Self-Synchronizing Robust Texel Watermarking in Gaussian Scale-Space

Mathias Schlauweg, Dima Pröfrock, Benedikt Zeibich, and Erika Müller
Institute of Communications Engineering, Faculty of Computer Science and Electrical Engineering
University of Rostock
Rostock 18119, Germany
{mathias.schlauweg, dima.proefrock, benedikt.zeibich, erika.mueller}@uni-rostock.de

ABSTRACT

In this paper, we propose a new second generation watermarking method for still images that embeds information in the feature space using rational dither modulation. Embedding at a fixed raster, e.g., the pixel grid, is a big problem in all first generation watermarking approaches. In contrast, our approach does not work depending on any raster. We use a texture-based feature, the so-called gray-level blob in Gaussian scale-space, which is invariant to scaling, rotation, and translation. It is associated with the image content and thus independend of image geometry. Furthermore, we show that our new method is robust against a variety of attacks, such as lossy compression, contrast and luminance enhancement, bluring and sharpening as well as noise adding. In addition, due to embedding at no fixed raster and insertion/deletion error correction even after slight image cropping the watermark can be extracted.

Categories and Subject Descriptors

I.4.9 [**Image Processing and Computer Vision**]: Applications; D.2.11 [**Software Engineering**]: Software Architectures—*Information Hiding*

Keywords

Digital watermarking, Gaussian scale-space, texture-based feature points, RST-invariant, insertion/deletion error correction

1. INTRODUCTION

As opposed to first watermarking approaches, *second generation watermarking* embeds information depending on the image content. In [7], Kutter *et al.* describe this idea as using significant features of the host image for either carrier signal synchronization (helper scheme) or direct watermark embedding. These features can be edges or corners of image objects or spatial relations between objects, properties of object textures or object forms.

MM&Sec'08, September 22–23, 2008, Oxford, United Kingdom.

First generation methods consider the host image as a continuous 2-dimensional function in spatial or transform domain (e.g., DCT, DWT) and embed information by addition/multiplication or replacement. The most important property, and difference to second generation watermarking, is the fact that a fixed raster is used for embedding and extraction. For example, the pixel grid or a separation of the image into fixed-size blocks can be such a raster.

One of the big advantages of first generation watermarking is the often guaranteed big capacity, low complexity and hence low computational effort. But a big disadvantage is the low robustness against cropping, scaling, rotation, translation, and geometric attacks. In most cases, this weakness results from embedding at a fixed raster. Due to, for example, cropping, rotation, or scaling the relation of the fixed raster is destroyed (see Fig. 1) and synchronization within the watermark sequence can get lost during extraction. To improve the robustness against such attacks some approaches use the original image (not always applicable) for re-synchronization during extraction. Other approaches use so-called re-synchronization signals or correlation-based watermarks additionally embedded in the image, resulting in further quality degradations. Apart from that a possible security hole is opened because an attacker easily can find and delete these template signals.

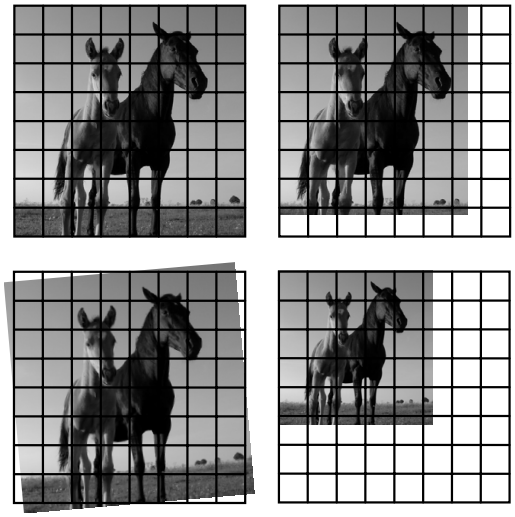

Figure 1: Fixed embedding and extraction raster at first generation watermarking schemes.

In addition to template-based approaches, there are lots of other first generation methods that embed information in a domain which is theoretically invariant to geometric attacks. These methods use, for example, the Fourier-Mellin-Transform, Zernike-Moments or the Radon-Transform. A detailed overview can be found in [20]. But still very often, cropping is a big problem for these so-called RST-invariant methods [1].

In this work, we describe a new second generation watermarking method where no fixed raster is used at all. The positions used for watermarking only depend on the image content. For direct embedding into content-dependend features we use texels or texture elements as the fundamental unit of texture space, known from computer graphics or biometric identification. We show that this kind of description of image content is very robust against a wide range of signal processing operations. At the same time, changing these texel features for watermark embedding is perceptually invisible due to its very high masking effect.

First, in section 2, we count and classify some other second generation watermarking approaches for still images to show which direction our new method follows. Afterwards, in sections 3 and 4, we propose our texel-based method in detail and discuss the problem of insertions/deletions that all second generation watermarking schemes are in trouble with. The solution presented in this paper can easily be adapted to other schemes and yields great performance improvements as demonstrated in section 5.

2. FEATURE-BASED WATERMARKING

As already stated, second generation watermarking methods can be classified into two sections, methods using features for synchronization and methods that directly embed the watermark information into features.

