Efficient Dense-Field Copy–Move Forgery Detection

Article *in* IEEE Transactions on Information Forensics and Security · November 2015

DOI: 10.1109/TIFS.2015.2455334

3 authors:

Davide Cozzolino
University of Naples Federico II
65 PUBLICATIONS **4,611** CITATIONS
SEE PROFILE

Giovanni Poggi
University of Naples Federico II
141 PUBLICATIONS **4,862** CITATIONS
SEE PROFILE

Luisa Verdoliva
University of Naples Federico II
161 PUBLICATIONS **8,449** CITATIONS
SEE PROFILE

Efficient dense-field copy-move forgery detection

Davide Cozzolino, Giovanni Poggi and Luisa Verdoliva

***Abstract*—We propose a new algorithm for the accurate detection and localization of copy-move forgeries, based on rotation-invariant features computed densely on the image. Dense-field techniques proposed in the literature guarantee a superior performance w.r.t. their keypoint-based counterparts, at the price of a much higher processing time, mostly due to the feature matching phase. To overcome this limitation, we resort here to a fast approximate nearest-neighbor search algorithm, PatchMatch, especially suited for the computation of dense fields over images. We adapt the matching algorithm to deal efficiently with invariant features, so as to achieve higher robustness w.r.t. rotations and scale changes. Moreover, leveraging on the smoothness of the output field, we implement a simplified and reliable post-processing procedure. The experimental analysis, conducted on databases available online, proves the proposed technique to be at least as accurate, generally more robust, and typically much faster, than state-of-the-art dense-field references.**

***Index Terms*—Digital image forensics, copy-move forgery detection, PatchMatch.**

I. Introduction

In the last few years, a large number of techniques have been proposed for the detection and localization of digital image forgeries [1]. Research has focused especially on passive techniques, which detect manipulations based only on the analysis of the image content, and many papers deal with copy-move forgeries [2]. In this case one or more regions of an image are cut and pasted elsewhere, in the same image, in order to duplicate or hide objects of interest. In fact, these forgeries are extremely simple to perform with modern image editing tools, such as Photoshop or Gimp. Detecting them, however, can be challenging, especially in the occlusive case, where pieces of smooth background are used to cover undesired details.

All detection algorithms proposed in the literature follow a common pipeline [3] based on three steps

- *feature extraction:* a suitable feature is computed for each pixel of interest, expressing the image behavior in its neighborhood;
- *matching:* the best matching of each pixel is computed, based on the associated feature;
- *post-processing:* the offset field, linking pixels with their nearest neighbors, is filtered and processed in order to reduce false alarms.

These operations can be carried out for each pixel of the image to generate a dense offset field, or for just some selected keypoints, in which case the field is sparse. Keypoint-based methods, working on a relatively small set of pixels, are usually much faster than those based on dense matching. A first robust keypoint-based algorithm was proposed in [4], where SIFT features are used to deal with different types of distortion. In particular, geometric distortions are modeled as affine transforms of pixel coordinates. Parameters of such transforms are estimated by means of the RANSAC algorithm [5]. More recent variants of this method take special care of multiple duplicated regions by hierarchical agglomerative clustering [6] or the J-linkage algorithm [7]. Following this path, other papers have considered to characterize the keypoints using well-known local descriptors, like SURF [8], LBP [9] and DAISY [10], and other local features [11], [12].

Unfortunately, these methods are intrinsically less accurate than dense-field ones. This is especially trye when copy-moves involve only smooth regions, which is typically the case with occlusive forgeries. This performance gap, which appears clearly in the benchmarking paper [3], is the main reason driving us to focus on the dense-field approach.

The problem, in this case, is complexity, since all pixels undergo the three phases of feature extraction, matching, and post-processing. Therefore, feature extraction is required to be intrinsically simple, and to produce features as short as possible to speed-up the matching phase. Using RGB values is a possible choice [13], but the resulting features tend to be unnecessarily long, and performance may be severely affected by JPEG compression, noise addition, and other common distortions. In the literature, to improve robustness, features are typically extracted through some transforms, like DCT [14], [15], [16], Wavelet [17], PCA [18], SVD [19], reducing also their length, thanks to the decorrelation of coefficients. Such features, however, do not perform well in the presence of rescaling and rotation. Therefore, a significant effort has recently been devoted towards the definition of features that deal satisfactorily with these situations. Circular harmonic transforms are well-suited to provide rotation-invariance, and several possibilities have been tested to this end, including Zernike moments [20], and polar sine and cosine transforms [21], [22]. As for scale-invariance, research has mostly focused on variations of the Fourier-Mellin Transform [23], [24], [25], [26], based on a log-polar sampling. The same sampling scheme is also carried out directly in the spatial domain in [27], producing a simple one-dimensional descriptor.

Besides feature selection, the literature has devoted much attention to the matching phase itself. In fact, exhaustive search of the best matching (nearest neighbor) feature is prohibitive due to its huge complexity, and faster techniques must be devised to produce the offset field in a reasonable time. A significant speed-up can be obtained by adopting

D.Cozzolino, G. Poggi and L.Verdoliva are with the DIETI, Università Federico II di Napoli, Naples, Italy.

E-mail: {davide.cozzolino, poggi, verdoliv}@unina.it.

some approximate nearest-neighbor search strategy, in which case accuracy becomes a further aspect of interest. Some techniques [23], [27] rely on simple lexicographic sorting, but this approach is very sensitive to noise and other forms of impairment. Robustness improves significantly with more sophisticated fast search techniques, like kd-trees [28], used in [13], [3], or locality sensitive hashing [29], considered in [20], [21]. These state-of-the-art matching techniques, however, are designed to work for very generic problems, like the retrieval of documents in large collections of data. Therefore, they do not take into account a major circumstance of interest in this context, namely, that the we are looking for a nearest-neighbor *field*, where features are extracted from a real-world image, and are not just a collection of unrelated queries.

This observation is at the core of the present work in which we propose an efficient and accurate technique for copy-move detection and localization, which deals successfully also with a number of geometric transformations. We follow the general workflow considered in [3], but propose innovative and efficient solutions for most of the key steps of the workflow. Matching of dense features, in particular, is carried out through the PatchMatch algorithm [30], specific for nearest-neighbor search over images, which greatly reduces the processing time while providing an accurate and regular offset field. Efficiency is pursued also in the other processing steps, by selecting compact, low-complexity features, and by implementing a simple and reliable post-processing scheme, which fully exploits the smoothness of the PatchMatch offset field. As a result, our dense-field technique turns out to be nearly as fast as keypoint-based techniques, but exceedingly more reliable than them. Moreover, by selecting suitable scale and/or rotation-invariant features, higher robustness w.r.t. a number of geometric distortions is also achieved. A suitably modified [31] version of PatchMatch is implemented to deal efficiently with this case.

Following the above general description, in the rest of the paper we will first focus on efficient matching, in Section 2, and post-processing, in Section 3; then, in Section 4, we will consider and discuss several features with scale-invariance and rotation-invariance properties. In Section 5 we will carry out a thorough performance analysis on databases available online. Finally, Section 6 draws conclusions.

II. Computing a dense NNF via PatchMatch

Let

$$I = \{I(s) \in R^K, s \in \Omega\} \tag{1}$$

be an image defined over a regular rectangular grid Ω. With each pixel[1], s, we associate a feature vector, $f(s)$, which describes the P-pixel image patch centered on s. In the simplest case, $f(s)$ might just be the KP-vector formed by stacking all image values observed in the patch. More often, to improve efficiency, the feature is a compact description of the patch, with length much smaller than KP.

Given a suitable measure of distance between features, $D(f', f'')$, we define the nearest neighbor of s as the pixel, $s' \in \Omega, s' \neq s$, which minimizes this distance over the whole image. Rather than the nearest neighbor field (NNF) itself, in the following we will often consider the equivalent offset field, $\{\delta(s), s \in \Omega\}$, where

$$\delta(s) = \arg \min_{\phi: s+\phi \in \Omega, \phi \neq 0} D(f(s), f(s+\phi)) \tag{2}$$

and, of course, the NN is $s' = s + \delta(s)$.

Finding the *exact* NNF is computationally very demanding, even for a relatively small image, since the complexity grows quadratically with the image size. Most real-world applications, however, do not really need the exact NN, and perform almost as well with some good approximation of it. Finding an approximate NN can be orders of magnitude less expensive than finding the exact one, depending on feature statistics and on the desired accuracy. As a matter of fact, a large number of techniques have been proposed in recent years for this task, the most promising of which are based on kd-trees and on hashing. Indeed, some of these techniques have been already applied in the context of copy-move detection, as in [3], where feature matching is carried out through the kd-tree based methods of the FLANN package [28], or in [20], where locality-sensitive hashing [29], [32] is used.

These techniques, however, are not really suited for the problem under analysis, as they consider each feature to match as an *independent* query, neglecting altogether the spatial regularity typical of natural images. Image smoothness and self-similarity, instead, imply that the NNs of close pixels are very often spatially close themselves. By exploiting this simple property one can improve both the efficiency and the quality of the final NNF. Indeed, the search can be accelerated by using the offsets of neighboring pixels as initial guesses for the current one. This approach is well-known in the denoising literature, where it is exploited for nonlocal filtering techniques based on block-matching. Moreover, by resorting to spatial prediction, smoother offset fields are automatically obtained. This is extremely valuable in the context of copy-move detection, as it allows one to avoid complex (and time-consuming) post-processing algorithms to regularize the field afterwards.

Based on these considerations, we resort here to a technique recently proposed for the specific problem of fast NNF computation over images.

A. The PatchMatch algorithm

PatchMatch [30] is a fast randomized algorithm which finds dense approximate nearest neighbor matches between image patches.

Initialization. The offset field is initialized at random, as

$$\delta(s) = U(s) - s \tag{3}$$

where $U(s)$ is a bi-dimensional random variable, uniform over the image support Ω. We note explicitly, here, that $\delta(s) = 0$ is discarded, as it corresponds to a trivial and useless solution. Likewise, since we are looking for matches relatively far apart from the target, we exclude all offsets smaller than a given threshold, $\|\delta(s)\|_\infty < T_{D1}$, a condition applied implicitly in all further developments. Most of the initial random offsets are

[1]To the extent possible, to keep a light notation, we avoid double indices and boldface, using the single normal-type variable s to indicate pixel location.

