Volume 171, Nos. 2–3, 13th September 2007
ISSN 0379-0738
ELSEVIER
FORENSIC SCIENCE INTERNATIONAL
FSI
An international journal dedicated to the applications of medicine and science in the administration of justice
Editor-in-Chief
P. Saukko
Associate Editors
A. Carracedo
C. Cattaneo
O.H. Drummer
P. Margot
G. Willems

Available online at www.sciencedirect.com

ScienceDirect

Forensic Science International 171 (2007) 180–189

Forensic Science International

www.elsevier.com/locate/forsciint

Detection of copy–move forgery using a method based on blur moment invariants

Babak Mahdian [*], Stanislav Saic

Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 4, 182 08 Prague 8, Czech Republic

Received 5 June 2006; received in revised form 9 October 2006; accepted 7 November 2006
Available online 11 December 2006

Abstract

In our society digital images are a powerful and widely used communication medium. They have an important impact on our life. In recent years, due to the advent of high-performance commodity hardware and improved human–computer interfaces, it has become relatively easy to create fake images. Modern, easy to use image processing software enables forgeries that are undetectable by the naked eye. In this work we propose a method to automatically detect and localize duplicated regions in digital images. The presence of duplicated regions in an image may signify a common type of forgery called copy–move forgery. The method is based on blur moment invariants, which allows successful detection of copy–move forgery, even when blur degradation, additional noise, or arbitrary contrast changes are present in the duplicated regions. These modifications are commonly used techniques to conceal traces of copy–move forgery. Our method works equally well for lossy format such as JPEG. We demonstrate our method on several images affected by copy–move forgery.

Keywords: Authentication; Image forensics; Image tampering; Duplicated image regions; Copy–move forgery; Image moments

1. Introduction

In our society digital images are a powerful and widely used medium of communication, containing a huge amount of information. They are a compact and easy way in which to represent the world that surrounds us. The question is, how much can we trust a photograph which is not obtained from a secure source.

Nowadays, images have an important impact on our society and play a crucial role in most people's lives. Without a doubt, image authenticity is significant in many social areas. For instance, the trustworthiness of photographs has an essential role in courtrooms, where they are used as evidence. Every day newspapers and magazines depend on digital images. In the medical field physicians make critical decisions based on digital images. As a consequence, we should pay a special attention to the field of image authenticity.

As pointed out in [1], photograph tampering has a long history. In today's digital age, due to the advent of low-cost, high-performance computers, more friendly human–computer interface, and the availability of many powerful and easy to control image processing and editing software packages, digital images have become easy to manipulate and edit even for non-professional users. It is possible to change the information represented by an image and create forgeries, which are indistinguishable by naked eye from authentic photographs. This introduces a need for a reliable tamper detection system for digital images. Such a system can determine whether an image has been tampered with. A reliable forgery detection system will be useful in many areas, including: forensic investigation, criminal investigation, insurance processing, surveillance systems, intelligence services, medical imaging, and journalism. Such a system can evaluate the authenticity of digital image.

Existing digital forgery detection methods are divided into active [2–6], and passive (blind) [7–10] approaches. Active approaches could be further divided mainly into digital watermarks and signatures. The passive (blind) approach is regarded as the new direction. In contrast to active approaches,

[*] Corresponding author. Tel.: +420 266052211; fax: +420 284680730.
E-mail addresses: mahdian@utia.cas.cz (B. Mahdian), ssaic@utia.cas.cz (S. Saic).

0379-0738/$ – see front matter © 2006 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.forsciint.2006.11.002

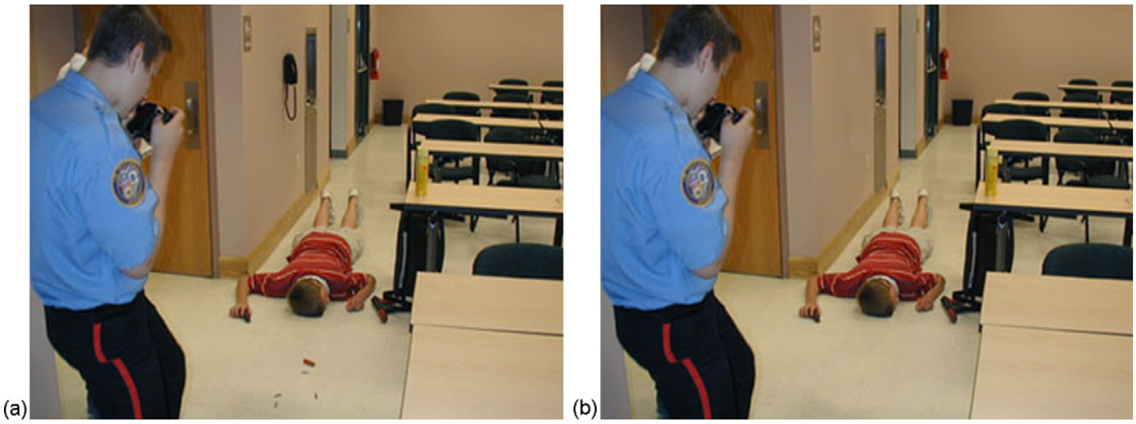

Fig. 1. An example of a copy–move forgery. The photograph of crime scene (from [11]) is altered by the copy–move forgery. The original (left) and forged version (right).

passive approaches do not need any explicit priori information about the image. Therefore, it does not require watermarks or signatures.

It is obvious that there are many ways to manipulate and alter digital images. An attempt of categorization has been proposed by Farid [1]. As mentioned, passive methods are regarded as a new approach and have not yet been thoroughly researched by many. Different methods for identifying each type of forgery must be developed. Then, by fusing the results from each analysis, a decisive conclusion may be drawn.

In this work we focus on detecting a common type of digital image forgery, called copy–move forgery. In copy–move forgery, a part of the image is copied and pasted into another part of the same image, with the intention to hide an object or a region of the image. Fig. 1 shows an example. We can determine whether an image contains this type of forgery by detection of duplicated regions. Duplicated regions may not always match exactly. For example, this could be caused by a lossy compression algorithm, such as JPEG, or by possible use of the retouch tool.