Kutter *et al.* formulated some properties that a feature used for watermarking should have [7]. In addition to robustness against compression, multiplicative and additive noise, a suitable feature should withstand a wide range of geometrical transformations (rotation, translation, scaling, etc.). Further, it should be possible to detect the same (remaining) feature points at the same positions if the host image has been cropped.

2.1 Using Features for Synchronization

In [7], a Mexican-Hat wavelet decomposition is applied to the host image for finding robust feature points. Afterwards, information is embedded periodically by using a spread spectrum approach at the Voronoi regions spanned by these feature points. Due to periodic embedding, the peaks of the autocorrelation function can be used to reconstruct simple geometric attacks. The described feature detection is theoretically invariant to rotation, translation and noise. But robustness against scaling is not given inherently. To make the system applicable, exhaustive search has to be employed for the underlying spread spectrum-based first generation watermarking. Other methods using the same idea of features for synchronization can be found in [2], [3], [9], or [18].

In another approach, Dittman *et al.* use self-spanning-patterns in [5] where a canny edge detector and afterwards a Harris-Corner-detector are applied to find corners within the edges of the host image. These corners are combined and different templates are embedded that help to re-synchronize the image in case of geometric attacks.

In other approaches, the SIFT detector based on scale-space theory or a modified circular Hough transform or Harris-Affine detector [15] are applied to get rotation-invariant circular regions for embedding a spread spectrum-based watermark for re-synchronization. Weinheimer *et al.* in [19] use a modified Harris-Corner detector for determination of the circle centres.

2.2 Using Features for Direct Embedding

Whereas there are lots of suggestions for the idea of using features only for synchronization, Kutter's second idea of direct embedding into the image features has attracted just minor attention in literature.

In [16], Solachidis *et al.* suggest to embed information in vector graphics by using Fourier descriptors, which are a compact description of closed object contours. The resulting amplitude spectrum is invariant to rotation, and after normalization it is also invariant to translation and scaling of the objects.

Another idea given by Maes and van Overveld in [10] is the embedding of a watermark by warping feature points determined from the host image in pixel domain as close as possible to fix points of a grid pattern. Their scheme is able to only hide one single bit, in contrast to the method presented by Pröfrock *et al.* in [12], which uses so called gravity centers of pixel blocks with different size but same amount of image content. But both papers do not demonstrate robustness against geometric attacks.

3. DIRECT EMBEDDING USING TEXELS

Our new method follows the idea of direct embedding information into image features. We continue the approach of Kutter *et al.* using a Mexican-Hat decomposition of the host image. But we extend the decomposition in a way that it is a description of the image in texture space.

The major problem when we want to use texture as feature for embedding is its scale. For example, a raw texture viewed

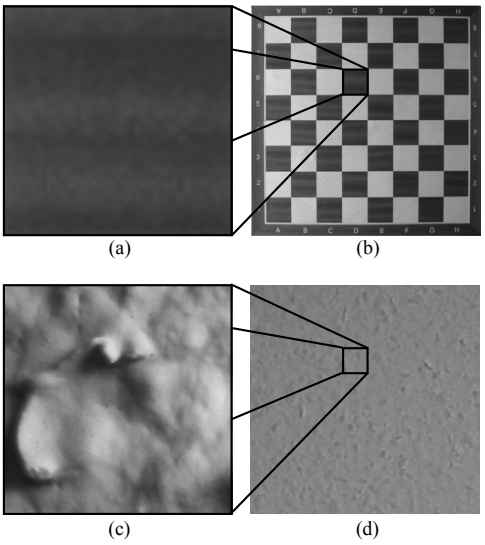

Figure 2: Example for the effect of a changed size of viewport. (a) near and (b) far view of a checkerboard, (c) near and (d) far view of a wallpaper.

from very close has completely other characteristics than the same texture viewed from far away, as can be seen in Fig. 2. That means to use texture as a feature for watermarking the texture description (detector) has to be scale-invariant.

3.1 Blobs - Features in Gaussian Scale-Space

The gray-level blob is a texture element that can be found in most images. It is a raise or decrease of gray pixel values similar to a 2D-Gaussian curve, detectable in the Gaussian scale-space by the scale-invariant detector proposed in [8]. Due to this scale-space approach the detector is theoretically scale-invariant.

To find gray-level blobs the image I_{orig} is filtered in Gaussian scale-space using so-called LOG-filter masks, where the scale is parameterized by the value $\sigma_r = \{\sigma_r \in \mathbb{R} : \sigma_{min} \leq \sigma_r \leq \sigma_{max}, r \in \mathbb{N} : r \leq R, R \in \mathbb{N}\}$ as in Eq. (1). The abbreviation LOG stands for Laplacian of the Gaussian, where the filter kernel is created by applying the Laplace operator to Gaussian functions, normalized to zero mean. Because the final kernel structure is similar to a Mexican sombrero, this filter is also known as Mexican-Hat. The resulting, e.g., $R = 16$, matrices are normalized using Eq. (3) to get scale invariance for the magnitudes of the filtering result. The search space for σ_r has to be limited to σ_{min} and σ_{max} to get a trade-off between embedding capacity, robustness and computational effort. During extraction, the scales will be normalized depending on the image size, as described in sub-section 3.3, to find the same blobs in a scaled image.

