A ROBUST PIPELINE FOR LOGO DETECTION

C. Constantinopoulos, E. Meinhardt-Llopis, Y. Liu *

Barcelona Media
Av. Diagonal 177, 08018 Barcelona
enric.meinhardt@upf.edu

V. Caselles †

Universitat Pompeu Fabra
C/Tànger 122, 08018 Barcelona
vicent.caselles@upf.edu

ABSTRACT

We present a method for detecting appearances of logos in low-resolution video sequences. The method is based on matching of SIFT descriptors, plus several heuristics. The logos must come from a small database of possible logos. The emphasis is not on speed but on reliability, although the method can be executed in real time using a parallel computer.

***Index Terms*—** logo detection, SIFT, RANSAC, tracking

1. INTRODUCTION

The detection of logos in broadcast videos of sport events is an interesting and challenging problem. This problem has commercial interest, because advertisers pay people to manually inspect the broadcasts to assess the visibility of their brands. Logo detection can be used to automate this process, but the quality of the results is not yet comparable to human visual inspection. While close-up shots of non-occluded logos are very easy to detect using standard methods, this is almost never the case.

Most of the time the available data is of low quality, and there are several reasons for this. For example the resolution could be 720×576 with two interlaced fields. The deinterlacing process sometimes fails producing serious degradations, like ghosting and blurring. There is also another source of blurring: as the camera follows the action, many captured logos would be out of focus or distorted by motion blur. Another problem is the perspective of the logo. Often the logos appear from an oblique angle. Scale is also an issue, because there are cases where a logo is only a few pixels high. We should also mention that often the logos are partially occluded (e.g., when a football player runs in front of them).

(a) Input (b) Output

Fig. 1. The goal of this paper is simultaneous detection of multiple logos in low-quality videos.

Finally the surface that carries the logo can degrade its appearance. For example a curved surface like the side of a car causes non-rigid deformations, while a LED screen is plane but causes discontinuities. Considering all the above reasons, it is not a surprise that challenging cases of logo detection are common in practice.

In this report, we present a system for offline usage that has high accuracy. The proposed system has two layers: a low level detector that operates on the individual frames, and a high level detector that operates on the sequence level, integrating the low level results. At the lower level, we detect as many logos as possible on each frame. At the high level we process the whole video three times. At the first pass we detect logos based only on the target image. At the second pass we detect logos based on the logo instances that we detected previously. Finally at the third pass we track all the detected instances in order to retrieve the missing detections and stabilize the results.

Logo detection can be regarded as a particular case of object detection. The additional constraints allow stronger detections than would be possible using a general object detector. Specifically, in logo detection the two images to be matched play very different roles and have different quality; most of the matching keypoints are expected to be outliers; there can be zero, one or many detections, and these cases must be told apart automatically. Since the pioneering method called "Video Google" [1] for general object retrieval, several works have improved on it with the focus on finding small synthetic images on low-resolution videos [2], [3], [4].

One standard method for logo detection is the following: first we compute the descriptors of a model logo and of the image to be analyzed; then, we pair the most similar descriptors of each image; and finally we try to find a subset of these

*This work was partially funded by Mediapro through the Spanish project CENIT-2007-1012 i3media and by the Centro para el Desarrollo Tecnológico Industrial (CDTI), within the Ingenio 2010 initiative. C. C. and Y. L. acknowledge partial support from the Torres Quevedo Program from the Ministry of Science and Innovation in Spain (MICINN), co-funded by the European Social Fund (ESF). V. C. and E.M. acknowledge partial support by MICINN project, reference MTM2009-08171, and by GRC reference 2009 SGR 773.

†V. C. also acknowledges partial support by "ICREA Acadèmia" prize for excellence in research funded by the Generalitat de Catalunya.

pairs that matches closely by an affine map. Usually, this subset and this affine map are found by RANSAC: randomly try small subsets of pairs, until one subset is found that matches correctly by an affine map, which is then enlarged as much as possible to a matching subset.

The standard method gives good results for large and high-resolution logos, but it is not immediately useful for solving the problem. For that, we developed an improved method. Our improvements to the standard method are the following: adaptation of RANSAC to the real asymmetric case, where the model logo and the query image have much different resolution and quality; fix automatically the thresholds of RANSAC using *a contrario* methods; use prior knowledge about the images to optimize the pairing of points and the random sampling of subsets; find more than one affinity within the set of pairs (in order to have multiple detections); build an orbit of model logos to improve the detection; track the detections along a video sequence; evaluate and report the quality and precision of each detection, etc. We have put together all these new techniques into a pipeline for video processing. The input of the pipeline is a video, and the output is a summary of the logos appearing on each frame, annotated with their positions and their visibility (according to different visibility criteria).

