptors in the attacked query file according to the increasing subpixel row location. Therefore, all the 129 new SIFT appear at the beginning of the modified image description file. Note the original image has no SIFT descriptor in the area covered by the text box.

The pirate then submits the attacked query descriptor file to the system. For fairness, we also query the database with the original unmodified query file

% of query	Original (511)			Modified (640)		
	#D	Score	Flag	#D	Score	Flag
20%	102	12	cont.	128	¬ 4	failed
30%	153	18	found	192	10	cont.
40%				256	13	cont.
50%				320	20	found

Table 3: Traces of executions with Stop-Rules.

sorted on row location. Table 3 details the results of the searches. This table is divided in two parts, the left part showing information when searching the original, unmodified image. The right part shows information when searching the attacked image.

Each line shows the number of descriptors used by the search process (#D) at each percentage. It also shows the score of the best match found so far and the value of the stop-rule flag. The 20% line of Table 3 says, for the original image, that after 102 query descriptors, the best score is 12, and this is not enough to decide to stop. Therefore, the search carrieson, and after having processed 30% of the query, the flag is set to "found", the search stopped and the recognition successful (saving 358 query descriptors).[1]

In contrast, for the modified image, that 20% line says that after 128 descriptors, the best score is 4, the corresponding image name is the wrong one (this is indicated by the ¬ sign; execution traces show the second best score is also 4), and that this score is so small with respect to #D that the stop-rule decides to abort the search, failing to return any result. This was to be expected as only the descriptors associated to the text box have been considered so far in the attacked image and it turns out that they match with nothing from the database. Therefore, a system with stop-rules can be deluded because of the order according to which query descriptors are processed. Note that once more than 129 descriptors are considered for the modified image (as if instead of first checking the stop-rule at 20%, the first check was at 30%), then normal matches are starting to be found. A simple counter measure would be to randomize the descriptors from query files.

5.4 Discussion

The three experiments presented in this section must be understood as proof of concepts. Our goal was to show that it is possible to produce attacks using the knowledge of the internals of the typical non-secret technologies in CBIRS, despite their robustness. We are well aware that the attacks we came up with are simple as this is very initial work. We are convinced that devising much stronger attacks, against more complex technology blocks, with more convincing proofs of deludings, with sophisticated strategies is just a matter of time. Our plans push this study further in that direction.

[1] We also queried the index with 49 standard modifications obtained by applying Stirmark on that original image (crops, rotations, rescalings, filtering, ...). The original image was always found, showing how robust the descriptions are.

The experiments, however, suggest that it is unlikely any pirate will break an entire system by attacking only one of its components. Instead, taking advantage of many little security flaws is probably the way to go. Deluding a real system certainly asks to produce attacks against each of the various stages of image description, then carrying on with attacking the different steps for retrieval, etc, possibly also attacking much earlyer, at database creation time. The attacks against each part can also be designed in a joint manner instead of independently, to increase the overall impact.

It is likely future CBIRS will embed mechanisms enforcing their security against users, as they tend to be considered as potential pirates, by default. But honnest users may also want to be sure that the system they are using is not dishonnest: How can right-holders assess that a CBIRS filtering pirated contents on a UGC plays its role fairly, and does not allow the upload of illegal material belonging to particular owners or with particular contents? What if the pirate is inside the system?

6 Conclusion

In this paper, we have analyzed content-based image retrieval systems from a *security* point of view. The starting point of this analysis is the intuition that a pirate can severely delude a system by exploiting the knowledge he can have on the technology blocks used in that system. We described the key security concepts applied to the case of CBIRS and the resulting threat analysis framework. We then detailed a first set of security flaws found in some of the main state-of-the-art blocks of existing systems. We then performed three experiments showing it is possible to delude these techniques, suggesting stronger and worst attacks are likely to happen against systems protecting valuable contents in the real world.

Challenging the security of CBIRS asks to study and cover the security flaws of systems, going through trust models, goals, knowledge leakages and potential oracles, and the secret versus non-secret design principles. But also, it asks to invent protocols to verify that "informed" attacks are absorbed, as well as inventing protocols verifying that systems are not themselves dishonest, hijacking the data from users to do something they are not supposed to do.

References

[1] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006.

[2] F. Cayre, C. Fontaine, and T. Furon. Watermarking security: Theory and practice. *IEEE Trans. Signal Processing*, 53(10), 2005.

[3] M. E. Choubassi and P. Moulin. On the fundamental tradeoff between watermark detection performance and robustness against sensitivity analysis attacks. In *Security, steganography, and watermarking of multimedia content VIII*, volume 6072 of *Proc. SPIE-IS&T Electronic Imaging*, 2006.

[4] P. Comesaña, L. Pérez-Freire, and F. Pérez-González. Blind newton sensitivity attack. *IEEE Proc. on Information Security*, 153(3), 2006.

[5] M. Datar, P. Indyk, N. Immorlica, and V. Mirrokni. *Locality-sensitive hashing using stable distributions.* MIT Press, 2006.

[6] J. Earl. Tangential sensitivity analysis of watermarks using prior information. In *Security, steganography and watermarking of multimedia contents IX*, volume 6505 of *Proc. SPIE-IS&T Electronic Imaging*, 2007.

[7] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. E. Abbadi. Approximate nearest neighbor searching in multimedia databases. In *Proc. ICDE*, 2001.

[8] M. Johnson and K. Ramchandran. Dither-based secure image hashing using distributed coding. In *Proc. IEEE Int. Conf. on Image Processing*, 2003.

[9] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, 9, 1883.

[10] S. Kozat, R. Venkatesan, and K. Mihcak. Robust perceptual image hashing via matrix invariants. In *Proc. of IEEE Int. Conf. on Image Processing*, 2004.

[11] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *Proc. CIVR*, 2007.

[12] H. Lejsek, F. H. Ásmundsson, B. Þ. Jónsson, and L. Amsaleg. Scalability of local image descriptors: A comparative study. In *ACM Multimedia Conf.*, 2006.

[13] C. Li, E. Chang, H. Garcia-Molina, and G. Wiederhold. Clindex: Clustering for approximate similarity search in high-dimensional spaces. *IEEE TKDE*, 14(4), 2002.

[14] D. Lowe. Distinctive image features from scale invariant keypoints. *IJCV*, 60(2), 2004.

[15] M. Malkin and R. Venkatesan. The randlet transform: application to universal perceptual hashing and image identification. In *Proc. of the Allerton Conf.*, 2004.

[16] Y. Mao and M. Wu. Unicity distance of robust image hashing. *IEEE Trans. Information Forensics and Security*, 2(3), 2007.

[17] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conf.*, volume 1, 2002.

[18] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10), 2005.

[19] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 65(1-2), 2005.

[20] S. Poullot, M. Crucianu, and O. Buisson. Scalable mining of large video databases using copy detection. In *ACM Multimedia Conf.*, 2008.

[21] H. Samet. *Foundations of Multidimensional and Metric Data Structures.* Morgan Kaufmann, 2006.

[22] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE Intl. Conf. on Computer Vision*, volume 2, 2003.

[23] A. Swaminathan, Y. Mao, and M. Wu. Robust and secure image hashing. *IEEE Trans. Information Forensics and Security*, 1(2), 2006.

[24] J. Vila-Forcen, S. Voloshynovskiy, O. Koval, F. Perez-Gonzalez, and T. Pun. Worst case additive attack against quantization-based data-hiding methods. In *Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681 of *Proc. SPIE-IS&T Electronic Imaging*, 2005.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399