$$LOG\left(x, y, \sigma_r\right) = \left(\frac{x^2 + y^2}{2 \cdot \pi \cdot \sigma_r^6} - \frac{1}{\pi \cdot \sigma_r^4}\right) \cdot \exp^{\frac{\left(x^2+y^2\right)}{2 \cdot \sigma_r^2}} \quad (1)$$

$$I_{LOG}\left(\sigma_r\right) = LOG\left(\sigma_r\right) * I_{orig} \quad (2)$$

$$I^*_{LOG}\left(\sigma_r\right) = \sigma_r \cdot I_{LOG}\left(\sigma_r\right) \quad (3)$$

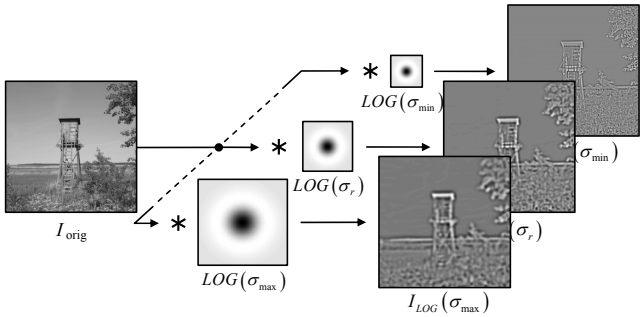

Figure 3: Image filtered using LOG-filter masks with different scales.

Afterwards, from all R results at every pixel position (x, y) the optimal scale $\sigma_{opt}\left(x, y\right)$ is selected that yields the biggest magnitude value $I_{LOG_{opt}}\left(x, y\right)$:

$$\sigma_{opt}\left(x, y\right) = \arg\max_{\sigma_r} \left|I^*_{LOG}\left(x, y, \sigma_r\right)\right| \quad (4)$$

The corresponding magnitude $I_{LOG_{opt}}\left(x, y\right)$ is stored for this position, too. Fig. 4 shows the result for an example image.

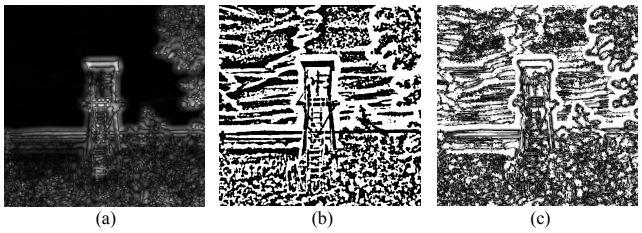

Figure 4: Final selection result for the filtered image: (a) Magnitude of $I_{LOG_{opt}}$, (b) sign of $I_{LOG_{opt}}$, and (c) the scale σ_{opt}.

This blob detection is theoretically invariant to scaling, rotation, translation, and horizontal/vertical mirroring of the image. Further, except for (gray-value pixel range [0...255]) clipping, also changes of luminance should be accepted.

To further reach robustness against contrast enhancement we employ the *rational dither modulation* (RDM), proposed by Pérez-González *et al.* in [11], to embed the watermark information as explained below.

3.2 Embedding by Texel Quantization

First, we choose the value $I_{LOG_{opt}}\left(x, y\right)$ with the biggest magnitude, which we call reference blob B_0. The position of this reference point is necessary for the determination of the watermark embedding order.

After the position of the reference point has been specified, in a successive selection process the algorithm looks for M more (largest) blobs, smaller than the reference blob. The list of chosen blobs is $B = \{B_i\left(x_i, y_i\right) : i = 0, 1, ..., M\}$, where (x_i, y_i) specifies a blob position. Thereby, around every single blob a circle is marked as "reserved region", where no other blob can be chosen from (see Fig. 5). The size of this circle is determined from the size of the corresponding scale $\sigma_{opt}\left(x, y\right)$. It is also possible to choose a value smaller than this scale during embedding and hence to allow blob overlapping. But in that case, either a recursive embedding is necessary or embedding is not optimal due to interfering blobs. In other words, the following inequality must be hold for any two blobs B_i and B_j within the list of chosen coordinates, where d_{ij} is the distance between both blobs:

$$\sigma_{opt}\left(x_i, y_i\right) + \sigma_{opt}\left(x_j, y_j\right) \leq d_{ij}. \quad (5)$$

$$d_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2}. \quad (6)$$

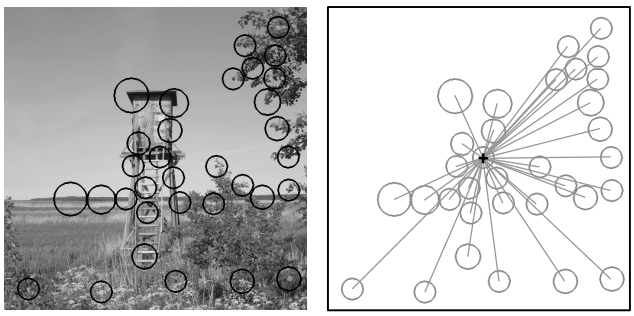

Figure 5: Example image with $M + 1 = 33$ largest blobs selected, $\sigma_{min} = 3$, $\sigma_{max} = 5$.