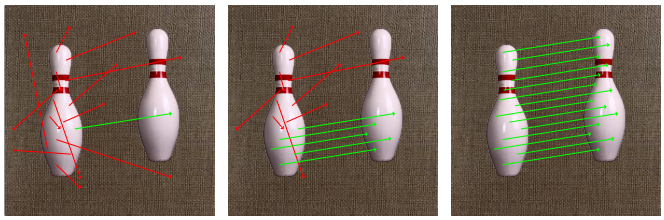


Fig. 1. Offset propagation. Initially (left), most offsets are wrong (red); a random good offset (green) propagates along the scanning order (center) until the whole copy-moved region is covered (right).

just useless, but it is very likely that a certain number of them will be optimal or near-optimal. The main idea of PatchMatch is to quickly propagate such good offsets, updating iteratively the whole field. In the generic iteration, there are two phases: propagation and random search.

Propagation. In this phase the image is raster scanned top-down and left-to-right, and for each pixel s the current offset is updated as

$$\delta(s) = \arg \min_{\phi \in \Delta^P(s)} D(f(s), f(s+\phi)) \tag{4}$$

where $\Delta^P(s) = \{\delta(s), \delta(s^r), \delta(s^c)\}$, and s^r and s^c are the pixels preceding s, in the scanning order, along rows and columns, respectively. In practice, the algorithm checks whether the offsets associated with the causal neighbors improve the matching quality w.r.t. the current one. Therefore, if a good offset is available for a given pixel of a region with constant offset, this will very quickly propagate, filling the whole region below and to the right of it. To avoid biases, the scanning order is then reversed (bottom-up and right-to-left) at every other iteration, In Fig. 1, using a simple toy example, we provide some insight into the rationale of this procedure for the copy-move detection application. On the left, some of the initial offsets are shown (only on a bowling pin, to avoid cluttering the figure), which are all wrong (red) except for one of them (green), which is correct by chance. After the first iteration of the algorithm, the correct offset propagates to the bottom-right part of the pin (center), and to all of it after the second iteration (right). With more complex geometries, the complete propagation might require some more iterations.

Random search. The above propagation procedure is greedy, and as such suboptimal, depending on the quality of the random initialization. Therefore, to minimize the risk of being trapped in local minima, after the updating of equation (4), a random search phase is also considered, based on a random sampling of the current offset field. The candidate offsets $\delta_i(s), i = 1, \ldots, L$ are chosen as

$$\delta_i(s) = \delta(s) + R_i \tag{5}$$

where R_i is a bi-dimensional random variable, uniform over a square grid of radius 2^{i-1}, excluding the origin. In practice, most of these new candidates are pretty close to $\delta(s)$, but large differences are also allowed, with small probability. The random-search updating reads therefore as

$$\delta(s) = \arg \min_{\phi \in \Delta^R(s)} D(f(s), f(s+\phi)) \tag{6}$$

where $\Delta^R(s) = \{\delta(s), \delta_1(s), \ldots, \delta_L(s)\}$.

For an image of, say, 1024×1024 pixels, $L \leq 10$. Considering that the procedure typically converges after a few iterations, the whole computational load is in the order of 10^2 feature distance computation per pixel, as opposed to 10^6 for full-search, which fully explains the algorithm speed. Of course, PatchMatch relies on the implicit hypothesis that the NNF is mostly regular, and in particular regular over the regions of interest where a match is looked for, otherwise the crucial propagation step would be basically ineffective. However, this is exactly the condition encountered in copy-move detection.

B. Modifying PatchMatch to deal with rotation and rescaling

The basic algorithm described above does not deal with scale changes and rotations, which are instead very common in copy-move image tampering. In a subsequent paper [33], however, the authors of PatchMatch generalized and extended their algorithm in several respects, including the ability to search across scales and rotation angles, going beyond mere translations, and to match patches based on arbitrary descriptors and distances, rather than just the Euclidean norm of original patches used in the basic version.

The solution proposed in [33] is straightforward: rather than analyzing only the 2d space spanned by the offset components (δ_1, δ_2), a 4d space is considered, comprising two further dimensions, scale, α, and rotation θ. All steps are then carried out as before, with obvious adjustments of minor significance, except for two main differences:

1) given the current values of α and θ, the target patch is suitably rescaled/rotated, interpolated and resampled, to be comparable with the original one;
2) likewise, in the propagation step, the candidate offsets are not just copied from the neighbors' offsets, but computed based on them and on the local transformation identified by α and θ.

The generalized algorithm preserves thus the simplicity of the original version, an appealing property. However, it presents some significant drawbacks. First of all, the computational complexity increases sharply, not only for the need to carry out patch interpolation but also for the increased number of iterations necessary to converge in a 4d search space. Moreover, the algorithm is not suitable to be used with compact features, like those described in Section IV, renouncing their potential for higher descriptive power and scale/rotation invariance, as well as the reduced complexity associated with their shorter length. Last, but not least, experiments on copy-move forgery detection [31] show generalized PatchMatch to be much less reliable than the basic version in the fundamental case of simple rigid translation, causing a significant loss in performance. Our conjecture is that with an higher-dimensionality optimization space, a large number of suboptimal matchings are available, which trap the algorithm into local minima. This problem should be solved by the random search phase but random sampling becomes too sparse in such a large space, and hence less effective.

Given these dismaying results, in terms of both accuracy and complexity, in [31] we proposed a different modification of the

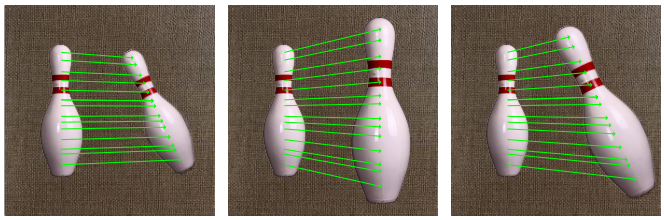


Fig. 2. Offsets vary linearly over the copy-moved region in the presence of rotation (left), rescaling (center), or both (right).

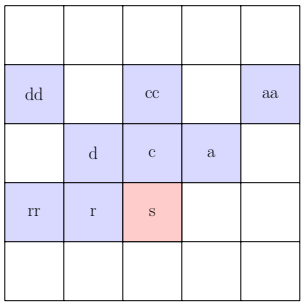


Fig. 3. Predictor geometry for direct scanning. Blue pixels are used to build various zero- and first-order predictors for the offset of pixel s (red).

basic PatchMatch algorithm, based on the use of scale/rotation invariant features. In fact, numerous such features have been proposed in recent years to describe image patches, solving in advance (as far as invariance holds) the problem of matching patches subject to such transformations.

Thanks to the use of invariant features, the basic algorithm needs be modified exclusively for what concerns the propagation phase, by changing the set of candidate offsets available for updating. In the original PatchMatch algorithm, the current offset for pixel s is compared with two other candidate offsets, $\delta(s^r)$ and $\delta(s^c)$, which are simply the causal zero-order predictions of $\delta(s)$ along image rows and columns, renamed here accordingly as $\widetilde{\delta}^{0r}(s)$ and $\widetilde{\delta}^{0c}(s)$, respectively. Of course, zero-order predictors are effective only in regions characterized by a constant offset, corresponding to copy-moves with rigid translations. Rotated, rescaled, and rotated-rescaled copy-moves, instead, are described by linearly varying offset fields, as shown in Fig. 2, in which case, first-order predictors can be expected to work correctly. Therefore we enlarge the set of candidates in (4) considering both zero-order and first-order predictors

$$\begin{aligned} &\widetilde{\delta}^{0x}(s) = \delta(s^x) \\ &\widetilde{\delta}^{1x}(s) = 2\delta(s^x) - \delta(s^{xx}) \\ &x \in \{r, d, c, a\} \end{aligned} \tag{7}$$

where s^{xx} is the pixel preceding s^x along direction x in the scanning order (see Fig. 3), and we include also the diagonal and antidiagonal directions, d and a, respectively, obtaining eventually the enlarged set of predicted offsets

$$\begin{aligned} \Delta^P(s) \quad = \quad & \{\delta(s), \widetilde{\delta}^{0r}(s), \widetilde{\delta}^{0d}(s), \widetilde{\delta}^{0c}(s), \widetilde{\delta}^{0a}(s), \\ & \widetilde{\delta}^{1r}(s), \widetilde{\delta}^{1d}(s), \widetilde{\delta}^{1c}(s), \widetilde{\delta}^{1a}(s)\} \end{aligned} \tag{8}$$

With this modification, whenever a correct offset field is found over a couple of neighboring pixels it will quickly propagate to the rest of the interested region within two iterations. Moreover, thanks to the zero-order predictor, and to the random search phase, it is not difficult to reach the initial condition which triggers the propagation. It is worth noting that the generalized PatchMatch algorithm [33] also uses a first order prediction which requires an affine transformation matrix. This matrix is not immediately available when using scale/rotation invariant features and therefore the linear term is estimated using the model of (7).

Some comments are in order. First of all, the complexity of the proposed version is very close to that of the basic one, as only the propagation phase is modified, and in a quite inexpensive way. Moreover, the opportunity to adopt compact features in place of pixel values grants a stronger efficiency gain. As for accuracy, using features that are scale and rotation invariant, we can match a very general class of copy-moves. It could be pointed out that most features proposed in the literature are only rotation-invariant. However, PatchMatch exhibits a remarkable robustness w.r.t. limited scale changes [31], thanks to the random search phase. Hence, one can explore the scale dimension through a brute-force approach in a limited number of steps, a non-elegant solution, viable only thanks to PatchMatch speed, which proved to be very effective in practical applications [34], [35]. Finally, it should be pointed out that, on the wake of PatchMatch, other dense-field techniques have been proposed recently [36], [37], providing further improvements in search efficiency by replacing the random search phase with some smarter initialization, based on fast approximate NN search techniques. Likewise, there is intense activity on dense image matching with non-linear offset models, e.g. [38]. This is a very active field of research, and we are confident that some of these ideas may be included in future versions of our algorithm.

