Security-Oriented Picture-In-Picture Visual Modifications

Thanh-Toan Do, Laurent Amsaleg, Ewa Kijak, Teddy Furon

Security-Oriented Picture-In-Picture Visual Modifications

Thanh-Toan Do
Université de Rennes 1
Campus de Beaulieu, Rennes, France
thanh-toan.do@irisa.fr

Laurent Amsaleg
CNRS, IRISA
Campus de Beaulieu, Rennes, France
laurent.amsaleg@irisa.fr

Ewa Kijak
Université de Rennes 1
Campus de Beaulieu, Rennes, France
ewa.kijak@irisa.fr

Teddy Furon
INRIA Rennes Bretagne–Atlantique
Campus de Beaulieu, Rennes, France
teddy.furon@inria.fr

ABSTRACT

The performance of Content-Based Image Retrieval Systems (CBIRS) is typically evaluated via benchmarking their capacity to match images despite various generic distortions such as crops, rescalings or Picture in Picture (PiP) attacks, which are the most challenging. Distortions are made in a very generic manner, by applying a set of transformations that are completely independent from the systems later performing recognition tasks. Recently, studies have shown that exploiting the finest details of the various techniques used in a CBIRS offers the opportunity to create distortions that dramatically reduce the recognition performance. Such a *security perspective* is taken in this paper. Instead of creating generic PiP distortions, it proposes a creation scheme able to delude the recognition capabilities of a CBIRS that is representative of state of the art techniques as it relies on SIFT, high-dimensional k-nearest neighbors searches and geometrical robustification steps. Experiments using 100,000 real-world images confirm the effectiveness of these security-oriented PiP visual modifications.

Categories and Subject Descriptors

H.3.1 [**Information Systems**]: Information Storage and Retrieval—*Content Analysis and Indexing*; I.4 [**Image Processing and Computer Vision**]: General

General Terms

Security, Content-Based Image Retrieval

Keywords

SIFT, Forensics, Picture in Picture, Image Database, Visual Modifications

ICMR '12, June 5-8, Hong Kong, China

1. INTRODUCTION

Content-Based Image Retrieval Systems (CBIRS) get increasingly involved in various multimedia forensics applications. They mainly perform automatic detection of sensitive contents like illegal copies of copyrighted material, child pornography images or terrorist images and videos propaganda. Their central role can be explained by the recent progress made by the technology they use. CBIRS are very efficient, allowing extremely fast recognition even when managing collections containing several million images [13]. They have very good recognition capabilities as they identify images that are similar to a query despite rather severe visual modifications such as crops, rotations, scale changes, compressions, ...

Recognition wise, *Picture in Picture* are among the most difficult visual modifications to cope with. A Picture in Picture (PiP) visual modification tries to preclude the recognition of a copyrighted picture by inserting a downscaled version of that picture inside another innocuous and distracting large scale picture.[1] The identification of the copyrighted picture tends to fail because the recognition of the distracting image dominates. According to the specifications of the TRECVID competition as well as the results obtained by many research teams worldwide [20], coping with PiP is probably the hardest task when doing content-based copy detection. For a fair evaluation, TRECVID produces a set of PiP modifications challenging the competitors' CBIRS. In general, PiP modifications (by TRECVID or anyone else) are created in a quite generic way, independently of the systems that will later battle them. This image modification process typically belongs to a set of generic content transformations used to benchmark recognition robustness.

Assessing the performance of systems from a robustness point of view is mainstream in the Computer Vision or Multimedia literature. None of these communities explored the performance of systems from a *security perspective*, however. The security of a CBIRS is its ability to resist to some dedicated attacks led by malicious pirates against the specific techniques this system uses. Recently, a handful of papers have warned the community about the poor security levels of CBIRS [11, 5, 7]. These papers describe various strategies endangering the recognition capabilities of systems re-

[1]Inserting a copyrighted video inside a distracting video raises very similar issues. The PiP term we use here refers to both problems.

lying on SIFT [18] for recognition. These strategies produce very specific visual modifications that are SIFT aware, i.e., which fully benefit from a deep knowledge of SIFT internals to minimize visual distortion while dramatically reducing descriptor matching.