The contribution of our work is a method for logo detection with performance comparable to visual inspection, specially for images where the logos are small and severely distorted. The rest of the paper is organized as follows. In Section 2 we describe a method for detecting logos on a single image. In Section 3 we describe a method for detecting logos on a video, which builds on the method of Section 2. In Section 4 we describe the results of our experiments. Finally, we summarize our conclusions in Section 5.

2. FRAME-LEVEL DETECTION

We have investigated many ways of making a robust detection method. Our approach is based on the SIFT method proposed in [5]. The key ideas behind SIFT is first to find keypoints in the image that are scale and rotation covariant. Second, to compute descriptors that are invariant to said transformations. These descriptors are based on histograms of gradients in the vicinity of the keypoints. If we compute the SIFT descriptors in the logo image and the video frame, then we can match the corresponding keypoints and compute an affine transformation between the two images. Since the invariance of SIFT to the observed image distortions is not perfect, we proceed as in [6], and build a set of distorted versions of the images, before matching them.

2.1. Standard methodology (SIFT+RANSAC)

Recall that many different problems can be regarded as clustering of pairs of points between two images: stereo matching, optical flow segmentation, and logo detection, which is our goal. The keypoints that we use are standard SIFT keypoints, which we match using and absolute threshold to find a list of pairs of points. The clustering technique we propose is a variation of RANSAC [5], [7].

Thus, the basic building block of our method is the following **core algorithm**, that finds occurrences of one image into another.

1. Get SIFT keypoints of the two images
2. Find matching pairs (see §2.3)
3. Find affinities between subsets of pairs (see §2.4)

Notice that multi-RANSAC gives a (possibly empty) list of affinities. Each of these affinities corresponds to a detection. These affinities map the bounding box of the model logo to a set of parallelograms in the image domain, which we call the *detection rectangles*.

The problem of this "core" method is that, as stated, it does not work very well for logo detection. Besides being very slow, it fails to detect the small and distorted logos which appear in many low resolution videos. Our goal is to improve the reliability of this method to attain a performance comparable to human visual inspection. We do not focus on running time, but on the precision of the output. However, some of the proposed modifications increase the speed of the algorithm.

The rest of this section is a list of the proposed heuristics and improvements to the core algorithm for detecting one logo on one frame. The next section deals with further improvements specific to the detection of multiple logos on video sequences.

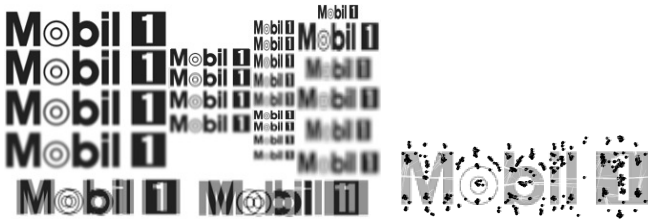
2.2. Orbit of logo distortions

An orbit of an image is simply a list of distorted versions of the image (see Figure 2). We compute the keypoints of each image in the orbit and store them in a single list, using a common system of coordinates. There are two reasons for using orbits: The first reason is that we need more keypoints, because the standard SIFT detector produces too few keypoints on small resolution logos (about 10 keypoints). The second reason is that the query images have very large distortions, way beyond the practical range of invariance of SIFT.

We use orbits to enlarge the set of keypoints of the model logos and of the query frames. The set of distortions in each case is different. For the model logo we build a small orbit with some combinations of zoom, blur, horizontal motion, and de-interlacing artifacts. Note that there are no rotations nor shears in the orbit: most logos in broadcast videos appear in horizontal position. For the query frames, we build an orbit of four images: the original frame, the de-interlaced frame, a double-resolution version of the original frame, and a double-resolution de-interlaced version.

2.3. Thresholding the SIFT distance to obtain matchings

Once we have the SIFT keypoints and descriptors of the image and the model logo, we have to match them. There exist several matching criteria, depending on whether we want a



(a) Orbit images (b) Orbit keypoints

Fig. 2. A small orbit of 25 images. Each image has between 12 and 90 keypoints. The whole orbit has 870 different keypoints. The smallest images in the orbit have size 50×20 and are severely distorted.