III. Post-processing based on dense linear fitting

Ideally, the offset field obtained through feature matching should be mostly chaotic except for some large smooth regions with linear behavior in correspondence of cloned objects. In practice, because of noise, compression, geometric deformations, illumination changes, look-alike regions, the computed offset field rarely follows this model and some post-processing is necessary to

1) regularize the offset field to increase the probability of detecting actual copy-moves;
2) add some suitable constraints to reduce the probability of false alarms.

The first problem is especially challenging, and previous papers tackled it through sophisticated and relatively slow methods, such as the well-known RANSAC [5] or SATS [39]. In our case, however, thanks to the implicit filtering enacted by PatchMatch, the offset field is regular enough to consider a simpler approach, based on dense linear fitting (DLF).

We want to fit, in a suitable N-pixel neighborhood of s, the true offset field $\delta(s)$ through a linear (more precisely, affine) model

$$\widehat{\delta}(s_i) = As_i, \quad i = 1, \ldots, N \tag{9}$$

with the parameters of the transformation, A, set so as to minimize the sum of squared errors w.r.t. the true data

$$\epsilon^2(s) = \sum_{i=1}^{N} \|\delta(s_i) - \widehat{\delta}(s_i)\|^2 \tag{10}$$

Although the offset field is bi-dimensional, the model parameters can be optimized independently for each of the two components, so in the following lines, in order to simplify notations, we will treat $\delta(s)$ as a single component field.

With this understanding, we can write the problem as

$$a^{\text{opt}} = \arg\min_{a} \|\delta - Sa\|^2 \tag{11}$$

where $\delta = [\delta(s_1), \delta(s_2), \ldots, \delta(s_N)]^T$ is the vector of offsets output by the matching phase, $a = [a_0, a_1, a_2]^T$ is the vector of parameters that identifies the affine transform, and S is the $N \times 3$ matrix of the homogeneous coordinates of all pixels in the neighborhood

$$S = \begin{bmatrix} 1 & s_{11} & s_{12} \\ 1 & s_{21} & s_{22} \\ & \vdots & \\ 1 & s_{N1} & s_{N2} \end{bmatrix} \tag{12}$$

so that,

$$\widehat{\delta}(s_i) = a_0 + a_1 s_{i1} + a_2 s_{i2}, \quad i = 1, \ldots, N \tag{13}$$

This is a well known multiple linear regression problem [40], with solution

$$a^{\text{opt}} = (S^T S)^{-1} S^T \delta \tag{14}$$

The corresponding sum of squared errors (SSE) is therefore

$$\begin{aligned} \epsilon^2(s) &= \|\delta - S(S^T S)^{-1} S^T \delta\|^2 \\ &= \|(I - H)\delta\|^2 \\ &= \delta^T (I - H)\delta \end{aligned} \tag{15}$$

where we have exploited the fact that $H = S(S^T S)^{-1} S^T$ is symmetric and idempotent ($HH = H$). If the coordinates in (12) are taken relative to s, and the neighborhood has constant shape, the matrix H does not depend on s, hence computing the SSE reduces to evaluating the quadratic form (15) for the two offset components. However, the processing cost can be further reduced by decomposing the rank-3 matrix H as

$$H = QQ^T, \qquad Q = [q_1, q_2, q_3] \tag{16}$$

where q_j is a column vector of length N, and hence

$$\epsilon^2(s) = (\delta^T \delta) - (\delta^T q_1)^2 - (\delta^T q_2)^2 - (\delta^T q_3)^2 \tag{17}$$

computed through a few filtering operations and some products.

We can now outline the complete post-processing procedure, which comprises the following steps:

1) median filtering on a circular window of radius ρ_M;
2) computation of the fitting error, $\epsilon^2(s)$, w.r.t. a least-squares linear model over a circular neighborhood of radius ρ_N;
3) thresholding of $\epsilon^2(s)$ at level T_ϵ^2;
4) removal of couples of regions closer than T_{D2} pixels;
5) removal of regions smaller than T_S pixels;
6) mirroring of detected regions;
7) morphological dilation with a circular structuring element of radius $\rho_D = \rho_M + \rho_N$.

As already said in Section II, a locally linear model is certainly appropriate for the copy-moves considered in this work, but minimum mean-square error (MMSE) fitting is very sensitive to outliers. Therefore, before the DLF, we carry out a median filtering process, which removes outliers, but leaves the signal unaltered where it has a linear behavior. Alternatively, one could resort to iteratively-reweighted least squares [41]. In step 3, the image is segmented to single out candidate copy-moved regions. Here, to keep complexity limited, a simple thresholding is considered, but plenty of methods are available to improve this process. Steps 4 and 5 are meant to remove matchings obtained by chance. Spurious matchings abound in natural images because of repeated patterns or uniform background. In the first case, however, similar details are typically quite small, and can be removed based on the size constraint. In the second case, we exploit the fact that background regions are generally not as uniform as they appear. Gradual luminance changes imply that, in a uniform background region, patches similar to the target are also close to it, and hence the additional constraint on distance eliminates these regions. It goes by itself that an image portraying multiple replicas of the same object can still give rise to false alarms, especially if scale and rotation invariant features are used. Once we decide that pixel s belongs to a copy-move region, we also mark pixel $s + \delta(s)$ as copy-moved in step 6. Finally, considering that both median filtering and model fitting tend to erode the support of the copy-moved regions, in step 7 we restore them through a complementary dilation.

The various steps of the procedure are illustrated In Fig. 4 on a real-world example.

IV. Feature Extraction

A large number of features have been proposed in recent years for the purpose of copy-move detection, and many of them have been considered in the extensive experimental comparison carried out in [3]. Here, we will focus on features based on the family of Circular Harmonic Transforms (CHT) [42] which possess desirable invariance properties.

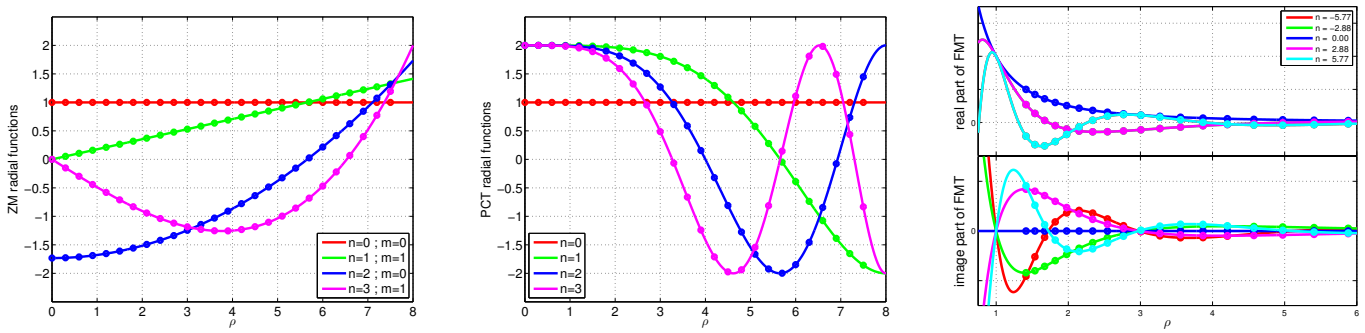
Let $I(x, y)$ be a scalar image defined on a continuous space, $(x, y) \in R^2$, and let $I(\rho, \theta)$ be its representation in polar coordinates, with $\rho \in [0, \infty]$ and $\theta \in [0, 2\pi]$. The CHT coefficients are evaluated by projecting the image over the basis functions $K_{n,m}(\rho, \theta)$ of the transform

$$F_I(n, m) = \int_0^{2\pi} \int_0^{\infty} I(\rho, \theta)\, K_{n,m}^*(\rho, \theta) \rho \, d\rho \, d\theta \tag{18}$$

The basis functions have the form

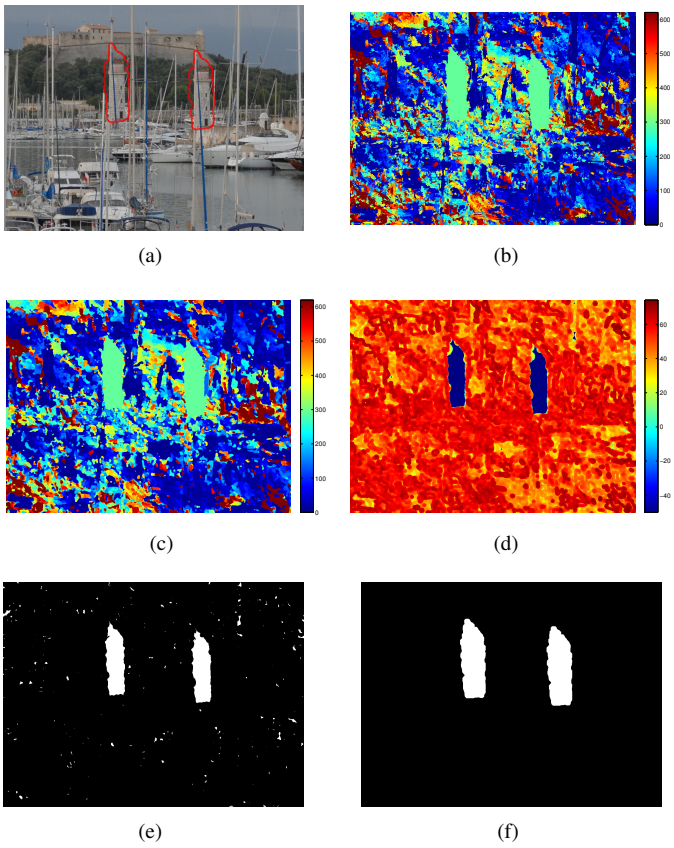
$$K_{n,m}(\rho, \theta) = R_{n,m}(\rho) \frac{1}{\sqrt{2\pi}} e^{jm\theta} \tag{19}$$

that is, they are obtained as the product of a radial profile $R_{n,m}(\rho)$ and a circular harmonic. Therefore (18) can be



(a) (b) (c)

Fig. 5. Radial profiles of some CHTs: Zernike (left), PCT (center), FMT (right).