This paper adopts a security point of view on the Picture in Picture visual modifications. Instead of producing PiP images in a generic way, we propose a technique producing PiP attacks able to delude the recognition capabilities of a system using SIFT, running high-dimensional k-nn (nearest neighbors) searches and final geometrical verification steps. Note that SIFT, k-nn and geometry are state-of-the-art building blocks of many real-life copyright oriented CBIRS. In a nutshell, existing PiP attacks embed the severely scaled-down version of the protected image in a distractor, which dominates recognition. Our attacking scheme does the opposite: it embeds a very small distractor image patch inside the (full-resolution) protected picture. That small patch is carefully determined such that it will catch almost all matches avoiding the full resolution image to be identified, hence deluding recognition.

This paper is organized as follows. Section 2 briefly describes how PiP visual modification schemes are created and how existing CBIRS recognize contents despite PiP. Section 3 describes how are produced these security-oriented PiP visual modifications. Section 4 shows the effectiveness of the proposed scheme through large scale image recognition experiments. Finally, Section 5 concludes.

2. PIP VISUAL MODIFICATIONS

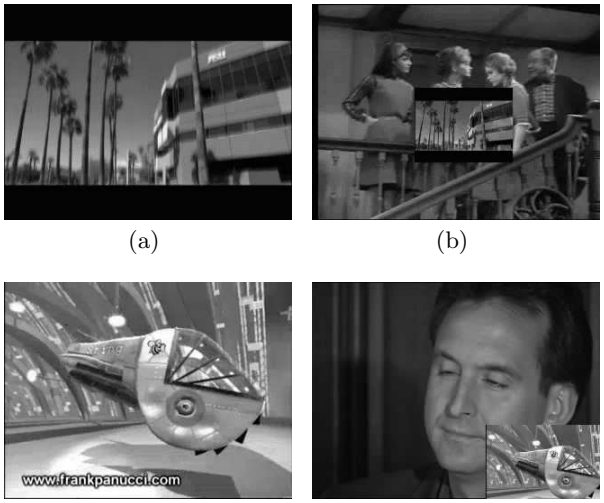
This section gives a brief overview of the typical process that eventually creates a Picture in Picture visual modification. It then discusses about the two families of solutions found in the literature recognizing contents despite PiP attacks.

The overall context assumes that a large database of contents to protect has been created somehow, offline. Query images later submitted are checked against this database and the system decides whether queries are or are not quasi-copies of the protected contents. The system succeeds when a quasi-copy that is indeed in the query is detected, despite the various visual modifications it has undergone (the query is a *true positive*). The system also succeeds when it asserts that the query is a *true negative*. In this case the system has not found any piece of content in the database similar to the query because this query is indeed purely made of innocuous material.

In contrast, the system fails when it can not detect that a protected piece of content is in the query *(false negatives)* or when it asserts that the query contains some protected contents while it is indeed not the case *(false positive)*. In general, the goal of the visual modifications are to produce as much as possible (i) false negatives precluding the detection of protected contents and/or (ii) false positives making the whole system unreliable in practice.

2.1 Producing PiP Visual Modifications

The way TRECVID produces PiP modifications is quite generic and works as follows [20]. First, an image (or video) is randomly picked from a collection of distracting contents, in practice unlikely to match with the database of protected material. Then, the image (or the video) that the modification process tries to hide from recognition is scaled down and is inserted in the distracting contents. TRECVID specifies the rules driving the downscaling as well as the location of the inserted contents. The downscaled size ranges between half and one third of its original size; the downscaled protected content can be inserted at five locations, the four corners or the center of the distracting image. Furthermore, the aspect ratio of the inserted image/video is typically fixed. TRECVID benchmark includes several hundreds of such PiP material. Figure 1 shows four examples from TRECVID. It is interesting to compute the PSNR between the original and modified versions. PSNR is naturally very small as pixels are very different. In this example, PSNR(1(a),1(b))=12.64 and PSNR(1(c),1(d))=8.35.



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

Figure 1: Four examples from TRECVID: (a) and (c) are the images to be protected, kept inside the database; (b) and (d) are the corresponding TRECVID PiP visual modifications.