Now, the embedding process takes place in ascending order of distance of the M blob positions to the reference point, marked by a cross in Fig. 5. If there are changes within the order of blobs (insertions/deletions of blob positions) after cropping or due to signal processing, errors can occur. Such de-synchronization errors usually cause big numbers of burst errors at common error correction. To overcome these errors, we use the idea of insertion/deletion error correction. In section 4, the problems of blob overlapping and blob de-synchronization are discussed in detail and the applied insertion/deletion error correction is explained.

The magnitudes of the reference point as well as the other M selected blobs have to be amplified by the value G_{base} to make sure that these points will be found again during extraction. The quantization-based RDM technique for watermark embedding also requires amplification or attenuation of the magnitudes of the M watermark blobs.

So, if the magnitude of a blob has to be changed for amplification or embedding, a positive or negative LOG-mask, $LOG^*_{gain}(x, y, \sigma_{gain})$, is added to the blob. This LOG-mask can be calculated using Eq. (7) and Eq. (8), where G_{diff} is the relative amplification, the difference between new and old LOG-magnitude. The scale for LOG-computation must be $\sigma_{gain}(x, y) = \sqrt{2} \cdot \sigma_{opt}(x, y)$.

$$LOG^*_{gain}(x, y, \sigma_{gain}) = \frac{G_{diff}}{K} \cdot LOG_{gain}(x, y, \sigma_{gain}) \quad (7)$$

$$K = \sigma^2_{opt} \cdot \sum_{x=-x_{max}}^{x_{max}} \sum_{y=-y_{max}}^{y_{max}} LOG_{gain}(x, y) \cdot LOG_{opt}(x, y) \quad (8)$$

Finally, for every blob position the normalized LOG-mask is added to the image. Fig. 6 shows a watermarked image and the difference to the original image. For a zoomed and not intensified view see Fig. 17.

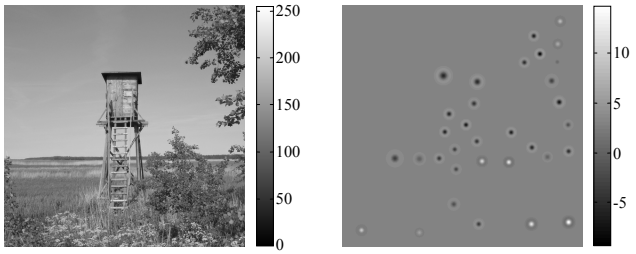

Figure 6: Watermarked image and difference to the original (intensified for visualization), $M = 32$, $\sigma_{min} = 3$, $\sigma_{max} = 5$, $P = 80$, PSNR = 55.119 dB.

For watermark embedding we use scalar rational dither modulation [11], a modified version of *dither modulation*, which is a low-complexity realization of *quantization index modulation* (QIM) [4]. The advantage of RDM over common dither modulation is its robustness against value-metric scaling. In other words, if the magnitudes of the used blobs are scaled by a constant value ρ, as in the case of contrast changes, the decoder should be still able to extract the watermark.

Data is embedded by quantizing the magnitude of a blob B_j to the closest quantization lattice point of one of two subsets of lattices, $\Lambda_{b_j} = \Delta\mathbb{Z} + b_j\Delta/4$, where $b_j = \{-1, 1\}$ is the information symbol to be embedded in the j-th blob. The quantization step size is Δ, where a larger step size yields higher robustness of the data against watermarking attacks but at the expense of higher embedding induced distortions. The appropriate Quantizers are denoted $Q_{-1}(\cdot)$ and $Q_1(\cdot)$ and embedding is performed as $B^*_j = Q_{b_j}(B_j)$, where $j \in [1, ..., M]$.

Now, RDM means that the step size Δ is adapted to the avarage norm of the host signal. Customized to our embedding and using l_2-norm, Δ is weigthed by factor g as in Eq. (10), where g is calculated as:

$$g = \left(\frac{1}{M} \sum_{j=1}^{M} |B_j|^2 \right)^{1/2} \quad (9)$$

$$\Delta = g \cdot P/255 \quad (10)$$

Thus, the step size depends on the magnitudes of all used blobs, where P is the embedding strength parameter of our watermarking system (in the range [0...255]).

Afterwards, the reference blob is amplified by Δ to be always larger than the M magnitudes of the embedding blobs.

$$B^*_0 = B_0 + \Delta \quad (11)$$

Since G_{base} must be larger than $\Delta/2$, we can set the basic blob amplification depending on the image content to $G_{base} = \lceil g \cdot P/(2 \cdot 255 - P) \rceil$ during embedding following Eq. (10) and Eq. (11). Hence, our test parameter is P.

3.3 Watermark Extraction

Before blob detection, we adjust σ_{min} and σ_{max} depending on the size $\widetilde{\mathrm{X}}$ by $\widetilde{\mathrm{Y}}$ of the watermarked image. That means, information embedding takes place at a fixed image size X by Y, and during extraction the scales are normalized to $\sigma_{min} = S \cdot \sigma_{min}$ and $\sigma_{max} = S \cdot \sigma_{max}$, where the scaling factor $S = \max\left(\widetilde{\mathrm{X}}/\mathrm{X}; \widetilde{\mathrm{Y}}/\mathrm{Y}\right)$.