Fig. 3. Our database of 21 logos

relative or absolute threshold, and we want to treat symmetrically or asymmetrically the two images to be matched. In our case, we found the following criterion the most useful: fix an absolute threshold for the Euclidean distance between SIFT descriptors of $t = 290$. Then, for each keypoint on the video frame, find the nearest keypoint of the model logo, as long as the distance is smaller than t. This matching is done by exhaustive search, which is the fastest method for datasets of our size. The result is a list of about 800 pairs for each frame, which is a reasonable input size for RANSAC.

There is a strong argument against using a relative threshold of SIFT distances for logo detection. The reason is that there can be more than one appearance of a logo in the query image, and the logo may have repeated letters. Thus, the closest matchings may all correspond to correct detections. In the ideal case, all these correct matchings will have a descriptor distance arbitrarily small, but the relative threshold may reject them. An absolute threshold does not have this problem, and this is the reason of our choice.

2.4. Automatic thresholding of RANSAC

RANSAC-like methods are used to fit a model to data points where a lot of them are outliers. Typically, these methods fix a maximum allowed error s, and then find the largest set of inliers than can be matched by some model with an error smaller than s. Since the set of possible models is very large, exhaustive search is not practical, and an educated random sampling is used instead.

Using a fixed maximum error is practical and, in many cases, it makes sense. However, we can think about the following example. Suppose that we have 1000 pairs of points in two images, and we find an affine transformation that matches 4 of these pairs with a maximum error of 1 pixel. Then, we also have another affine transformation that matches 900 of these points with a maximum error of 3 pixels. Clearly, the second transformation is better, because it explains much better the given data. However, if the error threshold is set to 2, the first transformation will be detected as good enough, and the second one will be ignored. This caricature problem illustrates an intrinsic problem of working with a fixed error threshold and optimizing only on the number of inliers. In practice, this problem manifests itself as an extreme sensitivity of the method to any fixed threshold. Thus, instead of using a fixed threshold, we need *a criterion to compare candidate transformations of different size, as a function of both their error and their number of inliers*.

Our approach is an adaptation of the method to find fundamental matrices in [7] to the case of affine transformations. Let S be a set of pairs of points in the plane. We want to find subsets of S which are well-approximated by affine maps. Given a subset $S' = \{(m_i, m_i')\} \subseteq S$ and an affine map A, we define the following scale-free error measure of A:

$$\alpha(A, S') := \sqrt{\frac{\pi}{|\Omega|}} \max_{S'} \{\|m_i' - A m_i\|\}$$

where $|\Omega|$ is the area of the image domain. Following [7], we define for each affine map A and each subset S' of size k of S, this *score* measure:

$$m(S, A, S') := -\log \left[(n-3) \binom{n}{k} \binom{k}{3} \alpha(A, S')^{2(k-3)} \right].$$

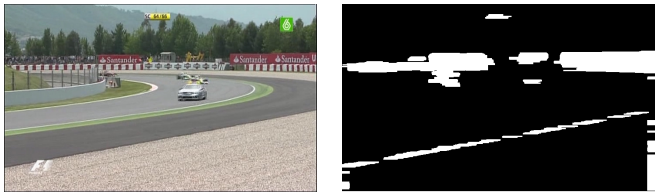
We say that A is *meaningful*, or that A describes correctly the match S', when $m(S, A, S') > 0$.

The score measure m can be regarded as a function $m = m(k, t)$, which is used to evaluate a candidate subset of k inliers whose maximum error is t. The condition $m(k, t) > 0$ defines the accepted subsets of inliers, but it is actually a threshold that can be adjusted. The advantage of this threshold is that it has a clear meaning: if we use $m(k, t) > m_0$ as a condition for acceptance, the expected number of false positives of the detector will be around e^{-m_0}. Thus, a high m_0 produces a robust detection, and a low m_0 produces a sensitive detection.

Using RANSAC with the score m gives rise to an algorithm that finds zero or one affinities for any given set of pairs S. We have extended this method to find an arbitrary number of affinities, by running it recursively on the set of outliers until no new affinities are found. This is what we call "*a contrario* multi-RANSAC".