The importance of digital images in forensic science creates a significant need for reliable detection of copy–move forgery. Due to the possibilities of today's standard image processing software, the creation of a high quality copy–move forgery has became particularly easy. Therefore, we can expect that this type of tampering will become more common. For example, with infringement of copyright, blackmail, insurance fraud and other schemes based on digital forgery. However, note that when creating high quality and consistent forgeries, several types of tampering techniques are employed simultaneously. For example, image splicing in combination with copy–move forgery and localized image retouching techniques. Thus, when we consider copy–move forgery, we often assume this tampering technique has been used simultaneously with others. Therefore, by having a reliable technique to detect the copy–move forgery, we will be able to detect forgeries that contain among others this type of tampering.

Fig. 1 shows an example of the use of copy–move forgery in a forensic investigation. Here the photograph of a crime scene is tampered with using the copy–move technique with; intention is to hide some important objects in the photograph. We believe that a reliable tamper detection system will useful in forensic applications, where making decisions are based or affected by imaging.

As pointed out in [7], ideal regions for using copy–move forgery are textured areas with irregular patterns, such as grass. Because the copied areas will likely blend with the background it will be very difficult for the human eye to detect any suspicious artifacts. Another fact which complicates the detection of this type of tampering is that the copied regions come from the same image. They therefore have similar properties, such as the noise component or color palette. It makes the use of statistical measures to find irregularities in different parts of the image impossible.

1.1. *State of the art*

As mentioned, despite of the strong need for a reliable detection of digital forgeries in the absence of watermarks and signatures, this area has an unexplored character. The field of copy–move forgery detection is even smaller: only two publications concerned with this topic have been found.

The first one has been proposed by Fridrich et al. [7]. This method tiles the image by overlapping blocks. The detection of duplicated regions is based on matching the quantizied lexicographically sorted discrete cosine transform (DCT) coefficients of overlapping image blocks. The lexicographically sorting of DCT coefficients is carried out mainly to reduce the computational complexity of the matching step. The second method has been proposed by Popescu and Farid [8] and is similar to [7]. This method differs from [7] mainly in the representation of overlapping image blocks. Here, the principal component transform (PCT) has been employed in place of DCT. The representation of blocks by this method has better discriminating features.

2. Detection of duplicated regions

To detect the copy–move forgery we focus our aim on detection of duplicated regions in the image. Since duplicated regions may signify this type of forgery. Existing copy–move forgery detection methods have limited abilities. In most cases of forgery investigated, they were able to detect duplicated regions in the tampered image despite of the presence of an acceptable amount of noise. This is mainly caused due to a quantitation step or a similarity threshold. Additionally, it allows for analysis images compressed with a lossy algorithm, such as JPEG. However, a skilled falsifier will be able to produce work undetectable by these methods.

Existing copy–move forgery detection methods take into account that in a copy–move forgery, duplicated regions may not match exactly. After the creation of the duplicated regions, the falsifier will often introduce variation to the duplicated region to make its presence less obvious. This can be carried out using retouch or other localized image processing tools. Existing methods are mainly concerned with additive noise, which can be added to the duplicated regions and causes that duplicated regions to match only approximately. This can be also achieved easily by blurring. Since the falsifier needs to keep the faked regions consistent with the rest of the image, we can assume that these degradations will not be strong.

An experienced falsifier can use a simple 2D convolution of the duplicated region with a blur filter mask to make detection of forgery even more difficult. Thus, to improve the detection abilities of the current available approaches, we can describe analyzed regions by some features invariant to the presence of unknown blur. From a mathematical point of view, we are looking for a functional B, which is invariant with respect to blur degradation. In other words, B satisfies the condition $B(f) = B(D(f))$, where operator D denotes the blur degradation. Furthermore, due to the fact that the falsifier can also use additive noise to make detection more difficult, these invariants should also work well with the presence of additive noise.

The aforementioned requirements are satisfied by blur moment invariants. They have been previously addressed by Flusser and Suk [12,13] and have found successful applications in many areas of image processing—such as: in face recognition on out-of-focused photographs, template-to-scene matching of satellite images, in focus/defocus quantitative measurement, etc. Blur moment invariants are suitable to represent image regions due to the fact that they are not affected by the blur degradation present in the region. Another advantage of moment invariants is that they are computed by a summation over the whole image, so they are not significantly affected by additive zero-mean noise.

We will define the problem of copy–move forgery detection in the following way. Given an image $I(x, y)$ containing an arbitrary number of duplicated regions of unknown location and shape, our task is to determine the presence of such regions in the image and to localize them. The aim of this investigation is create a method that can detect duplicated regions, even when some contain degradations caused by convolution with a shift-invariant symmetric energy-preserving point spread function (PSF) and additive random noise. As mentioned, the method should be able to also find duplicated regions which only match approximately. Formally: let $f(x, y)$ be a function describing the original region and $g(x, y)$ the acquired region created by the falsifier via convolution of $f(x, y)$ with the PSF, then

$$g(x,y) = (f * h)(x,y) + n(x,y),$$

where $h(x, y)$ is a shift invariant PSF, $n(x, y)$ an additive random noise and * denotes a 2D convolution. We would like to find all $g(x, y)$ created from $f(x, y)$ and $h(x, y)$ via the above equation. Due to the fact that moment invariants are utilized as features, we will assume the following restrictions. Both $f(x, y) \in L_1$ and $g(x, y) \in L_1$ are real functions and have a bounded support and nonzero integral:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\mathrm{d}x\,\mathrm{d}y > 0, \qquad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\,\mathrm{d}x\,\mathrm{d}y > 0.$$

Moreover, the PSF is assumed to be axial symmetric and energy-preserving, i.e.:

$$h(x,y) = h(-x,y) = h(y,x), \qquad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x,y)\,\mathrm{d}x\,\mathrm{d}y = 1.$$

These assumptions do not cause a significant limitation. Most imaging systems that we are interested in perform some type of symmetry. By supposing other types of symmetries, like central, four-fold or circular symmetry, we can also construct blur invariants based on moments. However, generally, the higher degree of symmetry of the PSF is assumed, the more invariants can be obtained [12].

The proposed copy–move forgery detection method is based on a few main steps:

- tiling the image with overlapping blocks,
- blur moment invariants representation of the overlapping blocks,
- principal component transformation,
- *k–d* tree representation,
- blocks similarity analyses,
- duplicated regions map creation.

Each step is explained separately in the following sections.