Again $M + 1$ blobs with the biggest magnitudes are determined using the above blob detector. The largest value represents the received reference blob $\widetilde{B}_0$. The M received blobs are denoted as $\widetilde{B}_j$, responsible for the calculation of Δ at the receiver, following Eq. (12) and Eq. (13).

$$\widetilde{g} = \left(\frac{1}{M} \sum_{j=1}^{M} \left|\widetilde{B}_j\right|^2 \right)^{1/2} \quad (12)$$

$$\Delta = \widetilde{g} \cdot P/255 \quad (13)$$

Because of soft-decision decoding, the embedded information is not retrieved until error correction decoding.

3.4 Properties of Gray-Level Blobs

Before we carry out performance tests for the overall watermark embedding scheme, in this sub-section, we answer the question of how robust is a single blob detection. This makes sense since the effects of overlapping blobs and desynchronization (if no blob insertion/deletion error correction is applied) on the final bit error rate can be enormous.

To illustrate the properties of the proposed blob detection for a set of 60 images we determined one largest blob per image, B_0, applied a set of below shown attacks and analyzed the effects. The scales are $\sigma_{min} = 3$ and $\sigma_{max} = 5$.

First, in Fig. 7, we show the effects in terms of mean-absolute-error or peak-signal-to-noise-ratio if the magnitude of one blob is changed by the value G_{diff}. These effects are independent from the scale of the appropriate blob. In this sub-section, for easier comparison of the effects of attacks and below visualized distortions we draw the LOG-difference D as attack induced distortions. That means, for example, if a LOG-value 180 has been changed to the value 188 due to an attack, then $|D| = 8$, and, hence, to make the blob-detection robust against this attack, during quantization, the magnitude must be changed so that $|D| < |G_{diff}|$.

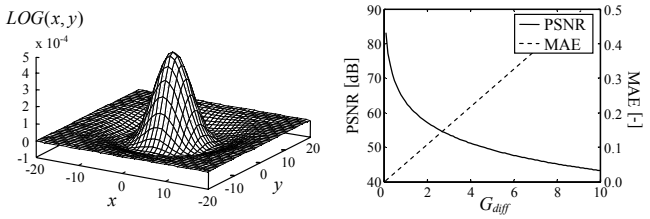

Figure 7: Inverse LOG-function, $\sigma_r = 5$ (left). Distortion due to single blob amplification independent from scale (right): Peak-signal-to-noise-ratio (PSNR) and mean-absolute-error (MAE).

A robust watermarking solution should be able to resist a wide set of signal processing operations, such as lossy image compression (e.g., JPEG, JPEG2000), luminance as well as contrast enhancement, low-pass filtering, sharpening, or noise adding. Due to the properties of the blob detection our approach is also tolerant to rotation, scaling, translation, cropping and mirroring of the image. Even slight skewing or local geometric warping are accepted by the detector.

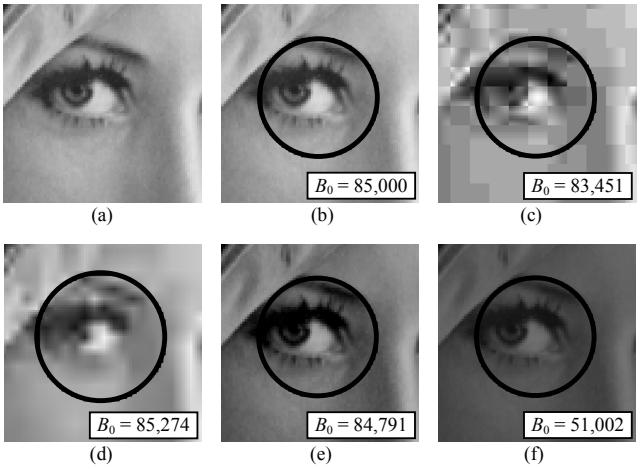

Figure 8: Example image with largest blob. (a) original, (b) image with changed blob $B_0 = 85$, marked image and detected blob after (c) JPEG compression using quality factor $QF = 5$, (d) JPEG2000 compression using target rate $r = 1/100$, (e) luminance change, and (f) contrast change.

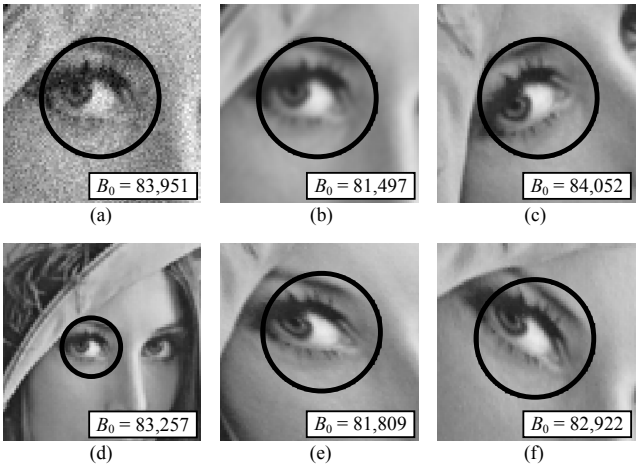

Figure 9: Marked image and detected blob after (a) additive Gaussian noise with $\sigma = 0.004$, (b) Gaussian low-pass filtering with mask size 3x3 and $\sigma = 2$, (c) rotation by $36°$, (d) scaling by factor 0.5, (e) horizontal skewing by 50%, (f) vertical skewing by 50%.