(a) (b) (c) (d) (e) (f)

Fig. 4. Post-processing steps: (a) original forged image, (b) magnitude of offsets, (c) median filtering, (d) fitting error $\epsilon^2(s)$ (dB), (e) thresholding of $\epsilon^2(s)$, (f) final mask.

rewritten as

$$F_I(n,m) = \int_0^\infty \rho R^*_{n,m}(\rho) \times \left[\frac{1}{\sqrt{2\pi}} \int_0^{2\pi} I(\rho,\theta) e^{-jm\theta} d\theta \right] d\rho \tag{20}$$

The integral in square brackets, let us call it $\widehat{I}(\rho)$, is the Fourier series of $I(\rho,\theta)$ along the angle coordinate. Therefore, a rotation of θ_0 radians in I contributes just a phase term $e^{jm\theta_0}$ in $\widehat{I}$, which disappears if one takes the magnitude of the coefficients, thereby obtaining rotation invariance.

The various CHTs differ in the radial profile. In this work we consider three choices, the Zernike Moments (ZM) [43], the Polar Cosine Transform (PCT) [44], and the Fourier-Mellin Transform (FMT) [45]. Zernike radial functions are defined as

$$R_{n,m}(\rho) = \sum_{h=0}^{(n-|m|)/2} C_{n,m,h} \rho^{n-2h} \tag{21}$$

for $\rho \in [0,1]$, with $C_{n,m,h}$ suitable coefficients that ensure orthonormality of the basis functions. Fig. 5(a) shows some of these functions, chosen among those with lowest order. In the PCT, the radial functions are just cosines with argument ρ^2,

$$R_n(\rho) = C_n \cos(\pi n \rho^2) \tag{22}$$

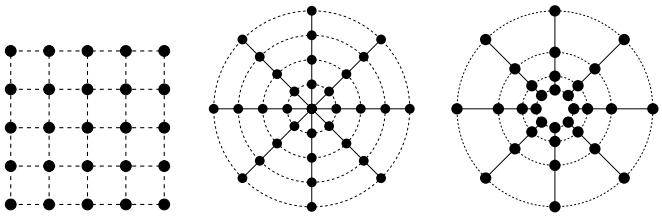
limited again to $\rho \in [0,1]$, with normalizing coefficients C_n, some of which are shown in Fig. 5(b). In the FMT, instead, they are defined as

$$R_\nu(\rho) = \frac{1}{\rho^2} e^{j\nu \ln(\rho)} \tag{23}$$

Notice however that, in this case, the functions are non-zero for all $\rho \geq 0$, they diverge at the origin, and the parameter ν is continuous-valued. With this choice, the integral (20) becomes just the Fourier transform of $\widehat{I}(\rho)$ after a coordinate remapping, while the whole FMT can be regarded as the bi-dimensional Fourier transform of I in log-polar coordinates. As a consequence, a scale change in I contributes only a phase in the FMT coefficients, which disappears after taking the absolute value, granting also scale invariance.

Now, we have to translate these theoretical definitions into practical finite-length features which characterize locally the image. These must be computed on the available data, sampled on a discrete grid, preserving the invariance properties. To this aim, we have to select a finite number of (n,m) couples, define a suitable patch size and, for each pixel s, compute the $F_{I(s)}(n,m)$ coefficients, by approximating the integral of (18) with a summations over the patch centered on s. Eventually, the feature $f(s)$ will be the collection of the magnitudes of these coefficients.

The patch size must guarantee a good compromise between discrimination and robustness. Patches too small might not catch the local image behavior, while if too large they might loose resolution and lead to false alarms. Likewise, features should not be unnecessarily long, to avoid slowing down all processing steps, but still expressive enough to allow correct



(a) (b) (c)

Fig. 6. Examples of rectangular (a), polar (b) and log-polar (c) sampling grids.

matches. We will not indulge in describing the preliminary experiments carried out to set these quantities, selected values are reported in Tab. I. Patches of 16×16 or 24×24 pixels are used, with features of length 12 for Zernike, 10 for PCT, and 25 for FMT, corresponding always to the lower order[2] basis functions.

Let us focus, instead, on the approximation of the integral (18). A straightforward solution is to resample the basis functions $K_{n,m}(\rho,\theta)$ on the grid points (x,y) of the analysis patch, W, where the image is defined (see Fig. 6(a)), computing therefore

$$F'_I(n,m) = \sum_{(x,y)\in W} I(x,y)\, K^*_{n,m}(\rho(x,y),\theta(x,y)) \qquad (24)$$

with $\rho(x,y) = \sqrt{x^2+y^2}$ and $\theta(x,y) = \pm\arctan(y/x)$. However, an equally viable solution is to resample the image on polar (or logpolar) coordinates (see Fig. 6(b)-(c)), and compute

$$F''_I(n,m) = \sum_{(\rho,\theta)\in W} I(x(\rho,\theta),y(\rho,\theta))\, K^*_{n,m}(\rho,\theta)\rho^i \qquad (25)$$

with $i = 1$ for the polar grid and $i = 2$ for the logpolar one. This seemingly minor difference has non-negligible consequences on performance, in particular on rotation invariance [46]. In fact, polar sampling guarantees perfect invariance for rotation angles multiple of the sampling step $\Delta\theta$, and a good approximation of it in all other cases, provided $\Delta\theta$ is not too large. On the contrary, with rotation angles close to $\pi/4 \pm k\pi/2$, features computed on the Cartesian grid can change significantly, undermining the invariance property, as also shown in [47] where an accurate analysis of errors induced by sampling is carried out.

In addition, the two solutions have the same computational efficiency, since, given ρ and θ, the interpolated values $I(\rho,\theta)$ in (25) are computed from available data points with fixed weights, falling back again to a filtering of the form (24), only with different weights. We will therefore resort to the polar sampling for both Zernike and PCT features, but keep also the Cartesian sampling as reference. For FMT, instead, we will use a log-polar sampling, aiming at scale invariance. However, we are forced to exclude points too close to the origin [48], that is to the central pixel s, where the radial functions diverge.

[2]For FMT, $\nu = 2n\pi/\log(\rho_{\max}/\rho_{min})$, for $n = 0, \pm1, \pm2$.

Fig. 7. Three forged images with different levels of activity from the FAU database. From top: smooth, rough, structured.

We note explicitly that CHT-based features have been already used for forgery detection. Zernike moments, for example, have been adopted in [49], [3], [20], with Cartesian sampling, providing promising results. Likewise, the PCT has been investigated in [21], again with Cartesian sampling. As for FMT-based features they have been also already considered for forgery detection [23], but with suboptimal results, as reported in [3]. However, the implementation proposed in [23], inspired by [50], includes further processing steps that disrupt the invariance properties, so useful for robust copy-move detection. Similarly, in [25], the features are formed by taking some cross-spectra, rather than the magnitude of the coefficients themselves.

V. Experimental results

In this section, we present the results of a number of experiments carried out in order to fine tune the proposed technique and assess its performance w.r.t. the state of the art. In order to guarantee reproducibility of results, our code is available online[3], and experiments are carried out on two databases also available online. The database used in [3], which we will call FAU[4] from now on, comprises 48 images with realistic copy-move forgeries, some examples of which are shown in Fig. 7, classified as smooth, rough or structured. These images are quite large, with typical size 3000×2400 pixels, with tampered areas covering about 6% of each image, on average. We prepared a further database[5] composed by 80 images, again with realistic copy-move forgeries, some of which are shown in Fig. 8. All these images have size

[3]http://www.grip.unina.it
[4]http://www5.cs.fau.de/
[5]http://www.grip.unina.it

768×1024 pixels, while the forgeries have arbitrary shapes, aimed at obtaining visually satisfactory results, with size going from about 4000 pixel (less than 1% of the image) to about 50000 pixels. In adding this new database, called GRIP from now on, we wanted to enrich the experimental set available to the community, and include also forgeries of relatively small size. However, we were also motivated by the practical need to run in a reasonable time the large number of experiments needed to fine-tune and validate the proposed technique in various situations of interest.

Results are provided both at pixel level and image level. To assess synthetically the image-level performance we use the F-measure, defined as

$$\mathrm{F} = \frac{2\,\mathrm{TP}}{2\,\mathrm{TP} + \mathrm{FN} + \mathrm{FP}} \tag{26}$$

where TP (true positive), FN (false negative), and FP (false positive) count, respectively, the number of detected forged images, undetected forged images, and wrongly detected genuine images. Similar definitions are used at pixel-level for each image to obtain, after averaging over all images, the pixel-level F-measure. At image-level we measure, therefore, only the ability to correctly recognize an image as forged or genuine, while the pixel-level measure accounts also for localization accuracy. At pixel-level we exclude pixels at the boundary between forgery and background from computation. The transparency is set to an intermediate value between 0 and 1 to avoid artifacts. Processing time is another key performance parameter, since reliable copy-move detectors are known to be rather slow, a non-negligible problem with images becoming larger and larger as imaging technology advances. We will therefore report also the average CPU time per image, measured on a computer with a 2GHz Intel Xeon processor, operating in single-thread modality.

Next subsection is devoted to analyze the proposed technique, while the subsequent one compares performance with the state of the art.