2.1. Overlapping blocks

This method begins with the image being tiled by blocks of $R \times R$ pixels. Blocks are assumed to be smaller than the size of the duplicated regions, which have to be detected. Blocks are horizontally slid by one pixel rightwards starting with the upper left corner and ending with the bottom right corner. The total number of overlapping blocks for an image of $M \times N$ pixels is $(M - R + 1) \times (N - R + 1)$. For instance, an image with the size of 640×480 with blocks of size 20×20 yields 286 281 overlapping blocks.

2.2. Blur invariants representation

Each block is represented by blur invariants, which are functions of central moments. The two-dimensional $(p + q)$th order moment m_{pq} of image function $f(x, y)$ is defined by the integral:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)\,\mathrm{d}x\,\mathrm{d}y.$$

The two-dimensional $(p+q)$th order central moment μ_{pq} of $f(x, y)$ is defined as

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-x_t)^p (y-y_t)^q f(x,y)\,\mathrm{d}x\,\mathrm{d}y,$$

where the coordinates (x_t, y_t) given by the relations:

$$x_t = \frac{m_{10}}{m_{00}}, \qquad y_t = \frac{m_{01}}{m_{00}}$$

denote the centroid or the center of gravity of $f(x, y)$. By supposing that

$$g(x,y) = (f * h)(x,y),$$

we can simply derive that central moments of $g(x, y)$ are defined as

$$\mu_{pq}^{(g)} = \sum_{k=0}^{p}\sum_{j=0}^{q}\binom{p}{k}\binom{q}{j}\mu_{kj}^{(f)}\mu_{p-k,q-j}^{(h)}.$$

We are looking for features invariant to blur. Feature B is called blur invariant if

$$B^{(f)} = B^{(f*h)} = B^{(g)}.$$

As mentioned, we consider only symmetric $h(x, y)$. By applying the algorithm as derived and described in [12,14], we can construct blur invariants based on central moments of any order by using the following recursive relation:

$$B(p,q) = \mu_{pq} - \alpha\mu_{qp} - \frac{1}{\mu_{00}}\sum_{n=0}^{K}\sum_{i=m_1}^{m_2}\binom{p}{t-2i} \times \binom{q}{2i} B(p-t+2i, q-2i)\mu_{t=2i,2i},$$

where

$$K = \left[\frac{p+q-4}{2}\right],$$
$$t = 2(K-n+1), \quad t = 2(K-n+1),$$
$$m_1 = \max\left(0, \left[\frac{t-p+1}{2}\right]\right), \quad m_2 = \min\left(\frac{t}{2}, \left[\frac{q}{2}\right]\right),$$
$$\alpha = 1 \Leftrightarrow p \wedge q \text{ are even}, \quad \alpha = 0 \Leftrightarrow p \vee q \text{ are odd}.$$

The proposed algorithm uses 24 blur invariants up to the seventh order to create the feature vector:

$$B = \{B_1, B_2, B_3, \ldots, B_{23}, B_{24}\}$$

of each block. Some examples of utilized invariants in their explicit forms are listed below:

$$B_1 = \mu_{11}, \quad B_2 = \mu_{12}, \quad B_3 = \mu_{21}, \quad B_4 = \mu_{03}, \quad B_5 = \mu_{30},$$
$$B_6 = \mu_{13} - \frac{3\mu_{02}\mu_{11}}{\mu_{00}}, \quad B_7 = \mu_{31} - \frac{3\mu_{20}\mu_{11}}{\mu_{00}},$$
$$B_8 = \mu_{32} - \frac{3\mu_{12}\mu_{20} + \mu_{30}\mu_{02}}{\mu_{00}},$$
$$B_9 = \mu_{23} - \frac{3\mu_{21}\mu_{02} + \mu_{03}\mu_{20}}{\mu_{00}}.$$

Because we will use an Euclidean metric space, the invariants should be normalized to have the same weight. To achieve this, the normalization described in [13,14] is used

$$B_i' = \frac{B_i}{(R/2)^r \mu_{00}},$$

where R is the block size and r the order of B_i. Please note that in this manner normalized invariants are also invariant to contrast changes, which improves the duplication detection abilities of the algorithm.

As is obvious, each block is represented by a feature vector of length 24 in the case of gray-scale images. For RGB images, moment invariants of each block in each channel are computed separately, resulting in feature vector $\mathbf{B}_{\text{rgb}} = \{\mathbf{B}_{\text{red}}, \mathbf{B}_{\text{green}}, \mathbf{B}_{\text{blue}}\}$ of length 72.

2.2.1. *Stability of moment invariants under additive random noise*

As mentioned moment invariants are computed by a summation over the whole image, so they are not significantly affected by additive zero-mean noise. For a more detailed discussion about the robustness of moment invariants to the additive random noise and an example of the stability of moments corrupted by Gaussian zero-mean noise with different standard deviations, see [14].

2.2.2. *Stability of moment invariants with respect to boundary effect*

So far we have assumed that we can work with blocks that result from full linear convolution with the original block and a shift-invariant filter. However, in practice we work with truncated versions of the filtered blocks, which are additionally corrupted by the neighboring pixels. If our blocks have $R \times R$ pixels and the size of PSF support is $H \times H$ pixels, the correct size of the resulting block must be $(R + H - 1) \times (R - H - 1)$. In our case, the value of H is unknown. If $H \ll R$ the errors of invariant calculation caused by the boundary effect is negligible. If H is relatively large as in the case of heavy blur, the boundary effect will cause significant miscalculations of the invariant values. For an experiment on this topic, see [14].

2.3. *Principal component transformation*

In the case of an RGB image, the dimension of the feature vector is 72 (24 invariants per channel). Using the principal component transformation we reduce this dimension. Typically the new orthogonal space has dimension 9 (fraction of the ignored variance along the principal axes is set to 0.01). In PCT, the orthogonal basis set is given by the eigenvectors set of the covariance matrix of the original vectors. Thus, it can be easily computed on very large data sets. Note that PCT preserves the Euclidean distance among blocks.

Also please note that if the distribution of the overlapping vectors, $\mathbf{B}_i$, is a multi-dimensional Gaussian, then the PCT using the first N eigenvectors of the PCT basis gives the best N-dimensional approximation in the least squares sense. The newly created space by PCT is a convenient space in which to identify duplicated blocks. Due to the fact that there exist many publications about PCT (for example [15]) this topic will be not discussed in further detail here.