In Fig. 8 and Fig. 9, some examples are shown for the set of attacks we apply to the watermarked images. The blob with the biggest magnitude, B_0, is amplified by $G_{base} = 15$ and rounded to the value 85. Using these parameters the algorithm is able to detect this blob at the same position (marked by a circle) after all listed attacks.

Since the following figures show curves for different basic blob amplifications (without data embedding) one can see that basic blob amplification is necessary. Otherwise, if $G_{base} = 0$ and if not the same blob is found during detection we obtain outlier results, where G_{base} denotes the basic amplification, and D is the attack induced distortion.

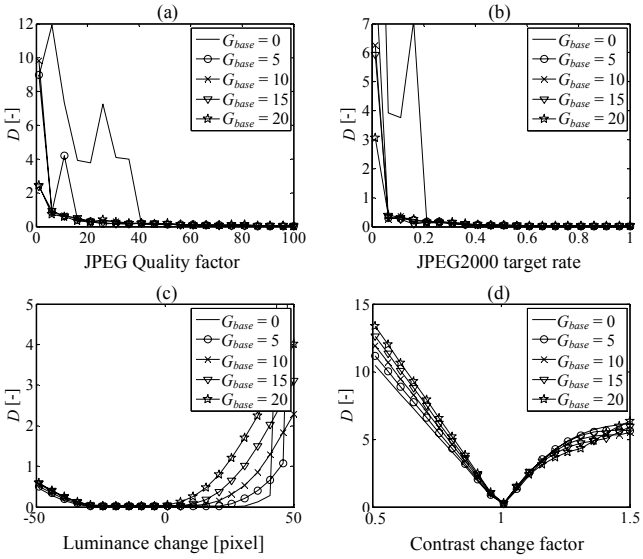

Figure 10: Results of single blob robustness test: (a) JPEG compression, (b) JPEG2000 compression [target rate 0...1] (*JasPer*-codec),(c) luminance changes, (d) contrast changes.

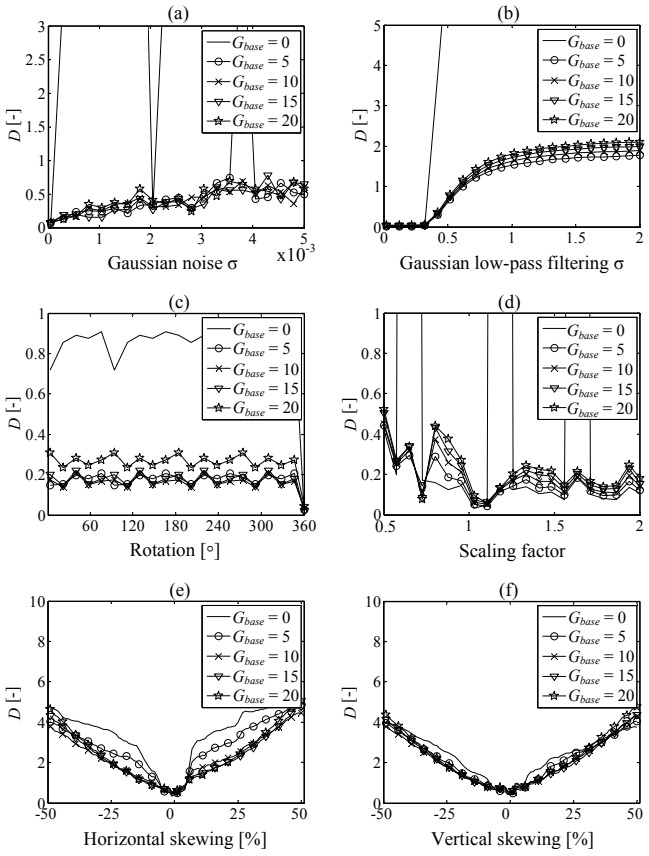

Figure 11: Results of single blob robustness test: (a) Gaussian noise, (b) Gaussian low-pass filtering (mask size 3x3), (c) image rotation, (d) scaling, (e) horizontal skewing, and (f) vertical skewing.

As can be seen in Fig. 10(d), the change of blob magnitude linearly depends on the change of contrast. For that reason, we use rational dither modulation, which scales the quantization step size (Eq. (13)), and thus, adjusts the interpretation of the M magnitude values to the contrast of the overall image.

3.5 Security Aspects

Since an attacker maybe has knowledge about the watermarking scheme, he could find and change the used blobs. Hence, to prevent easy manipulations blob detection as well as quantization must be secured by a key. The later can usually be handled by the application of a pseudorandom dither signal which randomizes the quantization index modualtion as proposed in [4] or [6].