2.2 Recognizing Contents Despite PiP

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

2.2.1 Prior Knowledge for Separating Images

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

Some image separation techniques use edge/line information to detect the boundaries of the inserted image. The detection typically relies on a Canny edge detector or on a Hough transform [19, 17, 22]. In [17], a Canny edge detector is run to locate horizontal and vertical edges which can possibly be at the boundaries of the inserted image. Candidate regions which will later be used as queries are obtained by

grouping four edges in pairs made of two horizontal and two vertical edges. Some candidate regions are then eliminated according to some additional criteria such as their aspect ratio as well as their location. Finally, the remaining candidate regions are extracted from the global image, up-scaled and probe the system. The one with the highest similarity score is returned as the final answer. [19] uses a Hough transform to detect persistent strong horizontal lines from which the candidate regions are determined.

Separating the distracting from the potentially protected contents is somehow similar when dealing with videos [22]. In this case, Canny is run on every single frame of the video, which in turns creates a series of detected edge images. A time-sliding window processes consecutive edge images and produces average edge images. A Hough transform is then applied on these average edge images to detect persistent lines, and short lines as well as non horizontal or vertical lines are eliminated. The remaining lines create candidate regions, probing the system one after the other.

Some other methods detect corners of regions instead of lines. In [3, 2], a Sobel operator and a Laplacian kernel are applied to video frames. Pixels in frames that are found to be part of either horizontal or vertical edges are flagged. The corners of candidate regions are thus the pixels in images where the maximum accumulation of edge points on horizontal and vertical directions are found.

Other methods make an even more rough use of the prior knowledge of the rules driving the creation of PiP visual modifications. For example, [15] extracts from the whole image a series of candidate regions located at the four corners as well as around the center, having sizes of 30%, 40% and 50% of the whole image. These regions are then used to probe the database.

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

2.2.2 *Identification Without Prior Knowledge*

In contrast to the approaches mentioned above, some other techniques do not separate the images. They rather compute feature vectors over the whole image and then query the database with every single descriptor, without separating anything [12, 9, 24, 25]. Once all the query descriptors have probed the database, the list of candidate images is then post-processed in order to identify the potential protected contents. This post-processing step typically checks the geometry consistency between the query and all images in the candidate list. Very robust tools estimating the geometrical consistency are used, such as RANSAC [8, 16, 21] or the Generalized Hough Transform [1, 18].

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

These techniques strongly assume that the distracting part of the image/video query does not belong to database—this is a very serious drawback. Otherwise, when the distracting part is in (or turns out to be in) the database, then the result returned by the CBIRS is much more ambiguous. When the distracting part dominates recognition then the inserted part, also protected, is not identified. This in turn might cause severe difficulties in applications where copyright holders get remunerated on a pay-per-view type-of basis. This is typically a security threat as a pirate well aware of the details of the application as well as of the monetizing rules can bias the system and cause a fraudulent behavior. Like for the techniques based on a prior knowledge, it is not a problem of robustness, but a problem of security.

3. CBIRS-AWARE PIP CREATION

In contrast to the generic way of creating PiP visual modifications, we now detail a technique producing PiP attacks deluding the recognition capabilities of a CBIRS based on the SIFT description, an approximate high-dimensional k-nn search and a final geometrical verification. These three techniques managing image description, indexing and retrieval as well as removing false positives are representative of how most state-of-the-art systems work. They are used in real-life applications because they cope with large data collections, their recognition power is excellent, their speed is impressive and the results they return is high quality despite approximations. We first present the salient aspects of these techniques before detailing the CBIRS-aware PiP creation process.

3.1 Representative CBIRS Techniques

Attacking a system from the security side requires to deeply know its internals. Like some prior works [11, 6, 4], this paper aims at modifying the content of images to skew their description such that they are hardly detected by the system. It is therefore key to detail the SIFT image description [18].

SIFT computes local features in three steps. First, it detects a keypoint located in (x, y) in the image at scale σ if it is a local extremum of the DoG (Difference of Gaussian) response. In the second step, the main orientation θ is computed based on gradient directions locally around (x, y). The keypoint is defined as $kp = \{x, y, \sigma, \theta\}$. The third step computes the descriptor on a support region centered on (x, y) and whose size depends on scale σ. The support region is divided into 16 subregions, and the 8-bin quantized histograms of weighted gradient orientation of the subregions are concatenated into a 128-dimensional vector.