A. Fine tuning the PatchMatch-based copy-move detector

In the proposed technique there are a number of design choices and numerical parameters to be defined. After some preliminary experiments, we set all parameters to the reasonable and non-critical values reported in Tab. I. The number of PatchMatch iterations, N_{it}, however, deserves a deeper analysis, since it can have significant impact on the overall performance. We pointed out already that a good offset field can be obtained after a small number of iterations, but how small is "small", exactly, and what are the effects of this choice on performance? To gain insight into this point, we ran a series of experiments on the GRIP database, considering various situations of interest: noise addition, JPEG compression, resizing and rotation. In these experiments, we used the Zernike features with polar sampling, with the default parameters of Tab. I. Results on accuracy are reported in Fig. 9, and show that the performance is already very good with as little as 4 iterations. Rotated forgeries are the only exception, where increasing N_{it} to 8 or even 16 guarantees some improvements, up to 0.1, for large angles. In Tab. II we report instead the

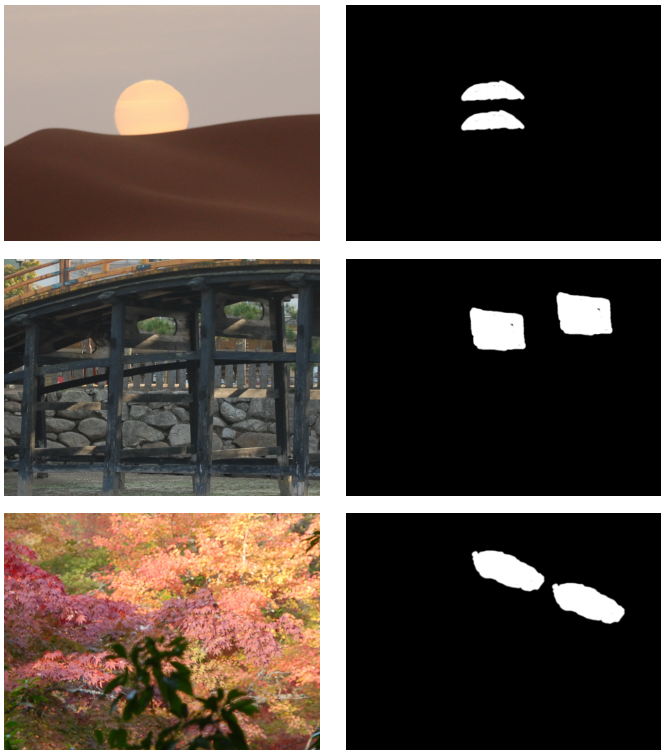


Fig. 8. Three forged images from the GRIP database with different levels of activity. From top: smooth, mixed, textured.

param.	value	phase	meaning
$\|W\|_C$	16×16	F	(x, y) samples in Cartesian grid
$\|W\|_P$	26×32	F	(ρ, θ) s.s in polar grid ($\rho \leq 8$)
$\|W\|_{LP}$	26×32	F	(ρ, θ) s.s in logpolar grid ($\rho \leq 12$)
$\|f\|_{\mathrm{Z}}$	12	F	length of Zernike features
$\|f\|_{\mathrm{PCT}}$	10	F	length of PCT features
$\|f\|_{\mathrm{FMT}}$	25	F	length of FMT features
N_{it}	4-16	M	# PatchMatch iterations
T_{D1}	8	M	minimum length of offsets
T_{D2}	50	PP	minimum distance between clones
T_ϵ^2	300	PP	threshold on DLF error
T_S	1200	PP	minimum size of clones
ρ_M	4	PP	radius of median filter
ρ_N	6	PP	radius of DLF patch
ρ_D	10	PP	radius for morphological dilation

TABLE I
RELEVANT PARAMETERS FOR THE VARIOUS PHASES OF THE PM-BASED TECHNIQUE AND PROPOSED SETTING.

CPU time. This is constant for feature extraction and nearly so for post-processing. It grows almost linearly with N_{it} for the matching phase, which weights heavily on the overall efficiency of the algorithm. With $N_{it} = 4$, the proposed algorithm takes about 11 seconds/image, on the average, a time that almost doubles for $N_{it} = 16$. Larger values make no sense as they have no effect on accuracy. Therefore, the user can be interested in several profiles, from FAST ($N_{it} = 4$) with an overall processing time competitive with that of keypoint-

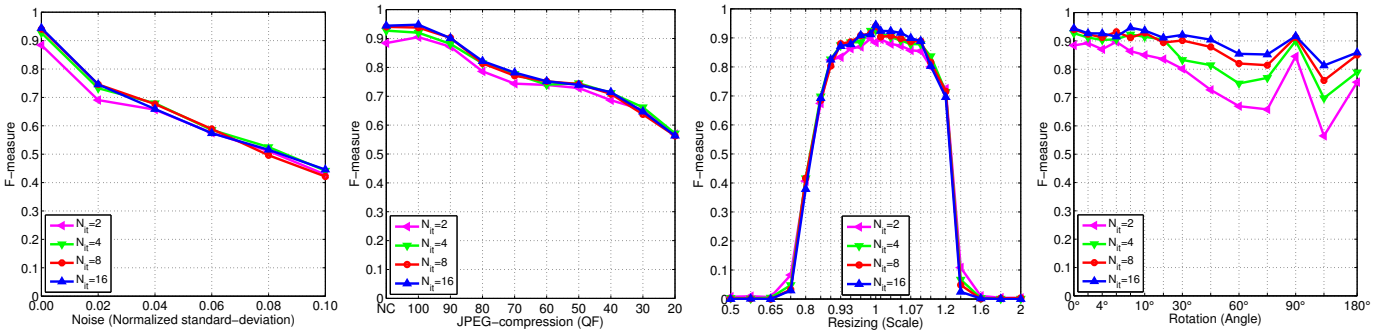


Fig. 9. Pixel-level F-measure curves for the proposed PM-based technique (Zernike-polar feature) for different values of N_{it}

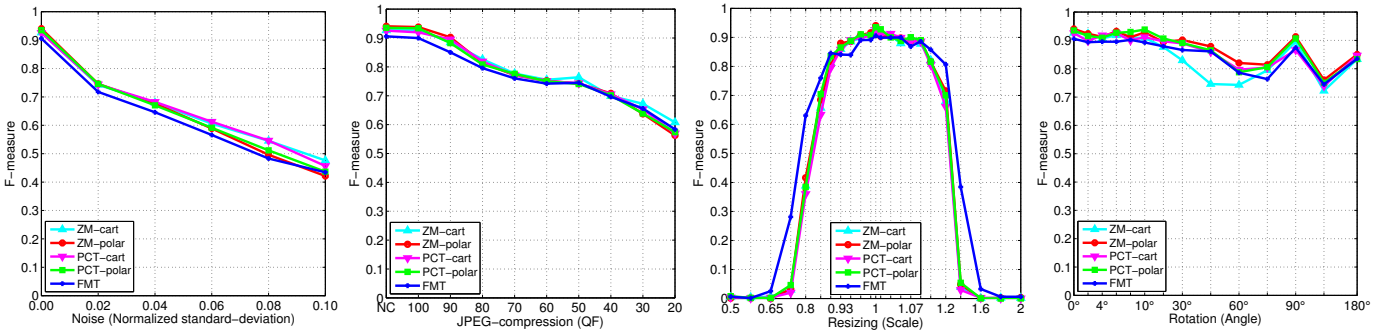


Fig. 10. Pixel-level F-measure curves for the proposed PM-based technique ($N_{it} = 8$) with different features.

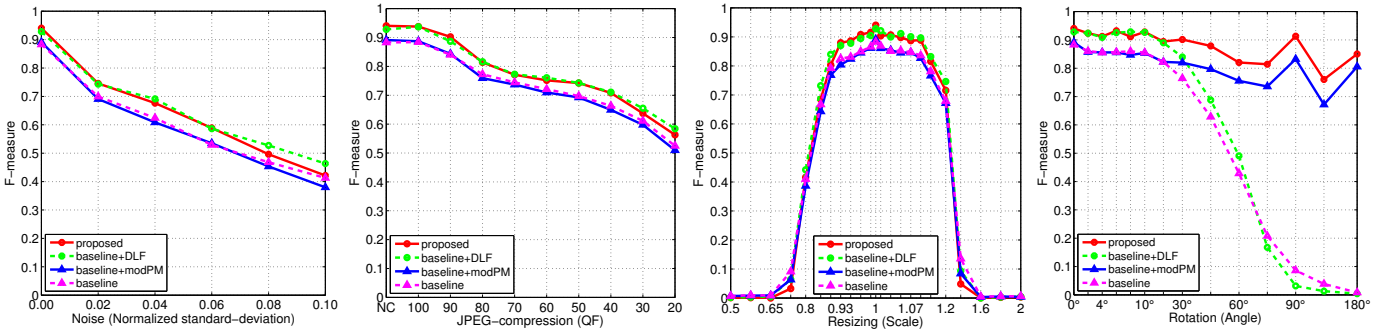


Fig. 11. Pixel-level F-measure curves for the proposed technique, the baseline reference, and intermediate versions.

N_{it}	Feat. Ext.	Matching	Post-Proc.	Total
2	2.36 (0.48)	3.19 (0.32)	1.93 (0.18)	7.48 (0.66)
4		6.54 (0.96)	2.12 (0.30)	11.02 (1.49)
8		11.96 (1.68)	2.10 (0.25)	16.42 (2.00)
16		16.81 (1.51)	1.77 (0.31)	20.92 (1.85)

TABLE II
CPU-TIME PERFORMANCE (SECONDS/IMAGE) VS N_{it} USING ZERNIKE-POLAR FEATURES; IN PARENTHESES THE STANDARD DEVIATIONS.

Feature	Feat. Ext.	Matching	Post-Proc.	Total
ZM-cart	2.55 (0.52)	12.00 (1.70)	2.08 (0.25)	16.63 (2.02)
ZM-polar	2.36 (0.48)	11.96 (1.68)	2.10 (0.25)	16.42 (2.00)
PCT-cart	1.79 (0.29)	10.78 (1.48)	2.30 (0.32)	14.86 (1.66)
PCT-polar	1.74 (0.24)	10.79 (1.25)	2.16 (0.29)	14.69 (1.40)
FMT	9.82 (1.47)	15.23 (1.72)	1.80 (0.16)	26.86 (2.83)

TABLE III
CPU-TIME PERFORMANCE (SECONDS/IMAGE) VS FEATURE USING $N_{it} = 8$; IN PARENTHESES THE STANDARD DEVIATIONS.

based techniques, but a much better accuracy, to ACCURATE ($N_{it} = 16$), which guarantees the best performance at the cost of a longer, but still reasonable, CPU time.