2.4. *k–d tree representation*

In blocks similarity analysis (see next section) we will need to efficiently identify all blocks which are in a desired similarity relation with each analyzed block. A simple exhaustive search computes the distance from the block to all others. This approach is very inefficient and its computational cost is $O(N)$. To improve the efficiency of finding neighboring blocks, some hierarchical structures have been proposed. The *k–d* tree is a commonly used structure for searching for nearest neighbors.

The *k–d* tree preprocesses data into a data structure allowing us to make efficient range queries. It is a binary tree that stores points of a k-dimensional space in the leaves. In each internal point, the tree divides the k-dimensional space into two parts with a $(k-1)$-dimensional hyperplane. If a *k–d* tree consists of N records, it requires $O(N \log_2 N)$ operations to be constructed and $O(\log_2 N)$ to be searched. Because of these reasons, the proposed method transforms blocks representation to a k-d tree for a more effective closest neighbors search. Since there exist many publications about *k–d* tree [16,17], this topic will be not discussed here in further detail.

2.5. *Blocks similarity analyses*

The main idea of this step is that a duplicated region consists of many neighboring duplicated blocks. If we find two similar blocks in the analyzed space and if their neighborhoods are also similar to each other, there is a high probability that they are duplicated and they must be labeled.

The similarity measure s employed here is defined by the following formula:

$$s(\mathbf{B}_i, \mathbf{B}_j) = \frac{1}{1+\rho(\mathbf{B}_i, \mathbf{B}_j)},$$

where ρ is a distance measure in the Euclidean space:

$$\rho(\mathbf{B}_i, \mathbf{B}_j) = \left(\sum_{k=1}^{\dim} (\mathbf{B}_i[k] - \mathbf{B}_j[k])^2 \right)^{1/2}.$$

For each analyzed block represented by the feature vector $\mathbf{B}$, we look for all blocks with an equal or larger similarity relation. It must be an equal or larger similarity to the threshold T. Fig. 2 shows an example of a two-dimensional feature space. In this example black dots represent overlapping blocks. The method finds all similar blocks for each one (similar to the nearest neighbors search) and analyses their neighborhood. This is done efficiently using the *k–d* tree structure, which was created in the previous step.

If $s(\mathbf{B}_i, \mathbf{B}_j) \geq T$, where T is the minimum required similarity, we also analyze the neighborhood of $\mathbf{B}_i$ and $\mathbf{B}_j$. Note that the threshold T plays a very important role. It expresses the degree of reliability with which blocks i and j correspond with each other. It is obvious that the choice of T directly affects the precision of results of the method. Due to the possibility of the presence of additive noise, a boundary effect, or JPEG compression, this threshold should not be set to 1.

After two blocks with the required similarity have been found, a verification step begins. In the verification step, similar blocks with different neighbors will be eliminated.

For analyzing the blocks neighborhood, we choose 16 neighboring blocks with a maximum distance of 4 pixels from the analyzed block (distance from their upper left corners). If 95% of these neighboring blocks satisfy the similarity condition, the analyzed block is labeled as duplicated. More formally, block 1 with coordinates (i, j) and block 2 with coordinates (k, l) are labeled as duplicated if

$$s(\text{block}(i+x_r, j+y_r), \text{block}(k+x_r, l+y_r)) \geq T,$$

where $x \in \langle -4, -3, \ldots, 4\rangle$ and $y \in \langle -4, -3, \ldots, 3, 4\rangle$ and $r = 1, \ldots, 16$. This part of the algorithm also determines the minimum size of the copied area, which can be detected by the algorithm.

To have more precise results, the verification step additionally uses information about the image distances of

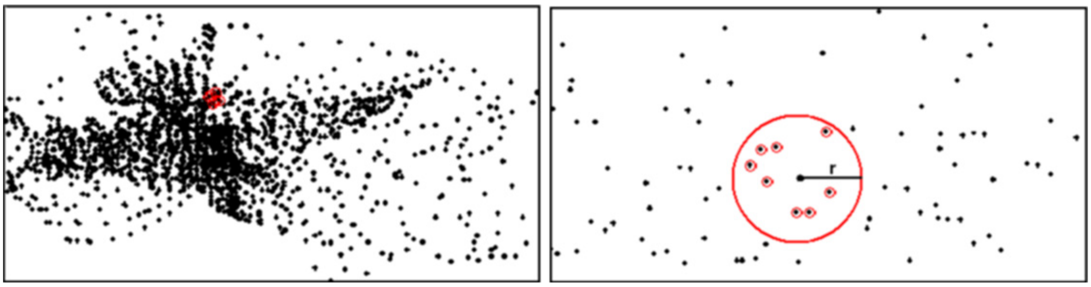

Fig. 2. A two-dimensional feature space. Black dots represent overlapping blocks, which have to be analyzed (left image). The method finds all similar bocks to each block and analyses their neighborhood (right image). In other words, we find all blocks inside a circle, which has the analyzed block as centroid. The radius r is determined by the similarity threshold.

analyzed blocks. If $s(\text{block}(i, j), \text{block}(k, l)) \geq T$, but $\sqrt{(i-k)^2 + (j-l)^2} \leq D$, these blocks will not be further analyzed and will not be assigned as duplicated. Threshold D is a user-defined parameter determining the minimum image distance between duplicated regions.

The output of this section is matrix Q with the same size as the input image. Elements of this matrix are either zero or one. An element of this matrix is set to one if the block at this position is duplicated.

2.6. Duplicated regions map creation

The output of the method is a duplicated regions map showing the image regions, which are likely duplicated. It is created by the multiplication of each element of $I(x, y)$ by its respective element in $Q(x, y)$. Matrix $Q(x, y)$ is created in the previous section.

3. Results

In this section we show a few examples of copy–move forgery and their corresponding duplication maps constructed by the proposed detection method. For this purpose an experimental version of the proposed method was implemented in Matlab. The output of the method is a duplication map, in which likely duplicated regions are shown. Parameters of the method were set to $R = 20$ (block size), $T = 0.97$ (similarity threshold), $D = 24$ (blocks image distance threshold). In the PCT step, the fraction of the ignored variance along the principal axes, ε, was set to 0.01. Please note that the computational time of the method is highly dependent on these parameters (specially on T and ε). In all cases, the tampering was realized by copying and pasting a region in the image with intent to conceal a person or object. Additionally, in order to make the detection of forgery more difficult and interesting most examples contain further manipulations of the pasted region, such as blurring.