2.5. Prior on the allowed affinities

Common RANSAC is blind. It selects three point pairs, and evaluates the affine map determined by them. Determining an affine map from three point pairs is a fast application of a constant formula. However, to evaluate an affine map, all the point pairs must be sorted according to their error, and then the best threshold for the error must be selected. This is a very expensive operation, because all the input data



(a) Video frame (b) Dilated color mask

Fig. 4. Reduction of search space using a color mask. Out of the 12628 SIFT keypoints on the original frame, only 2547 fall inside the masked region.

must be traversed. Since almost no affinities give meaningful matchings all this work is wasted. How can we know in advance, without expensive computations, whether an affinity $\binom{x}{y}' = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)\binom{x}{y} + \binom{p}{q}$ will not be meaningful? For a start, the determinant $ad - bc$ must be positive, since we do not care for mirror-images of logos. This observation allows us to discard 50% of the candidate affinities before doing any computations with them. By further constraining the coefficients a, b, c, d, limiting reasonably the zoom, shear and tilt of the logos, we can discard up to 90% of the affinities. This is the single most important optimizing heuristic that we use. It produces a 100-fold reduction of running time for low-resolution videos. It amounts to using a uniform prior on a compact subset of the 6-dimensional space of all affinities.

2.6. Keypoint rejection based on color mask

In many applications it is easy to discard large portions of the query image by a simple criterion (e.g., a color mask). By a standard reasoning, multiplication of the input size by p results in multiplication of the running time by p^3. This is a huge improvement, even for moderately small $p < 1$. This improvement can be easily achieved if we have prior information on the colors present on the scene, the colors of the logo, or the location of the logo. For example, if we look for logos in videos of football games, we can safely remove all the features which are on green regions, which will correspond to the grass. This may probably reduce the number of keypoints to 10%, boosting the efficiency a thousand times. See Figure 4 for an example of a simple color mask. In this case a naive RGB mask ($R > \max(G, B)$) was used. A more elaborate approach could be based in the color histogram method proposed in [8]. This kind of masks are very useful to boost the performance of most detectors.

2.7. Non-uniform sampling using the descriptor metric

This is a "soft" version of the absolute threshold on descriptor distances. The basic idea is that the pairs of features with similar descriptors are more likely to be correct matchings than the pairs of features with different descriptors. Thus, descriptor distance predicts the correctness of the match. In common RANSAC, the pairs are sampled uniformly. In our implementation (as proposed similarly in [9]), the sampling is biased towards the better matchings. That way, a good correspondence is likely to be found sooner.

Fig. 5. Image containing 154 logos. After 1 hour of common RANSAC trials, only 10 logos have been found. A few seconds of "localized RANSAC" suffice to find all the logos. This image has size 2304×3072 and has 63018 SIFT keypoints.

2.8. Non-uniform sampling using localization

Recall that the expected running time to find a meaningful affinity is proportional to the cube of the number of outliers. This is specially problematic for the detection of small logos in large images. An extreme case is shown in Figure 5. However, a small logo can be easily detected by cropping the image around that logo. This crop can be simulated in real time in the following way. Instead of sampling three keypoints uniformly over the whole image, sample the first point uniformly over the whole image, and the other two uniformly on a small neighborhood of the first point. This kind of local sampling can be implemented efficiently by distributing all the keypoints into overlapping rectangular regions of small size. We call this simple variation of RANSAC, which is part of the computer vision folklore, "localized RANSAC".

3. VIDEO-LEVEL DETECTION

At the first stage of the method we search throughout the video sequence. For each frame we use the low level detection method described in Section 2 (Figure 6). In this way we localize with high confidence as many instances of the target logo as is possible. Our aim here is to avoid any false detection, and collect a set of logo instances that is representative of the video. Thus it is not necessary to check every frame, but a subset of frames uniformly distributed inside the video.

At the second stage of the method we search again throughout the video, but this time our target is the original logo as well as the other instances detected at the first stage. Introducing the detected logos as targets lets us detect more difficult cases. We should note that we do not try to detect each logo instance independently, instead we merge their keypoints with those of the original logo. As in the previous stage, it is not necessary to search for logos at each frame, thus we suggest searching once every second.

At the third stage of the method we track the detections. Given all the detected instances at two frames that are one

(a) Primary (b) Secondary (c) Tracking

Fig. 6. Three rounds of detections. The primary detection matches the model logo against a single frame and finds some logos. The secondary detection adds the best of them to a "local orbit", and re-runs the detector using the enlarged orbit. The tertiary detection tracks each detected logo for the next few frames.

second apart, we track them forward and backward. Then we remove the overlapping detections.