To also randomize the process of blob selection we suggest to make the scales σ_{min}, σ_{max}, and further, the minimal distance d_{ij} between two blobs B_i and B_j depending on a secret key. Hence, the blob selection process becomes kind of chaotic following the condition of blob overlapping from Eq. (5) and Eq. (6). That means, if embedder as well as receiver apply the same key, the blob selection processes are equal. But for an attacker not knowing the key it is difficult to find the used blobs. For example, Fig. 12 demonstrates three further blob selection results due to slightly changed parameters for the image from Fig. 5.

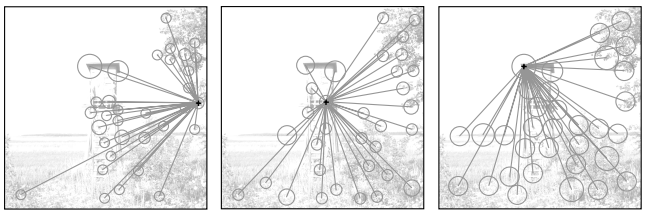

Figure 12: Slightly changed scales σ_{min}, σ_{max} and distances d_{ij} resulting in further possible blob selections as opposed to Fig. 5.

Usually, blobs with large magnitude values are associated with texture elements of smaller blob scales in natural images. That means, the major function of σ_{max} is in fact to limit the computational effort. But, if the range $[\sigma_{min}...\sigma_{max}]$ is chosen to small, the watermarking approach is more sensitive to cropping followed by scaling attack. Further, if σ_{min} is chosen to small, the overall robustness against attacks is lower. During our tests, we found out $\sigma_{min} = 3$ and $\sigma_{max} = 5$ are a good trade-off between robustness, computational effort, and security in terms of variations of blob selections.

4. BLOB INSERTION/DELETION ERROR CORRECTION

As mentioned in sub-section 3.2, the blobs for embedding are selected successively, first the one with the biggest magnitude (reference blob) and afterwards M more (largest) blobs smaller than the reference blob. During this selection process around every blob B_i an $\sigma_{opt}\,(x_i, y_i)$-sized circle is marked as "reserved region", where no other blob can be chosen from.

When signal processing takes place or rarely even after embedding due to rounding or clipping the blobs slightly change their positions. Two cases can occur with enormous effects on the successive blob selection process during watermark extraction. First, two blobs, although the inequality from Eq. (5) holds during embedding, can change their positions in a way that B_j would overlap the already chosen blob B_i during the selection process at watermark extraction. This case would yield the blob B_j not to be selected, where we talk about a deletion. Also demonstrated in Fig. 13, there is another case, known as insertion, if Eq. (5) was not fulfilled during embedding but during extraction. Here, a blob would be selected although no watermark was embedded originally.

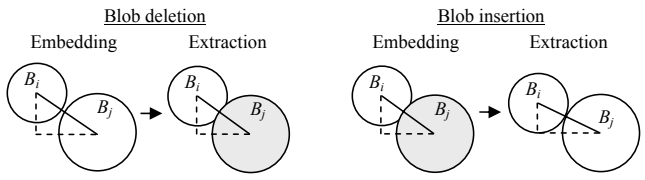

Figure 13: Blob deletion (left) or blob insertion (right) during the successive selection process at watermark extraction. The shaded blob would not have been selected.

As a consequence of such a deletion or insertion of a blob during extraction two effects can occur. First, the order

of all successively extracted data bits could be increased or decreased resulting in a non-linear de-synchronization problem within the watermark sequence and burst errors during common error correction decoding. Second, the selection process of all successive blobs could be affected resulting in not only bit de-synchronization but a completely disturbed or jumbled watermark signal. Both problems are analyzed in the following sub-section.

4.1 Consequences of an Insertion/Deletion

Fig. 14 shows what happens to the successive blob selection if a blob is inserted or deleted. The order of all successive blobs is increased or decreased respectively by one and hence the watermark bit sequence is de-synchronized. From the position of de-synchronization on, watermark information can not be extracted correctly if no re-synchronization is applied.

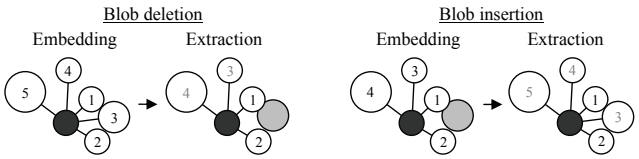

Figure 14: Blob deletion (left) or blob insertion (right) during watermark extraction. As a consequence the order of successive blobs is decreased or increased respectively.

If there is an insertion/deletion of a blob and if some blobs are very close to each other further disturbances of the blob selection process can occur. For example, the shaded blob in Fig. 15(a) that would not have been selected during embedding suddenly is selected during extraction, because Eq. (5) now is fulfilled. Its magnitude is larger, and as a consequence, the originally selected blob with number 2 can not be selected. In the second scenario, the inserted blob results in additional problems. Due to the very similar distance to the reference blob, compared to the third blob in the middle graphic, the order of these both blobs are changed, too.

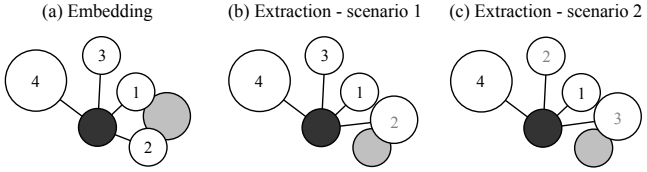

Figure 15: Blob deletion due to blob insertion during the successive selection process at watermark extraction. The shaded blob would not have been selected.