The first example is presented in Fig. 3. Fig. 3(c) shows the output of the method applied to the tampered image shown in Fig. 3(b). In this example, no further manipulations were carried out with the tampered regions. The tampered image was saved in JPEG format with quality factor 90. The output shows that the proposed method correctly detected the duplicated regions.

Fig. 4(c) shows the duplication map created by the proposed method applied to Fig. 4(b). Fig. 4(e) shows the output of the method applied to Fig. 4(d). Also here, in both examples, a part of the image was copied and pasted somewhere else in the same image with the intent to cover a part of the image. The tampered image in Fig. 4(b) was saved in JPEG format with quality factor 70. The tampered region in Fig. 4(d) was additionally blurred with a simple average mask of size 3 × 3. The tampered image in this case was also saved in JPEG format with quality factor 70.

Fig. 5(c) shows the duplication map generated by applying the proposed method to Fig. 5(b). Here, in addition, the forged region was blurred with a Gaussian blur filter with radius 0.2 pixels. Furthermore, after the blurring operation, the tampered area was corrupted by additive zero-mean Gaussian noise with standard deviation 5. The tampered image in this case was saved in JPEG format quality 90.

Fig. 6(c) shows the duplication map created by applying our method to Fig. 6(b). In this example, the tampered region was

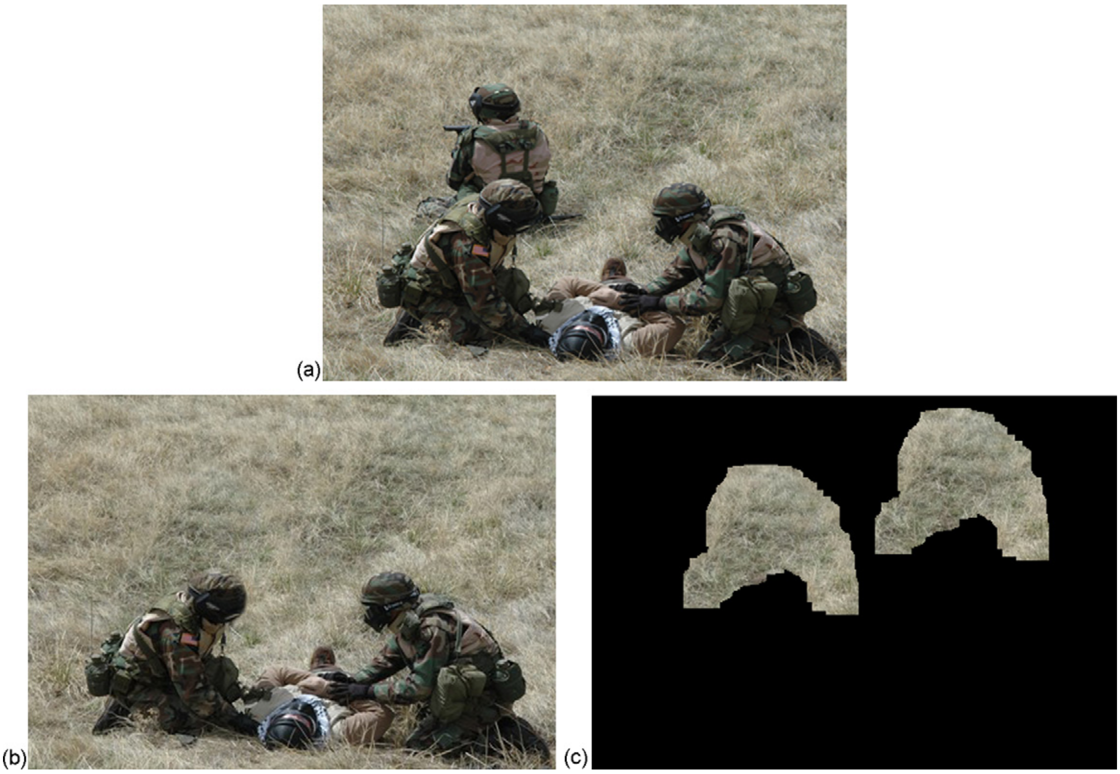

Fig. 3. Shown are the original version of the test image (a), its forged version (b) and the constructed duplication map (c).