The other major choice available to the user concerns the type of feature to use. Again, for each feature we set in advance the main parameters as reported in Tab. I. The pixel-level curves shown in Fig. 10, obtained with $N_{it} = 8$, are quite close to one another. With all features, the proposed technique behaves very well on rigid copy-moves, with an F-measure going from 0.90 for FMT to 0.94 for Zernike-polar. Moreover, the performance degrades in a similar manner for all features with increasing noise, compression ratio, rescaling factor, and rotation angle. Some minor differences can be observed for FMT, slightly better than the others for moderate

Version	Feat. Ext.	Matching	Post-Proc.	Total
baseline	2.06 (0.21)	9.00 (0.84)	82.68 (59.61)	93.74 (59.51)
+modPM	2.36 (0.48)	11.96 (1.68)	91.77 (57.23)	106.09 (56.95)
+DLF	2.06 (0.21)	9.00 (0.84)	2.01 (0.27)	13.07 (0.99)
proposed	2.36 (0.48)	11.96 (1.68)	2.10 (0.25)	16.42 (2.00)

TABLE IV
CPU-TIME PERFORMANCE (SECONDS/IMAGE) VS VERSION; IN PARENTHESES THE STANDARD DEVIATIONS.

rescaling factors, and for Zernike-Cartesian, slightly worse at large-angle rotations.

With these results, processing time becomes again a key decision element. Tab. III shows the average CPU-time for the selected features with the default parameters of Tab. I. It results that only the FMT feature presents a significantly different (higher) processing time, due to the larger patch used to compute the feature, and the longer features themselves.

Before comparing results with the state of the art, we present some experiments to assess the impact of each of the proposed improvements. In particular, keeping fixed the parameters selected before and using the Zernike-polar features when applicable, we consider a baseline reference technique with basic PatchMatch and SATS, two intermediate versions where only the matching phase or the post-processing are improved, switching to our modified PatchMatch or to the proposed post-processing based on dense linear fitting, respectively, and then the proposed technique including both improvements. Results are reported in Fig. 11 and Tab. IV. All versions provide a similar accuracy performance in the presence of noise, compression and resizing, except for a small but consistent improvement observed when the DLF-based post-processing is used. In the presence of rotation, instead, for angles beyond 15 degrees the performance drops off sharply for the versions using basic PatchMatch, because it does not use a first-order prediction, while it remains almost constant for those using modified PatchMatch.

As for computational efficiency, the basic and modified versions of PatchMatch are almost equivalent, while SATS is much slower than DLF, and responsible for most of the overall running time, when used. SATS complexity descends from its iterative nature [39]: given a few reliable close points, it estimates an affine transform explaining their offsets, and gradually enriches the set including only points obeying the same transform. Therefore, outliers are automatically rejected, improving robustness at the cost of higher CPU time. With PatchMatch, however, the offset field is already quite regular, allowing for the use of the much faster DLF without harm.

B. Comparison with the state of the art

To position the proposed approach w.r.t. the current state of the art, results are compared with those of several promising techniques recently proposed in the literature, most of them designed to deal also with rotation and rescaling. In particular we consider Bravo2011 [27], Christlein2012 [3][6], Amerini2013 [7], Ryu2013 [20], and Cozzolino2014 [31], the latter being a previous version of the technique proposed here, with Zernike-Cartesian feature, SATS postprocessing, and other minor differences. A comparison with the other techniques reviewed in [3] can be established using (with caution) the transitive property. Unfortunately, Ryu2013 turned out to be exceedingly slow on smooth images (about 15 hours on the 768×1024 image on the top of Fig. 8), so it does not appear in the following results, and we will design a separate experiment to compare with it. For the proposed PatchMatch-based technique (shortnamed PM from now on) we set $N_{it} = 8$ once and for all, but test all proposed features.

Technique	F-image	F-pixel	CPU
Christlein2012	93.20	93.52	4377.60
Bravo2011	96.97	90.52	2149.91
Amerini2013	74.07	50.11	156.88
Cozzolino2014	94.85	89.77	2366.57
PM-ZM-cart	93.07	93.11	287.50
PM-ZM-polar	94.95	93.72	244.67
PM-PCT-cart	94.00	93.17	281.37
PM-PCT-polar	95.92	93.62	293.84
PM-FMT	92.00	89.46	424.26

TABLE V
IMAGE-LEVEL AND PIXEL-LEVEL F-MEASURE AND AVERAGE CPU-TIME PER IMAGE

for rigid copy-moves on the FAU database.

Technique	F-image	F-pixel	CPU
Christlein2012	98.16	87.44	53.91
Bravo2011	95.81	84.44	102.78
Amerini2013	67.72	44.41	3.71
Cozzolino2014	94.67	88.67	54.74
PM-ZM-cart	92.94	92.66	16.63
PM-ZM-polar	94.05	94.06	16.42
PM-PCT-cart	94.61	92.55	14.86
PM-PCT-polar	94.05	93.51	14.69
PM-FMT	91.33	90.56	26.86

TABLE VI
IMAGE-LEVEL AND PIXEL-LEVEL F-MEASURE AND AVERAGE CPU-TIME PER IMAGE FOR RIGID COPY-MOVES ON THE GRIP DATABASE.

In Tab. V and Tab. VI we report, for the FAU and the GRIP databases, respectively, the image-level and pixel-level F-measure, together with the total CPU time, observed for rigid translation. In the top part of the tables we group reference techniques, and in the bottom part the various versions of the proposed one. Taking into account the obvious differences, results are very well aligned on the two databases. In particular, the proposed technique performs best with polar features, both Zernike and PCT, being competitive with the other dense-field

[6]We refer to the technique with Zernike features, kd-tree matching, and SATS post-processing.

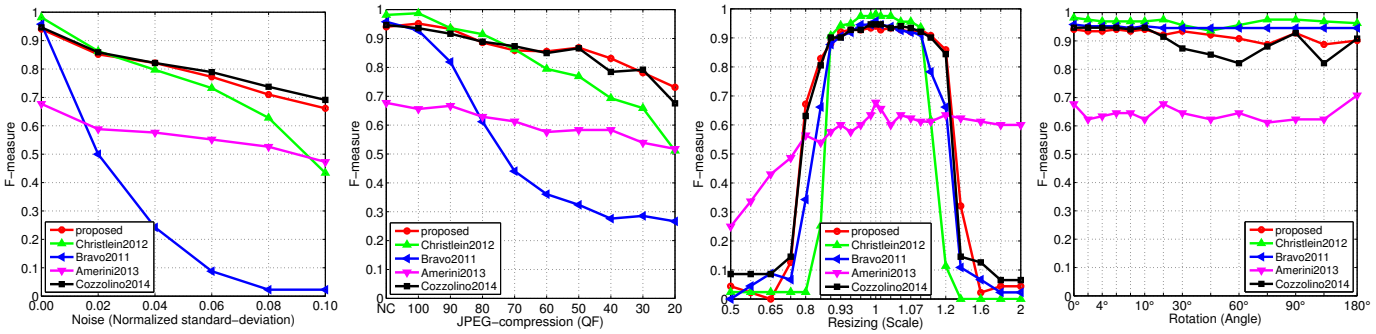


Fig. 12. Image-level F-measure curves for the proposed (PM-ZM-polar) and reference techniques.

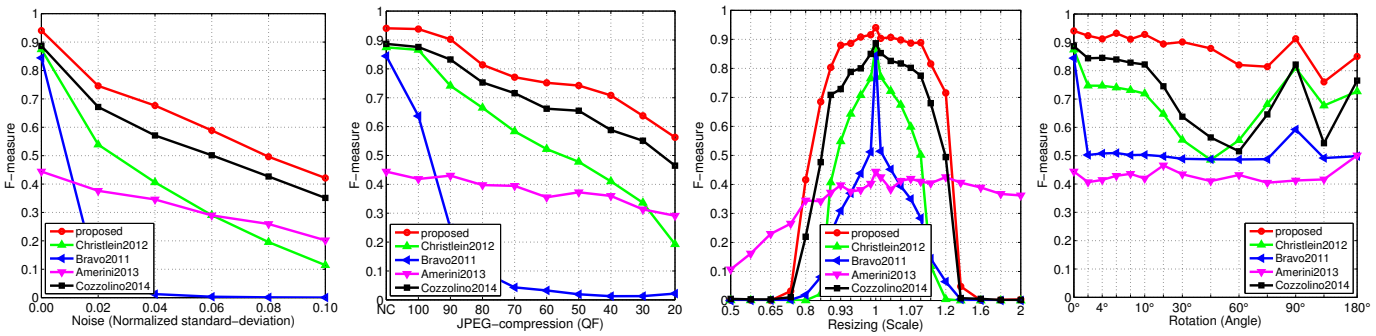


Fig. 13. Pixel-level F-measure curves for the proposed (PM-ZM-polar) and reference techniques.

techniques at image level, and generally better than them at pixel level. As expected, the only keypoint-based technique, Amerini2013, provides a much worse detection performance, but also the smallest processing time. Among dense-field techniques, PM is by far the most efficient, with all features. With Zernike-polar features, in particular, it is on average at least 3 times faster than the dense-field references on the GRIP database, and at least 9 times faster than them on the FAU database. The different speed-up factors suggest that, while the complexity of the PatchMatch-based technique scales almost linearly with the image size, this might not hold for the other dense-field techniques.

We now test robustness against noise addition, compression, rotation and resizing, showing the image-level and pixel-level F-measure curves in Figs. 12 and 13, respectively. For the sake of clarity, only the Zernike-polar feature is considered for the proposed technique. At image level, as already reported in the tables, the dense-field Bravo2011 and Christlein2012 exhibit the best performance in the ideal case, a performance which degrades rapidly, however, with increasing levels of noise and JPEG compression. At pixel-level, instead, the proposed PatchMatch-based detector outperforms consistently all the references, with a performance gain that becomes very significant in the presence of intense noise and large compression factors, as well as for moderate scale changes and critical rotation angles around 45^o and 135^o.

The SIFT-based Amerini2013 deserves a separate discussion. In fact, thanks to the use of keypoints, it provides the most stable performance across all conditions, including large-scale resizing, where all considered dense-field techniques fail. Unfortunately, its overall performance is impaired by the inability to discover copy-moves in smooth areas, lacking the keypoints, and localization seems to be quite imprecise. Given these complementary properties, a suitable fusion of keypoint and dense-field approaches may be expected to provide good results.