SIFT feature vectors are extracted from all images of the database and then inserted into an high-dimensional index, offline. Due to the curse of dimensionality, indexing is approximate to achieve good performance by trading-off result quality for response time. When a user submits a query to the system, its features vectors are first extracted and an approximate k-nn search is run for each query vector. Each database vector found during this process then votes for the

Figure 2: Example of an image after the full washing process. PSNR=30.03

Washing Step	# keypoints	# matches
Original image	1034	1034
Reducing Density	910	833
Changing Orientations	961	505
Local Smoothing	694	115

Table 1: Number of keypoints and number of matches between the washed images and their original counterparts. Average on 1,000 images.

database image it has been extracted from. This eventually gives a list of candidate similar images along with scores. A final step checks the geometric consistency of these candidate images with the query. This re-ranks the candidate images. The scores of the images in this list as well as their degree of geometrical consistency allows the CBIRS to decide whether or not the query contains protected content.

Overall, it can be seen that the ability of the CBIRS to correctly match descriptors has a direct impact on the final decision. The more different the descriptors are between images, the less likely they will be found similar by the system. The goal of the security-oriented PiP visual modification is therefore to eventually produce descriptors extracted from the quasi-copy that are as much as possible different from the one extracted from the image to protect while minimizing the visual distortion. Following the guidelines of [6, 4], two approaches can achieve this goal. The first one tries to remove as much keypoints as possible from the image to hide—fewer keypoints make matching less marked; the second one does not touch keypoints but touches their support regions to end up with quite different high-dimensional SIFT descriptors. Both techniques are summarized below.

3.2 Image Washing Before PiP

As proposed in [6, 4], a forged copy of a protected image is 'washed' to reduce as much as possible the number of matches between that forged copy and its original version. First, the regions of keypoints density in the quasi-copy (called image I) are identified using a window of size (50×50) sliding over I. When more than 60 SIFT-keypoints exist in that window, then the whole region in the image is smoothed by Gaussian function with σ equals *1.3*.

The washing carries on by manipulating the support regions of the remaining keypoints. [4] modifies the value of the pixels in some of these regions in order to change their dominant orientation in a non affine-way, since SIFT absorbs affine transforms. For every keypoint, it determines how the pixels in the support region should be changed for shifting its orientation by at least $\pi/6$. This, in turn, ensures that the descriptors computed from this tweaked support region and from the original support region are far enough in the high-dimensional space. The changes are applied in the picture if the PSNR computed between the original and tweaked support regions does not drop below a threshold.

Finally, a small, very local, smoothing is done where there remain some unmodified keypoints, hoping each is afterwards not anymore a local extremum of the DoG. Again, here, PSNR is checked to avoid too severe visual distortions.

Overall, after washing, the forged copy of a protected image has (i) few descriptors that perfectly match with its original version, (ii) many descriptors that are unlikely to match since they have been moved away in the high-dimensional space and (iii) several new keypoints created as side effects due to visual artifacts that may or may not match [7]. Querying the CBIRS with such a washed image is likely to identify the original but with a very low score, making the system less confident. Figure 2 and Table 1 illustrate this process. Table 1 gives the average number of keypoints that are detected in the various washed versions of 1,000 images. It also gives the average number of matches between a washed image and its original counterpart determined using Lowe's criterion [18].

3.3 Creating the PiP Attack

Washing even more reduces the number of keypoints that still match with their original counterpart and breaks recognition. Yet, so severely washed images are much too distorted to remain usable in practice and keep any commercial value. In contrast, as highlighted in the Section 2, making sure the distracting part dominates the recognition is one option; making it difficult to separate the two images is another. For these reasons, our scheme for producing CBIRS-aware PiP attacks adopts the following guidelines:

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

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

3.3.1 Determining Candidate Visual Patches

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

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

Analyzing these disclosed images bootstraps the creation of CBIRS-aware PiP attacks. First, these images are segmented according to the keypoint density and only dense enough regions are kept. Then, each dense region is repeatedly cropped around its center to create small square visual patches of 10×10 pixels, 15×15, 20×20, ... This, somehow, creates a dictionary of visual patches that are sorted by the number of keypoints still matching their origin image after cropping.[2] Then, we select a patch from this dictionary containing more keypoints than the number of remaining matching keypoints after the washing of the forgery. Inserting this patch in the washed image is going to produce enough geometrically consistent matches to push the original image down in the result list (at least with rank #2, see 3.3.4).