3.1. Secondary detection (local orbits)

Many false negatives occur because the descriptors of the logo and the video frame are too different. This is due, in part, to distortions not taken into account when building the orbit. The idea of "secondary detection" (see Figure 6(b)) is to add the keypoints of the best detections to the orbit, and re-run the detector using the enlarged orbit. That way, the best detections serve as an anchor between the model logo and the lower-resolution logos, which can not be detected directly.

3.2. Tracking between frames

To solve flickering of detections and enforce temporal stability we *track* the detections of one frame to the next one, as shown in Figure 6(c). The tracking is very fast, since we have already computed the keypoints. Moreover, the tracking is only computed when a logo detection disappears on a region of the image domain. To track a logo detection from one frame to the next one, we first define a rectangle that contains the detected logo with some margin. Then, we select the keypoints of the next frame which fall inside this rectangle. Finally, we use RANSAC to match the keypoints between the two rectangles.

The process of tracking has several advantages, and can be used in different ways: since it is faster than the primary detections, it permits us to skip the primary detection on a few frames, and then recover the lost detections using tracking. Also, it permits to follow a logo appearance whose quality fades away, well beyond the threshold of detectability for primary detection. This happens, for example, when a F1 containing the logo moves away from the camera.

3.3. Quality descriptors

When lowering the detection threshold of *a contrario* RANSAC, false detections appear. There are two kinds of false detections. The first kind is spurious detections, which happen even for random data (because we have increased the expected number of false alarms), in the middle or across textured regions. The second kind is partial detections of other

Fig. 7. False partial detection. The letters "nder" of "Santander" match very well the letters "odaf" of "Vodafone".

logos, where some letter or group of letters of the model logo is matched to the same letters on a different logo that appears on the image. See Figure 7 for an example.

Quality descriptors, independent to the meaningfulness score, can be used to reject the spurious detections on a post-processing step. Quality descriptors are numbers computed for each detection, which are increasing with respect to the quality of the detection. By setting minimum thresholds on these numbers, we aim to reject false positives. A detailed study of quality descriptors is very dependant on the application. Let us list the ones that we found most useful: the number of inliers of a detection, the maximum or average error of the matched inliers, the $\%$ of area of the model logo covered by inilers, the $\%$ of outliers inside the detection rectangle (within the query image), the correlation of the model logo against the detection rectangle (mapped back to the same system of coordinates).

3.4. Multiple logo detection

A common source of false positives is partial matchings to parts of other logos. For instance, when two different logos have some letters in common, an occurrence of either of them will trigger the detection of both of them. In general, the correct detection will have a higher score than the incorrect one. Thus, a simple criterion can be used to pick the correct one: whenever we have some overlapping detections, keep only the one with higher score. We can turn this to our favor: even when we are only interested in detecting one brand, adding more logos from other brands to the database allows us to lower the detection thresholds, and detect more correct logos of the initial brand.

4. RESULTS

We have run our detector on two short video sequences of a few thousand frames (see Figures 10 and 11). These videos have been manually annotated to describe how many logos of each brand appear on each frame, and we have taken this annotation as the ground truth. The quality of the results is nearly the same as the manual annotation. There are virtually zero false positives, and the false negatives occur only in extremely degraded images, which required a certain amount of ingenuity to be annotated. See Figure 9 for some examples of these false negatives.

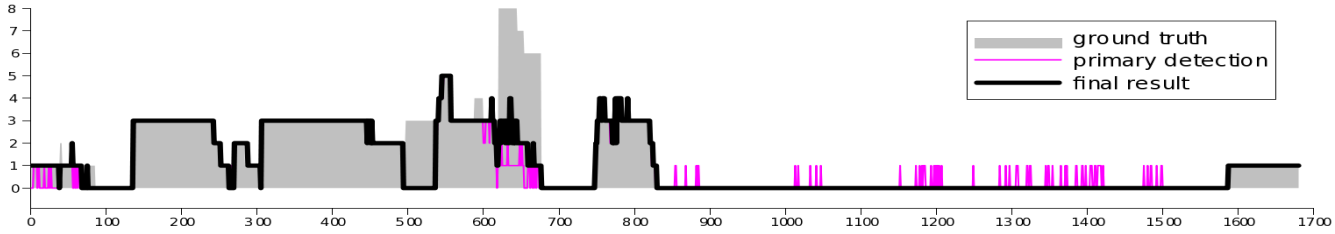


Fig. 10. Results of the detector for the logo "santander" on a sequence of 1680 frames of Formula 1 footage. The graphs show the number of appearances of this logo on each frame, according to the ground truth (obtained by visual inspection), the primary detection, and the final result.