Let us now describe the results of an *ad hoc* experiment designed to include the promising Ryu2013 technique, based on LSH and RANSAC, in the comparative analysis. To this end, we built a database comprising 40 textured images, a case in which Ryu2013 presents an acceptable CPU-time, with size 512×512-pixel, and random square forgeries. Fig. 14 shows the image-level F-measure curves. On this set of images, the proposed technique and Christlein2012 provide almost perfect detection in all cases, except for large rescaling and intense noise. Also the performance of Ryu2013 is quite stable, but starting from a worse result on rigid copy-moves. This is due to the large number of false alarms observed in genuine images. In fact, at pixel level, when only forged images are considered, Ryu2013 performs as well as the proposed technique as clear from Fig. 15. However, it spends on average 159.05 seconds to process these images, much more than the 21.27 and 4.70 seconds necessary, respectively, for Christlein2012 and for the proposed technique.

We conclude this section by showing some examples of challenging copy-moves from the GRIP database together with the color-coded detection mask output by Christlein2012, Amerini2013, and by the PatchMatch-based detector. Christlein2012 detects all forgeries, but the masks are rather inaccurate. Moreover, in the second image there are a few false alarms due to a flat background. Amerini2013 works quite well on high entropy images, but misses altogether the copy-moves in the first and last images, both corresponding to mostly or completely smooth areas. On the contrary, the pro-

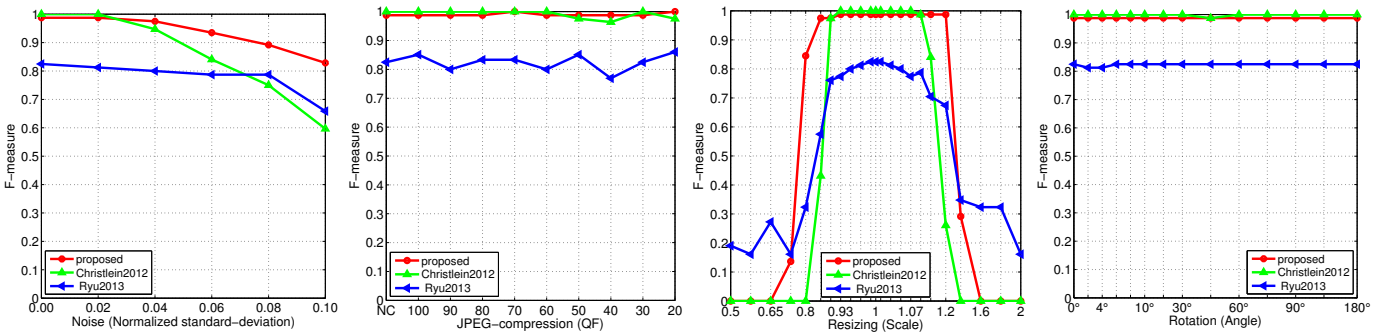


Fig. 14. Image-level F-measure curves for some selected techniques on textured images.

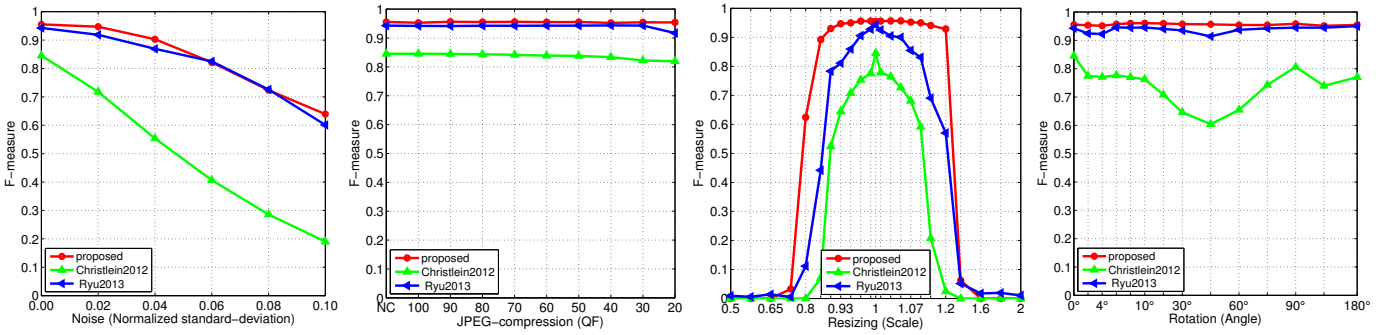


Fig. 15. Pixel-level F-measure curves for some selected techniques on textured images.

posed technique detects all forgeries with remarkably accurate masks and without false alarms.

VI. Conclusions

Copy-move forgeries are extremely common, and can be carried out easily and accurately. Detecting them, on the contrary, can be quite challenging, especially if they are of the occlusive type, with pieces of background copied elsewhere to hide some subjects of interest. In these cases, keypoint-based techniques are mostly ineffective, as they totally neglect low-entropy background areas. Dense-field techniques, on the other hand, tend to be very slow because of the feature matching phase, a problem which becomes worse and worse as the image size increases.

The technique proposed here is a first step towards fast and accurate copy-move detection. We use the PatchMatch algorithm to compute efficiently a high-quality approximate nearest neighbor field for the whole image. Given this major achievement, we then reduce the overall complexity by implementing also a fast post-processing procedure based on dense linear fitting. Moreover, by resorting to state-of-the-art invariant features, and a suitably modified version of PatchMatch, we achieve also a good robustness to various type of geometrical distortions.

Experimental results are quite satisfactory and suggest that the proposed technique provide state-of-the-art detection performance, with significant improvements in terms of localization accuracy and speed. Nonetheless, there is much room for further improvements. PatchMatch is certainly effective, but new fast matching algorithms are proposed by the day, and further progresses can be easily foreseen. Likewise, our post-processing, though effective, can be certainly further refined. Even so, with much larger images, which will probably become customary in the near future, all these tools may soon become ineffective, and multiresolution analysis is probably a more solid path for future research.

Turning to accuracy, a major goal is to achieve higher robustness to resizing, and to include also other forms of geometric distortion. Under this point of view, the discrete-domain FMT features fail to guarantee the invariance properties promised by their continuous-domain counterparts. Better implementations of FMT, or completely new and more robust features, can be probably proposed. Also in this case, a more interesting long-term research path is represented by a suitable fusion of tools at feature-level or detector-level.

Acknowledgments

The authors are grateful to Mr. Luca D'Amiano who prepared the forged images of the GRIP database.

References

[1] A. Piva, "An overview on image forensics," *ISNR Signal Processing*, pp. 1–22, October 2012.

[2] O.M. Al-Qershi and B.E. Khoo, "Passive detection of copy-move forgery in digital images: State-of-the-art," *Forensic Science International*, vol. 231, pp. 284–295, 2013.

[3] V. Christlein, C. Riess, J. Jordan, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pp. 1841–1854, 2012.

[4] X. Pan and S. Lyu, "Region duplication detection using image feature matching," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 857–867, December 2010.

[5] 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*, vol. 24, no. 6, pp. 381–395, June 1981.

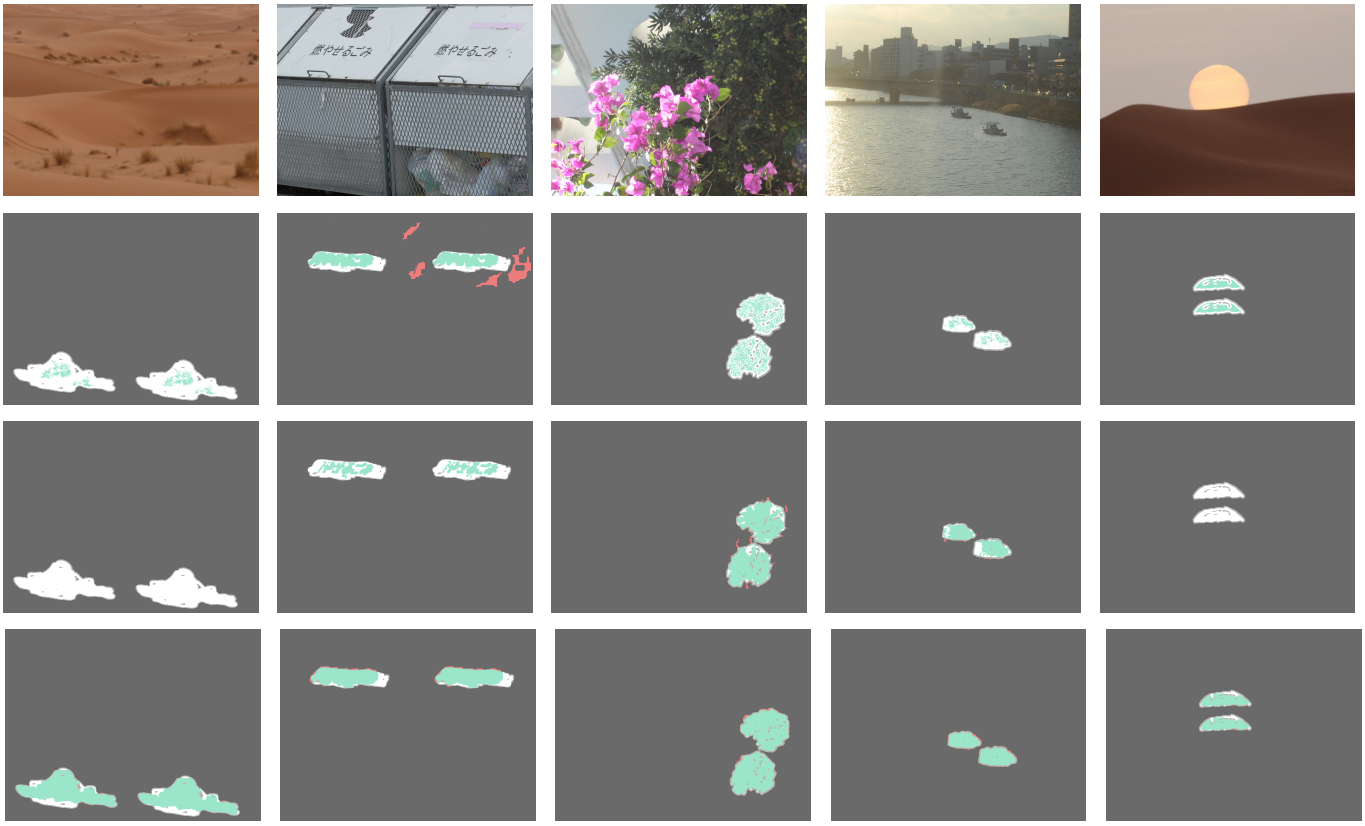


Fig. 16. Forgery detection masks for some images of the GRIP database. From top to bottom: forged images, masks output by Christlein2012, masks output by Amerinin2013, masks output by the proposed method. From left to right: noise with normalized std 0.02, JPEG compression with Q=60, rotation with $\theta = 45^o$, rescaling with $\alpha = 1.145$, occlusive rigid copy-move. Green indicates correct detection, false alarms are in red.