In some applications, it might be impossible to disclose what is inside the database. It is however possible to create a set of visual patches that are extremely likely to match with some images from this unknown database. Many images include textures that are very repetitive, such as tiles, bricks, tree leaves, window shutters, windows on a facade seen from far enough, friezes, fabrics, clothes, etc. Downloading a fair number of pictures from Flickr provides an easy way to bootstrap the construction of a dictionary of keypoint-dense visual patches, as described above. These patches typically contain regular and repetitive patterns. The pirate bets that these patterns will match with some of the database images even if no images from the database are known. Furthermore, it is likely these patches will be geometrically consistent with some of the database contents.

Overall, our strategy starts by building a dictionary of visual patches. Examples are given in the Table 2. Patches have different sizes since their keypoint density varies. It then picks from the dictionary the patches that have more keypoints than what remains in the washed image. This gives a list of candidate patches. The next sections describe the selection of one or more patches in that list and their insertion in the washed image.

3.3.2 Inserting a Visual Patch

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

[2]For example, a 50×50 pixels cropped patch containing 400 keypoints has only 300 matches with its origin image. The difference comes from the keypoints near the borders of the patch which can not match anymore.

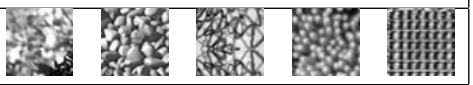
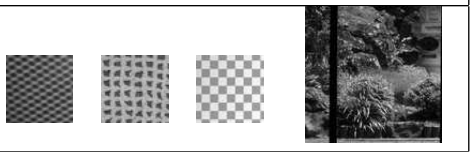
# of keypoints	Visual Examples
50–100	
100–200	
200–300	
300–400	
400–500	

Table 2: Dictionary of Keypoint-Dense Visual Patches.

Policy #1: PSNR-Oriented.

The first policy finds the place in the washed image where the patch can be inserted while reducing the PSNR as little as possible. Let I_w be the washed image into which the patch I_p has to be inserted, eventually producing the image I'_w. I_p is a $p \times p$ pixel square. First, I_p is turned into a column vector, centered $\tilde{I_p} = I_p - \bar{I_p}$, with $\bar{I_p}$ denoting the average luminance. $\tilde{I_p}$ is then sled over the washed image. Because SIFT is invariant to illumination changes (provided the local contrast is not too small), it is possible to adjust the illumination of $\tilde{I_p}$ to be as close as possible to $I_w(x, y, p)$ the $p \times p$ region of I_w around position (x, y). This boils down to finding parameter a and b such that:

$$\min_{a,b} \|a\tilde{I_p} + b - I_w(x, y, p)\|^2, \text{ subject to: } a \geq a_{min}. \quad (1)$$

The constraint $a \geq a_{min}$ avoids small values of a which flattens $\tilde{I_p}$ removing most (if not all) of its keypoints. Solving (1) gives a and b:

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

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

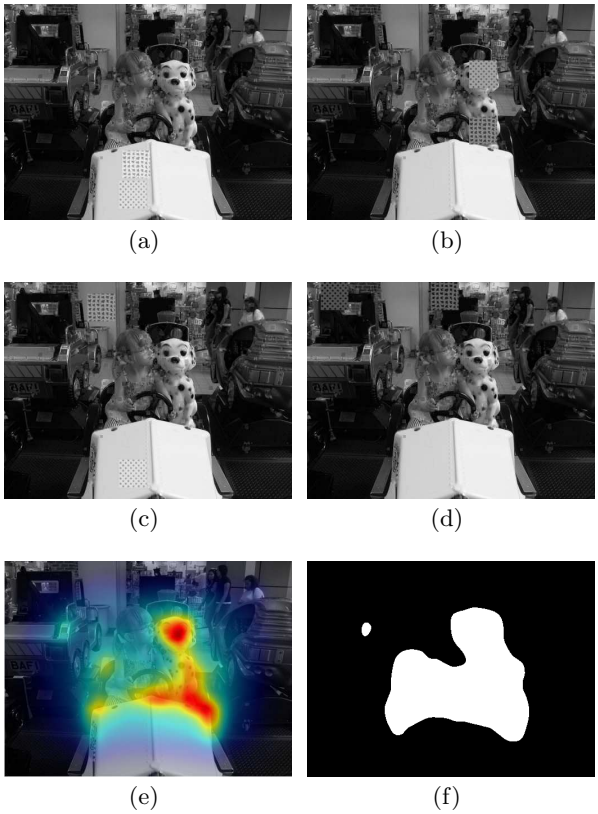


Figure 3: Illustrating the four patch insertion policies. (a), (b), (c) and (d) are Policies #1 to #4, respectively. PSNR are respectively 29.13, 24.77, 29.11, 25.14. (e) shows the map of salient regions found using GBVS. (f) gives in white the area where high-saliency forbids patch insertion

ate values for a and b, the washed image becomes I'_w. This policy tends to insert I_p in poorly textured regions.

Policy#2: Matching Density-Oriented.

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

Policy#3: Visual Attention-Oriented & PSNR.

Care must be taken to move away the patch from being inserted in the middle of, or close to, the visual centers of interest that exist in the image. This third policy therefore computes a visual saliency map for I_w using the Graph-Based Visual Saliency (GBVS) approach [10]. Policy#3 then simply determines salient-enough regions and then applies Policy#1, forbidding the sliding window to enter inside these regions.

Policy#4: Visual Attention & Matching Density.

Once the salient-enough regions have been determined, then Policy#2 is applied. This inserts the patch where the largest number of matching keypoints is found, outside salient regions.

3.3.3 Blurring Boundaries

Systems separating pictures to cope with PiP attacks heavily rely on horizontal and vertical segment detection. Challenging their security is possible by modifying the images such that detecting such segments becomes very difficult. One option is to blur the boundaries separating the two images. After inserting the patch, a small and local Gaussian ($\sigma = 1$) blurring around the frontier does the job. Then, vertical and horizontal edges are much harder to detect.

3.3.4 Inserting Multiple Visual Patches

Inserting one patch in the washed image pushes the corresponding original image at least at rank #2 in the result list. It might be desirable to push it further. In this case, several patches might be inserted in the washed image. Two inserted patches push the original image to rank#3, etc. Of course, it is not possible to push it extremely far without too severely degrading the washed image. Figure 3 shows examples where two patches have been inserted.

4. LARGE-SCALE EXPERIMENTS

In order to validate our CBIRS-aware PiP attacking schemes, we ran experiments with real images and with a real fully functioning CBIRS. We first present the evaluation setup before discussing the lessons learned from the experiments.

4.1 Dataset, Queries and Setup

We created an image collection composed of 100,000 random pictures downloaded from Flickr that are very diverse in contents. All images have been resized to 512 pixels on their longer edge. We then randomly picked 1,000 of these images to use them as query images. We also used a large set of random images to build the dictionary of visual patches.

We computed the local description of these images using the open-source SIFT-VLFeat code by Vedaldi [23]. The collection yields 103,454,566 SIFT-VLFeat descriptors. These descriptors are indexed by the NV-Tree high-dimensional indexing scheme [14] which runs approximate k-nearest neighbor queries. The NV-Tree returns a candidate list of the best 100 matching images.

That list is then processed in order to rerank candidates based on their degree of geometrical consistency with the query image. This process iteratively takes one candidate image, determines which of its descriptors are matching with the query using Lowe's criterion [18]. This checks whether the Euclidean distance ratio between matching descriptors and their nearest neighbor is below 0.8. The matching descriptors are the inputs of a Hough transform model estimation in order to measure the geometric consistency of the matches. The mean square error is used to keep the best model. The number of inliers for this model is used to rerank the 100 candidate images.

4.2 Experiment 1: Inserting One Patch

In this first experiment, we first wash the 1,000 query images and then determine which patch should be inserted in each washed image. We then insert patches in images

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

Table 3: Counting observed ranks of identified images, 1 patch, varying policies.

according to the four patch-insertion policies defined above, eventually ending-up with 4,000 CBIRS-aware PiP attacked images. Each attacked image is used to query the database. We first discuss the effectiveness of the attacks before giving details on the patches used to produce these attacks.