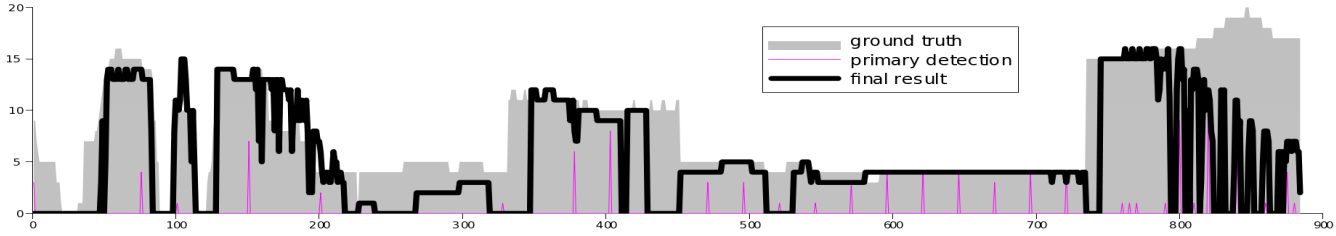


Fig. 11. Results of the detector on a sequence of 900 frames of Football footage. The graphs show the number of appearances of a logo on each frame, according to the ground truth (obtained by visual inspection), the primary detection, and the final result. In this experiment, the primary detection was only run every 25 frames, and the rest of the work was done by the local orbits and the tracking.

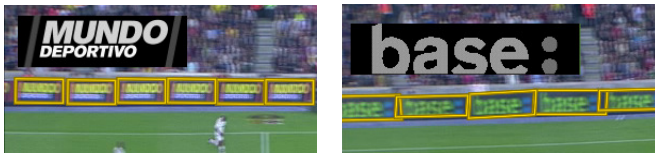


Fig. 8. Examples of the worst-case quality of logos that we have detected. Each image shows the model logo and the detection rectangles in a detail of the frame. Notice that the logo appearances are severely distorted, and barely distinguishable by visual inspection.

Fig. 9. Details of two frames where false negatives occur, the frame 520 (on the left) and the frame 650 (on the right) from the F1 sequence analyzed on Figure 10.

5. CONCLUSIONS

We have presented a robust algorithm for logo detection in low-resolution videos. The algorithm is based on a combination of standard techniques, which alone do not produce practical results. Our results are comparable to human visual inspection, and the running time is about $10s$ per frame (of size 720×576) on a single processor of a current commodity computer.

6. REFERENCES

[1] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. ICCV*, 2003, vol. 2, pp. 1470–1477.

[2] L. Ballan, M. Bertini, A. Del Bimbo, and A. Jain, "Automatic trademark detection and recognition in sport videos," in *Proc. ICME*, 2008, pp. 901–904.

[3] A.D. Bagdanov, L. Ballan, M. Bertini, and A. Del Bimbo, "Trademark matching and retrieval in sports video databases," in *Proc. of the Int. Workshop on Multimedia Information Retrieval*, 2007, pp. 79–86.

[4] J. Schietse, J.P. Eakins, and R.C. Veltkamp, "Practice and challenges in trademark image retrieval," in *Proc. Int. Conf. on Image and Video Retrieval*, 2007, pp. 518–524.

[5] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] J.M. Morel and G. Yu, "ASIFT: A new framework for fully affine invariant image comparison," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 438–469, 2009.

[7] L. Moisan and B. Stival, "A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix," *IJCV*, vol. 57, no. 3, pp. 201–218, 2004.

[8] F. Pelisson, D. Hall, O. Riff, and J.L. Crowley, "Brand identification using gaussian derivative histograms," in *Proc. of the Int. Conf. on Computer Vision Systems*, 2003, pp. 492–501.

[9] O. Chum and J. Matas, "Matching with PROSAC-progressive sample consensus," in *Proc. Conf. on Computer Vision and Pattern Recognition*, 2005, vol. 1, pp. 220–226.