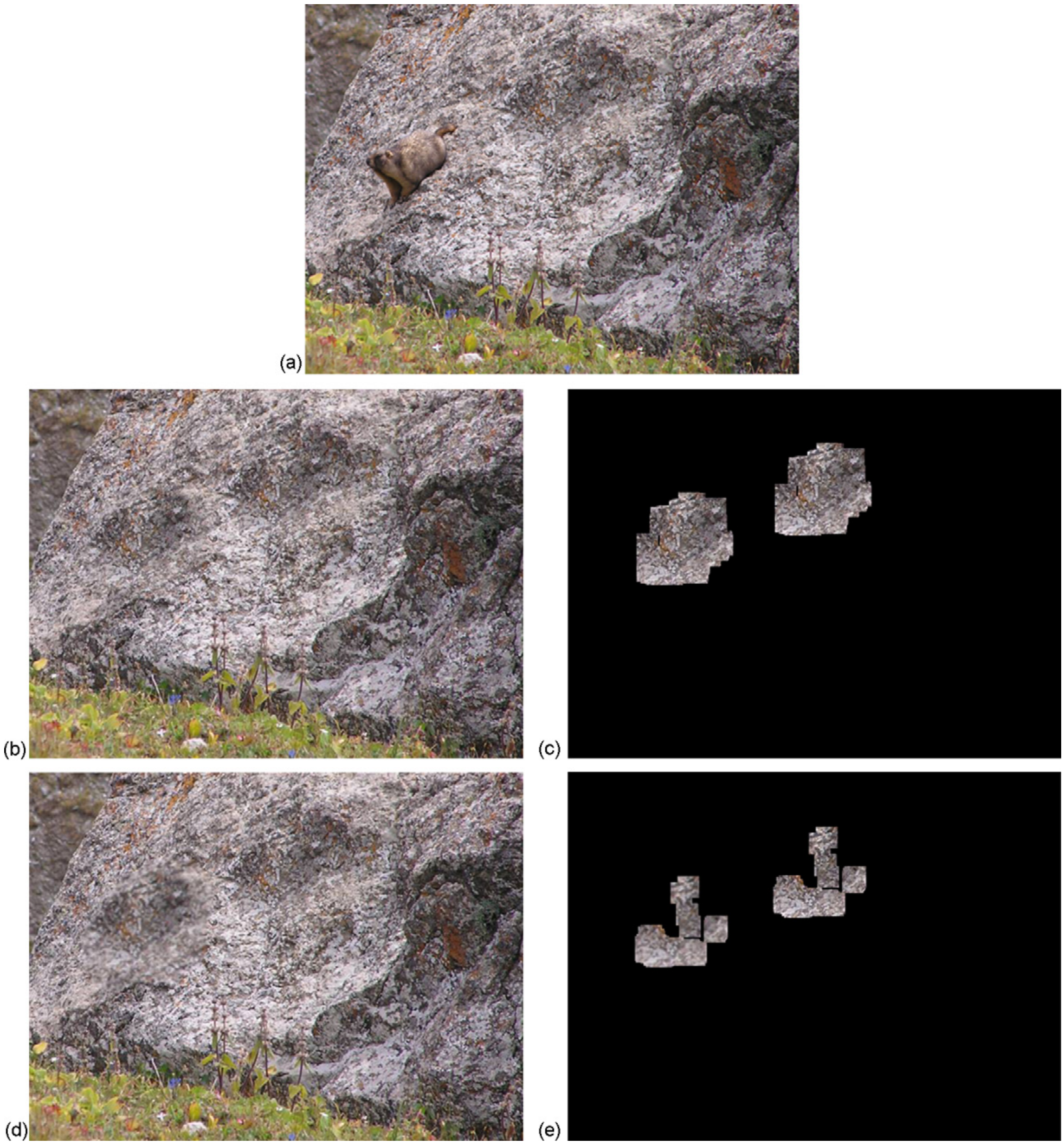

Fig. 4. Shown are the original version of the test image (a), its forged versions (b) and (d) and constructed duplication maps (c) and (e).

blurred with a Gaussian blur filter with radius 0.3 pixels. The tampered image in this case was saved in JPEG format quality 80.

Fig. 7(c) shows the duplication map generated by our method applied to Fig. 7(b). Here, the forged region was blurred with a Gaussian blur filter with radius 0.7 pixels. The tampered image in this case was saved in JPEG format quality 50. In this example we can additionally see some falsely identified regions. Such false results could be expected from all duplicated image region detection algorithms based on block analysis and matching (especially in uniform areas of the image).

4. Discussion

Our method's results show that the use of blur moment invariants can improve the detection abilities of the copy–move forgery detection methods. By using blur moment invariants we are able to additionally detect duplicated regions with presence of acceptable blur and additive Gaussian noise. By normalizing moment invariants they also become invariant against contrast changes. The proposed method also works with lossy JPEG format images.

In comparison to other existing methods [7,8] based on DCT and PCT, we notice that the description of duplicated regions by moment invariants has better discriminating features in cases where the regions were modified by blurring. Another benefit of using invariants in comparison to DCT and PCT is the ability to detect duplicated regions in presence of contrast changes; normalized blur moment invariants are invariant to them. Also please note that moment invariants are computed by a summation over the whole block, therefore they are not supposed to be affected by additive zero-mean noise significantly. We must mention that in general, we can always

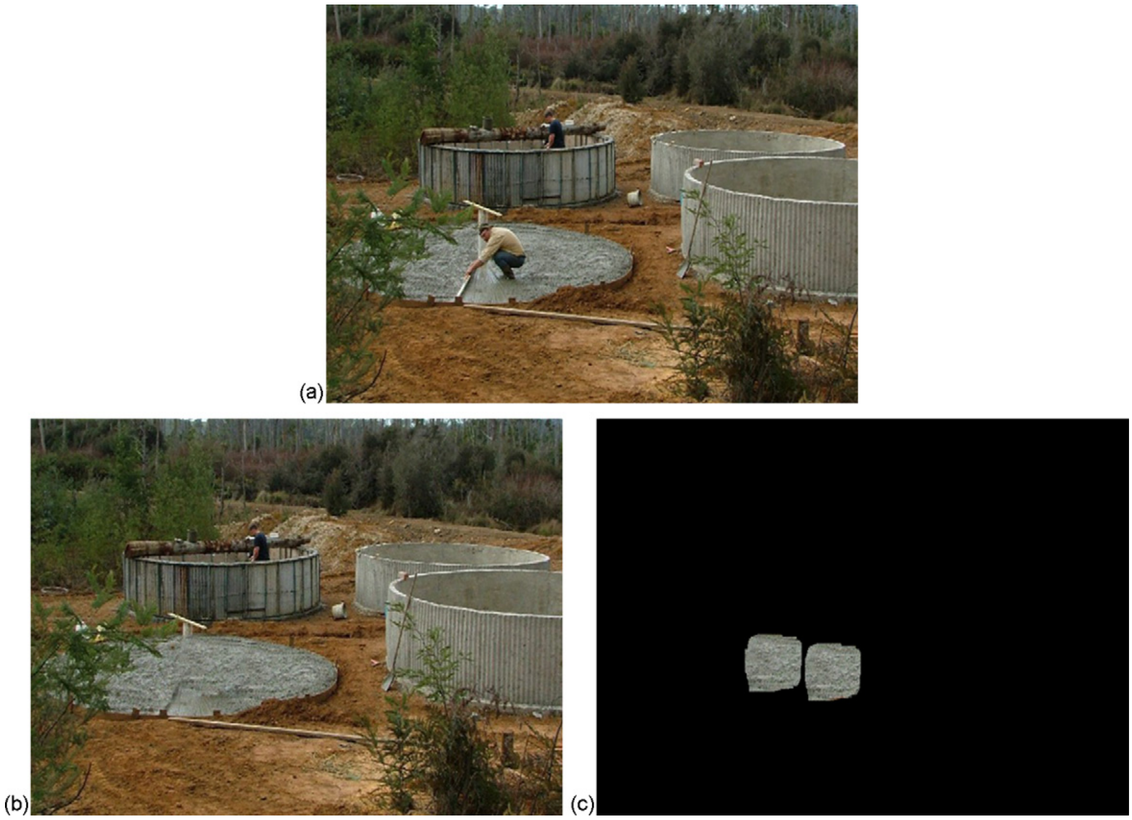

Fig. 5. Shown are the original version of the test image (a), its forged version (b) and the constructed duplication map (c).