4.2.1 *Effectiveness of the Attacks*

Table 3 shows the effectiveness results of this experiment for each of the four policies as given by the first column. The second column gives the average PSNR between the original images and their attacked version, for the four policies.

The third column gives the number of times the PiP attack was not successful since the washed and attacked images could still be identified by the system and be ranked first. It turns out that some images can not be attacked in practice while preserving their PSNR. They are too much textured, still have too many unchanged keypoints after washing and would require the insertion of huge visual patches to topple over recognition. Note the failure rate is very small (1.88%, roughly, across policies).

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

The fifth column gives the number of times the original image is not found in the 100 most similar images. This happens when the washing removes enough keypoints and/or triggers matches with other random images from the database in addition to what produces the patches.[3]

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

4.2.2 *Details on Patches*

In addition to keeping track of image similarities for determining the effectiveness of the attacks, we recorded detailed information on the patches used to create the PiP visual modifications. Figure 4 shows the number of times patches having the sizes given by the x-axis were used to produce the 1,000 attacks in the case of Policy#1 (PSNR-oriented), quantized every 5 pixels. Figure 5 shows the impact on the PSNR of using patches of varying sizes. Overall, this shows that most patches are smaller than 50×50 pixels, which is to contrast with images that are typically 512×384. Of course this in turn reduces the PSNR. It is interesting to observe that in some cases very small patches are picked from the dictionary: their keypoint density is very high and they tend to contain small repetitive visual patterns such as clothes or bricks. We conducted that same study with the three other policies without seeing any significant changes.

[3]Note that on average 4.75 original images were ranked between 3 and 100.

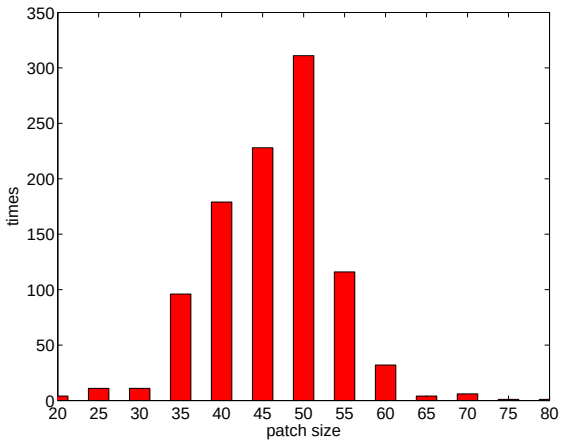


Figure 4: Patch Sizes Used

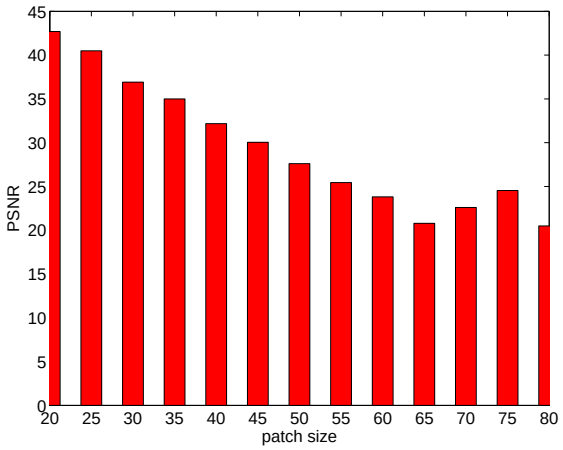


Figure 5: Patch Impact on PSNR

4.3 Experiment 2: Inserting Two Patches

This experiment demonstrates that it is possible to increase the rank of the original image by inserting more than one patch. This allows to make even more difficult the identification of the protected picture as (potentially) many other unrelated pictures dominate recognition. Table 4 gives effectiveness results when inserting two patches in the images, varying the policies. This table reads as the one given above. Here, the original image is (almost) never ranked first (col 3), occasionally ranked #2 (col 4), very often ranked #3 as desired (col 5). In addition, it is quite often not even found among the 100 most similar images (col 6). Note that on average 2.25 original images were ranked between 4 and 100.