[6] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra, "A SIFT-based forensic method for copymove attack detection and transformation recovery," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1099–1110, 2011.

[7] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, L. Del Tongo, and G. Serra, "Copy-move forgery detection and localization by means of robust clustering with J-linkage," *Signal Processing: Image Communication*, vol. 28, no. 6, pp. 659–1669, 2013.

[8] B. L. Shivakumar and S. Baboo, "Detection of region duplication forgery in digital images using SURF," *International Journal of Computer Science*, vol. 8, no. 4, pp. 199–205, 2011.

[9] J. Zhao and W. Zha, "Passive forensics for region duplication image forgery based on Harris feature points and Local Binary Patterns," in *Mathematical Problems in Engineering*, 2013, pp. 1–12.

[10] J.-M. Guo, Y.-. Liu, and Z.-J. Wu, "Duplication forgery detection using improved DAISY descriptor," *Expert Systems with Applications*, vol. 40, pp. 707–714, 2013.

[11] P. Kakar and N. Sudha, "Exposing postprocessed copy-paste forgeries through transform-invariant features," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1018–1028, June 2012.

[12] L. Chen, W.-. Lu, J. Ni, W. Sun, and J. Huang, "Region duplication detection based on Harris corner points and step sector statistics," *J. Vis. Commun. Image R.*, vol. 24, pp. 244–254, 2013.

[13] A. Langille and M. Gong, "An efficient match-based duplication detection algorithm," in *Canadian Conf. on Computer and Robot Vision*, 2006.

[14] J. Fridrich, D. Soukal, and J. Lukás, "Detection of copy-move forgery in digital images," in *proc. of Digital Forensic Research Workshop*, 2003.

[15] W. Sun Y. Huang, W. Lu and D. Long, "Improved DCT-based detection of copy-move forgery in images," *Forensic Science International*, vol. 3, pp. 178–184, 2011.

[16] L. Fan Y. Cao, T. Gao and Q. Yang, "A robust detection algorithm for copy-move forgery in digital images," *Forensic Science International*, vol. 214, pp. 33–43, January 2012.

[17] G. Muhammada, M. Hussain, and G. Bebis, "Passive copy move image forgery detection using undecimated dyadic wavelet transform," *Digital Investigation*, vol. 9, pp. 49–57, 2012.

[18] B. Mahdian and S. Saic, "Detection of copymove forgery using a method based on blur moment invariants," *Forensic Science International*, vol. 171, pp. 180–189, 2007.

[19] J. Zhao and J. Guo, "Passive forensics for copy-move image forgery using a method based on DCT and SVD," *Forensic Science International*, vol. 233, pp. 158–166, 2013.

[20] S.-J. Ryu, M. Kirchner, M.-J. Lee, and H.-K. Lee, "Rotation invariant localization of duplicated image regions based on Zernike moments," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1355–1370, August 2013.

[21] Y. Li, "Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching," *Forensic Science International*, vol. 224, pp. 59–67, 2013.

[22] L. Li, S. Li, H. Zhu, and X. Wub, "Detecting copy-move forgery under affine transforms for image forensics," *Computers and Electrical Engineering*, vol. 40, no. 6, pp. 1951–1962, August 2014.

[23] S. Bayram, H. Sencar, and N. Memon, "An efficient and robust method for detecting copy-move forgery," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 2009, pp. 1053–1056.

[24] T. Jing, X. Li, and F. Zhang, "Image tamper detection algorithm based on Radon and Fourier-Mellin transform," in *IEEE International Conference on Information Theory and Information Security*, 2010, pp. 212–215.

[25] Q. Wu, S. Wang, and X. Zhang, "Detection of image region-duplication with rotation and scaling tolerance," in *Computational Collective Intelligence. Technologies and Applications LNCS*, 2010, vol. 6421, pp. 100–108.

[26] Q. Wu, S. Wang, and X. Zhang, "Log-polar based scheme for revealing duplicated regions in digital images," *IEEE Signal Processing Letters*, vol. 18, no. 10, pp. 559–652, 2011.

[27] S. Bravo-Solorio and A. K. Nandi, "Automated detection and localisation of duplicated regions affected by reflection,rotation and scaling in image forensics," *Signal Processing*, vol. 91, pp. 1759–1770, 2011.

[28] M. Muja and D.G. Lowe, "Fast approximate nearest neighbors with

automatic algorithm configuration," in *International Conference on Computer Vision Theory and Applications*, February 2009, pp. 331–340.
[29] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *proc. of Int. Conf. Very Large Data Bases*, 1999, pp. 518–529.
[30] C. Barnes, E. Shechtman, A. Finkelstein, and D.B. Goldman, "PatchMatch: a randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, 2009.
[31] D. Cozzolino, G. Poggi, and L. Verdoliva, "Copy-move forgery detection based on PatchMatch," in *IEEE International Conference on Image Processing*, October 2014, pp. 5312–5316.
[32] A. Andoni, M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, *Locality-Sensitive Hashing Scheme Based on p-Stable Distributions*, T. Darrell and P. Indyk and G. Shakhnarovich (eds.), MIT Press, 2006.
[33] C. Barnes, E. Shechtman, D.B. Goldman, and A. Finkelstein, "The Generalized PatchMatch Correspondence Algorithm," in *European Conference on Computer Vision*, 2010, vol. 6313, pp. 29–43.
[34] D. Cozzolino, D. Gragnaniello, and L.Verdoliva, "Image forgery detection through residual-based local descriptors and block-matching," in *IEEE Conference on Image Processing (ICIP)*, October 2014, pp. 5297–5301.
[35] D. Cozzolino, D. Gragnaniello, and L.Verdoliva, "Image forgery localization through the fusion of camera-based, feature-based and pixel-based techniques," in *IEEE Conference on Image Processing (ICIP)*, October 2014, pp. 5302–5306.
[36] S. Korman and S. Avidan, "Coherency sensitive hashing," in *International Conference on Computer Vision*, 2011, pp. 1607–1614.
[37] I. Olonetsky and S. Avidan, "TreeCann - k-d tree coherence approximate nearest neighbor algorithm," in *European Conference on Computer Vision*, 2012, vol. 7575, pp. 602–615.
[38] Y. HaCohen, E. Schechtman, D.B. Goldman, and D. Lischinski, "Non-rigid dense correspondence with applications for image enhancement," *ACM Transactions on Graphics*, vol. 30, no. 4, July 2011.
[39] V. Christlein, C. Riess, and E. Angelopoulou, "On rotation invariance in copy-move forgery detection," in *IEEE International Workshop on Information Forensics and Security*, December 2010.
[40] M. Kutner, C. Nachtsheim, J. Neter, and W. Li, *Applied linear statistical models*, McGraw-Hill, 2004.
[41] P.W. Holland and R.E. Welsch, "Robust regression using iteratively reweighted least-squares," *Communications in Statistics Theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.
[42] Y. N. Hsu, H.H. Arsenault, and G. April, "Rotation-invariant digital pattern recognition using circular harmonic expansion," *Appl. Opt.*, vol. 21, pp. 4012–4015, 1982.
[43] M.R. Teague, "Image analysis via the general theory of moments," *Opt. Soc. Amer.*, vol. 70, no. 8, pp. 920–930, 1980.
[44] P.-T. Yap, X. Jiang, and A.C. Kot, "Two-dimensional polar harmonic transforms for invariant image representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1259–1270, July 2010.
[45] Y. Sheng and H.H. Arsenault, "Experiments on pattern recognition using invariant Fourier Mellin descriptors," *J. Opt. Soc. Amer.*, vol. 3, pp. 771–776, 1986.
[46] Y. Xin, M. Pawlak, and S. Liao, "Accurate computation of Zernike moments in polar coordinates," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 581–587, Febrauary 2007.
[47] S. X. Liao and M. Pawlak, "On the accuracy of Zernike moments for image analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1358–1364, December 1998.
[48] P. E. Zwicke and Z. Kiss, "A new implementation of the Mellin transform and its application to radar classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 2, pp. 191–199, 1983.
[49] S. Ryu, M. Lee, and H. Lee, "Detection of Copy-Rotate-Move Forgery using Zernike Moments," in *Information Hiding Conference*, 2010, pp. 51 – 65.
[50] C. Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y. M. Lui, "Rotation, scale,and translation resilient watermarking for images," *IEEE Transactions on Image Processing*, vol. 10, pp. 767–782, 2001.

Davide Cozzolino received the Laurea degree in computer engineering and the Ph.D. degree in information engineering from the University Federico II of Naples, Italy, in December 2011 and April 2015, respectively. He is currently a Post Doc at the same University. His study and research interests include image processing, particularly forgery detection and localization.

Giovanni Poggi is currently Full Professor of Telecommunications with the Department of Electrical Engineering and Information Technology of the University University of Naples Federico II, Naples, Italy. He is also Coordinator of the Telecommunication Engineering School. His research interests are in statistical image processing, including compression, restoration, segmentation, and classification, with application to the area of remote-sensing, both optical and SAR, and digital forensics. Prof. Poggi has been an Associate Editor for the IEEE Transactions on Image Processing and Elsevier Signal Processing

Luisa Verdoliva is currently Assistant Professor of Telecommunications with the Department of Electrical Engineering and Information Technology of the University University of Naples Federico II, Naples, Italy. Her research is on image processing, in particular compression and restoration of remote-sensing images, both optical and SAR, and digital forensics. Dr. Verdoliva lead the GRIP team of the Federico II University who ranked first in both the forgery detection and localization phases of the First Image Forensics Challenge organized in 2013 by the IEEE Information Forensics and Security Technical Committee (IFS-TC).