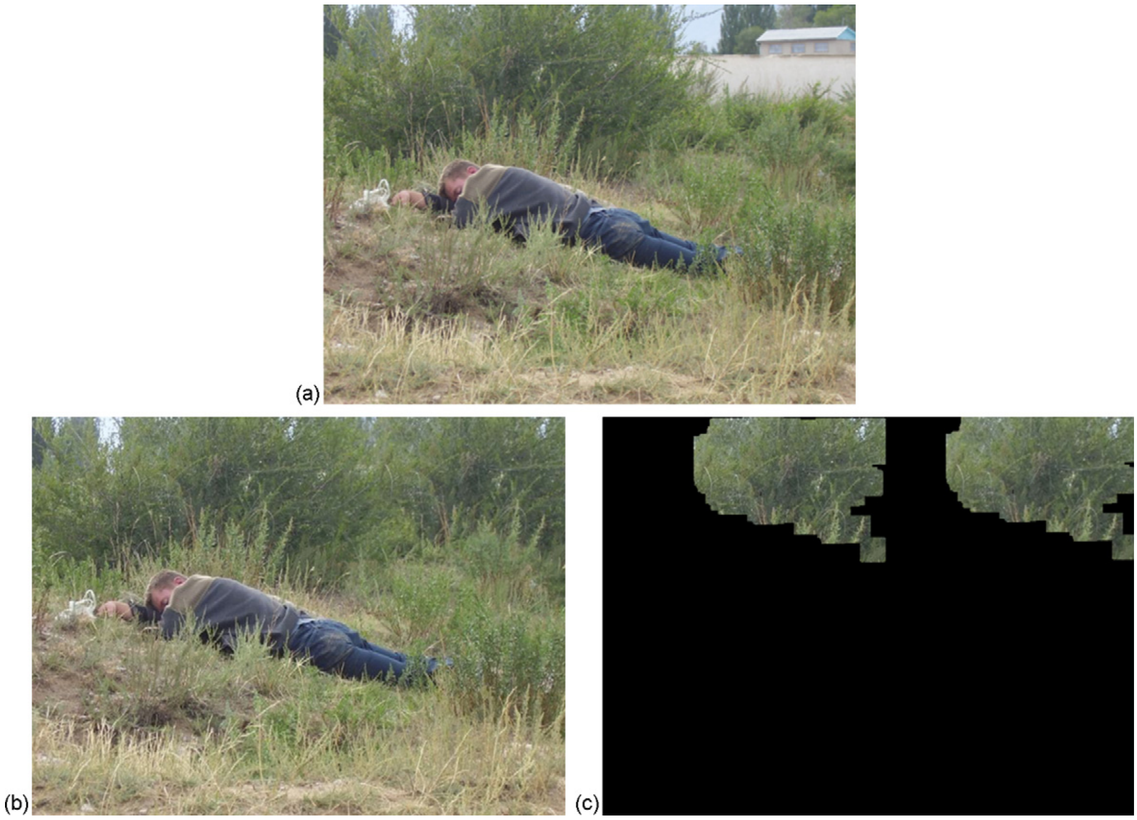

Fig. 6. Shown are the original version of the test image (a), its forged version (b) and the constructed duplication map (c).

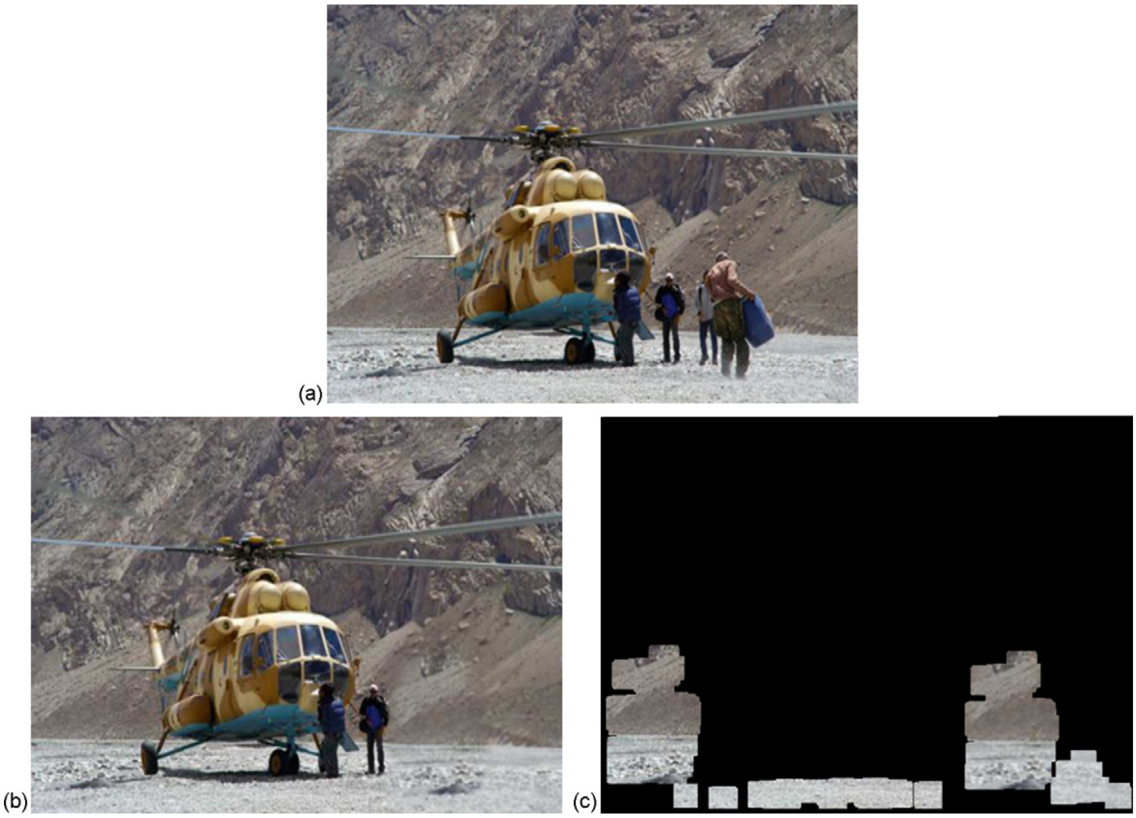

Fig. 7. Shown are the original version of the test image (a), its forged version (b) and the constructed duplication map (c).

find images where DCT or PCT will perform better then blur invariants and vice versa. Notice that all existing methods are directly or indirectly dependent on a similarity or a frequency threshold, which often determines the success of the method more than the block representation method used.