5. CONCLUSION

This paper is showing that creating Picture in Picture visual modifications from a *security perspective* is seriously

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

Table 4: Counting observed ranks of identified images, 2 patches, varying policies.

challenging the recognition power of CBIRS using techniques that are representative of the state-of-the-art. This is demonstrated by the evaluations made with a large database of 100,000 real images together with a real system using SIFT, k-nn and checking the geometrical consistency of matches. While the PiP creation scheme and the various policies detailed here are able to delude the system, substantial additional work is needed to reduce the visual impact when washing images as well as when inserting carefully chosen visual patches. It is likely inpainting methods will help preserving the visual quality of the attacked images. It is part of our plans to go one step beyond what is presented in this paper by integrating our security-oriented CBIRS attacks with inpainting strategies.

6. REFERENCES

[1] D. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2), 1981.

[2] J. M. Barrios and B. Bustos. Content-based video copy detection: Prisma. In *TRECVID Workshop, NIST*, 2010.

[3] J. Chen and J. Jiang. University of Bradford, content based copy detection task. In *TRECVID Workshop, NIST*, 2008.

[4] T.-T. Do, E. Kijak, L. Amsaleg, and T. Furon. Enlarging hacker's toolbox: deluding image recognition by attacking keypoint orientations. In *IEEE ICASSP*, 2012.

[5] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Challenging the security of content-based image retrieval systems. In *IEEE MMSP*, 2010.

[6] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Deluding Image Recognition in SIFT-based CBIR Systems. In *ACM Multimedia in Forensics, Security and Intelligence*, 2010.

[7] T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg. Understanding the security and robustness of SIFT. In *ACM Multimedia*, 2010.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 1981.

[9] N. Gengembre and S.-A. Berrani. The Orange Labs real time video copy detection system. In *TRECVID Workshop, NIST*, 2008.

[10] X. Hou, J. Harel, and C. Koch. Image signature: Highlighting sparse salient regions. *IEEE Transactions on PAMI*, 34(1), 2012.

[11] C.-Y. Hsu, C.-S. Lu, and S.-C. Pei. Secure and robust SIFT. In *ACM Multimedia*, 2009.

[12] H. Jégou, M. Douze, G. Gravier, C. Schmid, and P. Gros. INRIA LEAR-TEXMEX: Video copy detection task. In *TRECVID Workshop, NIST*, 2010.

[13] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on PAMI*, 2011.

[14] H. Lejsek, F. H. Amundsson, B. T. Jonsson, and L. Amsaleg. NV-Tree: An efficient disk-based index for approximate search in very large high-dimensional collections. *IEEE Transactions on PAMI*, 31(5), 2009.

[15] Y. Liang, B. Cao, J. Li, C. Zhu, Y. Zhang, C. Tan, G. Chen, C. Sun, J. Yuan, M. Xu, and B. Zhang. Thu-img. In *TRECVID Workshop, NIST*, 2009.

[16] Z. Liu, T. Liu, D. C. Gibbon, and B. Shahraray. Effective and scalable video copy detection. In *Multimedia Information Retrieval*, 2010.

[17] Z. Liu, T. Liu, and B. Shahraray. AT&T Research, content-based copy detection. In *TRECVID Workshop, NIST*, 2009.

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

[19] O. Orhan, J. Hochreiter, J. Poock, Q. Chen, A. Chabra, and M. Shah. University of central florida, content based copy detection and surveillance event detection. In *TRECVID Workshop, NIST*, 2008.

[20] P. Over, G. Awad, M. Michel, J. Fiscus, W. Kraaij, and A. F. Smeaton. An overview of the goals, tasks, data, evaluation mechanisms and metrics. In *TRECVID Workshop, NIST*, 2011.

[21] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.

[22] C. Sun, J. Li, B. Zhang, and Q. Zhang. Thu-img. In *TRECVID Workshop, NIST*, 2010.

[23] A. Vedaldi and B. Fulkerson. VLFeat - an open and portable library of computer vision algorithms. In *ACM Multimedia*, 2010.

[24] C. wah Ngo, S. ai Zhu, H. khoon Tan, W. lei Zhao, and X. yong Wei. VIREO: Semantic indexing, known-item search, and content-based copy detection. In *TRECVID Workshop, NIST*, 2010.

[25] D. B. Wan-Lei Zhao and T. M. Breuel. Semantic indexing & content-based copy detection tasks. In *TRECVID Workshop, NIST*, 2011.