The blur moment invariants approach, like all other existing methods, has a problem with uniform areas in images. Since we are looking for identical or similar areas in the image, it is to be expected that the method will logically label not duplicated parts as duplicated in uniform areas, such as the sky. Thus, a human interpretation of the output of any duplication image regions detection method is obviously necessary.

A disadvantage of the proposed method is its computational time. The average run time of the implemented experimental version with parameters $R = 20$ (block size) and $T = 0.97$ (similarity threshold) for 640 × 480 RGB images on a 2.1 GHz processor and 512 MB RAM is 40 min. The computational time is not the same for images with the same size. It is dependent on each image's characteristics (the dimension of space created after the PCT) and especially on the similarity threshold parameter of the algorithm. It is also important to note that the implemented experimental version was not optimized and there exist possibilities to improve the computational time.

A way to considerably improve the computational time is to eliminate large uniform areas in a preprocessing step and apply the proposed method to the rest of the analyzed image. Then the output of the method could consist of a duplication map and a uniform areas map. Please note that the computational time of an image containing large uniform or very similar areas is higher. This is caused by the fact that most of the computational time is required by the step in which block similarities and neighborhoods are analyzed. As previously mentioned, a 640 × 480 image with overlapping blocks of size 20 × 20 yields 286281 blocks to be analyzed. For each block found similar to the analyzed block, its neighborhood is also analyzed. Logically, having a high number of similar blocks causes protraction of computation. The hierarchical nature of the *k–d* structure allows us to make efficient range queries in multidimensional data. In spite of this efficiency, this step still requires roughly 60% of the computational time of the algorithm.

The second part, which significantly increases the run time of the algorithm, is the computing complexity of moment invariants [14]. This is dependent on the computing complexity of the central moments μ_{pq} and the number of overlapping blocks. Although several methods for fast moments computing have been published recently, most of them are not applicable in our case. They are mostly suitable for binary images only. The direct evaluation of central moments for an $N \times N$ image in the discrete version is

$$\mu_{pq} = \sum_{i=1}^{N}\sum_{j=1}^{N}(i - x_t)^p (i - y_t)^q f_{ij},$$

which needs $O(N^2)$ operations. We need to do these operations for each block. However, since the input image is tiled by overlapping blocks, there exists redundant information which could be utilized to improve the moment invariants computation complexity. This may significantly improve the run time of the proposed method, and was not explored in this work.

5. Conclusion

We have proposed an automatic and robust duplication image regions detection method based on blur moment invariants. It works in complete absence of digital watermarks or signatures and does not need any prior information about the tested image. The proposed method tiles the image with overlapping blocks and uses blur invariants to represent them. The dimension of the blocks representation is reduced by using the principal component transformation. Furthermore, a *k–d* tree is used to efficiently perform range queries in multi-dimensional data for block similarity analysis. The output of the algorithm is a duplicated image regions map.

The experimental results show the high ability of the proposed method to detect copy–move forgery in an image even with the presence of blur, noise or contrast changes in the copied areas. The method even works well with lossy JPEG format data. Thus, we believe that our method can be very useful in many areas of forensic science.

References

[1] H. Farid, Creating and detecting doctored and virtual images: implications to the child pornography prevention act, Technical Report, TR2004-518, Dartmouth College, Hanover, New Hampshire, 2004.

[2] Yeung, M. Minerva, Digital watermarking introduction, CACM 41 (7) (1998) 31–33.

[3] C. Rey, J.L. Dugelay, A survey of watermarking algorithms for image authentication, EURASIP J. Appl. Signal Process. 2002 (6) (2002) 613–621.

[4] J. Fridrich, Methods for tamper detection in digital images, in: Proceedings of the ACM Workshop on Multimedia and Security, Orlando, FL, (October 1999), pp. 19–23.

[5] M. Schneider, S. Chang, A robust content-based digital signature for image authentication, IEEE Int. Conf. Image Process. 2 (1996) 227–230.

[6] C.S. Lu, H.Y. Mark Liao, Structural digital signature for image authentication: an incidental distortion resistant scheme, in: Proceedings of the ACM Multimedia 2000, Los Angeles, CA, (November 2000), pp. 115–118.

[7] J. Fridrich, D. Soukal, J. Lukas, Detection of copy–move forgery in digital images, in: Proceedings of Digital Forensic Research Workshop, Cleveland, OH, August 2003.

[8] A.C. Popescu H. Farid, Exposing digital forgeries by detecting duplicated image regions, Technical Report, TR2004-515, Department of Computer Science, Dartmouth College, Hanover, New Hampshire, 2004.

[9] M.K. Johnson, H. Farid, Exposing digital forgeries by detecting inconsistencies in lighting, in: Proceedings of the ACM Multimedia and Security Workshop, New York, NY, (2005), pp. 1–9.

[10] A.C. Popescu, Statistical tools for digital image forensics, PhD Dissertation, TR2005-531, Department of Computer Science, Dartmouth College, Hanover, New Hampshire, 2005.

[11] http://www.columbuspolice.org.

[12] J. Flusser, T. Suk, Degraded image analysis: an invariant approach, IEEE Trans. Pattern Anal. Machine Intell. 20 (6) (1998) 590–603.

[13] J. Flusser, T. Suk, S. Saic, Image features invariant with respect to blur, Pattern Recogn. 28 (1995) 1723–1732.

[14] J. Flusser, T. Suk, S. Saic, Recognition of blurred image by the method of moments, Pattern Recogn. 5 (1996) 533–538.

[15] R. Duda, P. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.

[16] R. Sagawa, T. Masuda, K. Ikeuchi, Effective nearest neighbor search for aligning and merging range images, in: Proceedings of the 3DIM 2003, 2003, pp. 79–86.

[17] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (1975) 509–526.