Further, if the host image is cropped, blobs can be deleted and hence the blob selection process can be also affected like in above graphics.

4.2 Overlapping Blobs During Extraction

During watermark embedding Eq. (5) must be hold or, in other words, two blobs must not overlap. But during extraction we suggest to also consider blobs that slightly overlap. Two selected blobs that are very close to each other during embedding are likely to overlap at extraction site. We define an overlapping factor $C_{ij} := \{C_{ij} \in \mathbb{R} : -\infty < C_{ij} < 1\}$ as an extension of Eq. (5) and allow this value to be larger than zero during extraction with respect to Eq. (14):

$$C_{ij} = \frac{\sigma_{opt}(x_i, y_i) + \sigma_{opt}(x_j, y_j) - d_{ij}}{\sigma_{opt}(x_i, y_i) + \sigma_{opt}(x_j, y_j)}. \qquad (14)$$

For example, if $C_{ij} = 0.01$ the blobs are allowed to overlap one per cent during extraction resulting in decreased probability of occurrence of deletions.

4.3 Re-Synchronizing Error Correction

If an overlapping of blobs is allowed during extraction there are fewer deletions due to changes of blob positions but blob deletions caused by image cropping are still possible. Further, we can not avoid the occurrence of insertions by this strategy. That means blobs that do not fulfil Eq. (5) during embedding, now, at the extraction process could be selected.

To overcome these de-synchronization problems we have to employ an error correction solution that is able to also correct insertions/deletions in addition to common substitution errors (binary: 0→1 or 1→0). Such a scheme was proposed by Solanki *et al.* in [17]. It is based on punctured channel coding and the ability of some error correcting codes to handle erasures at known symbol positions within the message. At extraction site, deletions are treated as erasures and insertions become substitution errors (see Fig. 16).

Another scheme was proposed by Schlauweg *et al.* in [13] and in [14] based on extended *dynamic programming* during FEC-decoding (forward error correction) using multiple parallel-interconnected Viterbi decoders. Each of the decoders is one bit out of sync with the others and each receives a stream containing information about the reliabilities of the received symbols. By monitoring the appropriate message paths the overall system is able to determine which is the most likely stream, and, hence the correct message.

We tested both approaches and decided to use the solution of Schlauweg *et al.*, which can be applied using the following side conditions that we formulated for our second generation watermarking scheme.

4.3.1 Input to the Decoder

First, additionally to the reference blob, we not only select M but $M + N + K$ largest blobs, where N blobs represent candidates for insertions and K blobs are candidates for deletions (see Fig. 16). For this, we introduce a user defined blob overlapping threshold $\tau \in \mathbb{R}^+$, e.g., $\tau = 0.01$. The selection criterion for the first M blobs is $C_{mj} \leq -\tau : 0 \leq m \leq M$ as well as $C_{nj} \leq 0 : 1 \leq n \leq N$. The N blobs, representing candidates for insertions, must fullfil the criterion $C_{nj} \leq 0 : 0 \leq n \leq M + N$. All other K blobs in the range $0 < C_{kj} \leq \tau : 0 \leq k \leq M + N$ are candidates for deletions.

Second, we can use the information of how likely the occurrence of an insertion/deletion is as weighting factor. That means C_{ij} can be used as a certainty of decision. If the absolute value of C_{ij} is very small, the occurrence of an insertion/deletion is likely. In contrast, if C_{ij} is large negative, two blobs are far away from each other and neither deletion nor instertion are possible. Or vice versa, if the value is positive and two blobs are strongly overlapping, then a deletion is also not very likely.

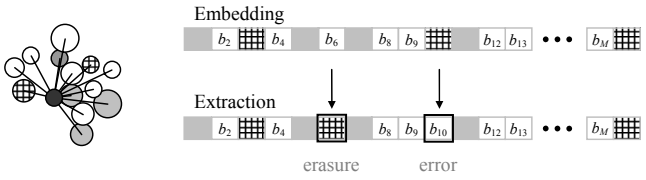

Figure 16: Insertion/deletion/substitution (IDS) error correction using erasures during watermark extraction. There are M white fix blobs, N checkered candidate blobs, and K shaded candidate blobs.

Finally, there remains one problem using either Solanki's or Schlauweg's FEC-approach. If there are blobs near the edge region of the host image, these blobs can get lost when cropped. But, as long as the blobs remain within the image our new watermarking scheme is robust against cropping.

5. RESULTS AND FURTHER RESEARCH

In several tests we examined the robustness of the proposed watermarking algorithm against numerous attacks. We used 60 different natural images of size 512x512 pixels and embedded 32 bit of random data into each of them. We used the embedding strength parameter $P = 80$. For example, embedding 32 bit results in PSNR $\approx$ 55dB at $P = 80$. But since the PSNR value is maybe not the ideal measure for distortions here, in Fig. 17, several cuttings around distorted blobs are visualized. It can be seen that the proposed blob feature has very high masking effect. Blobs can be understood as whole texture elements.

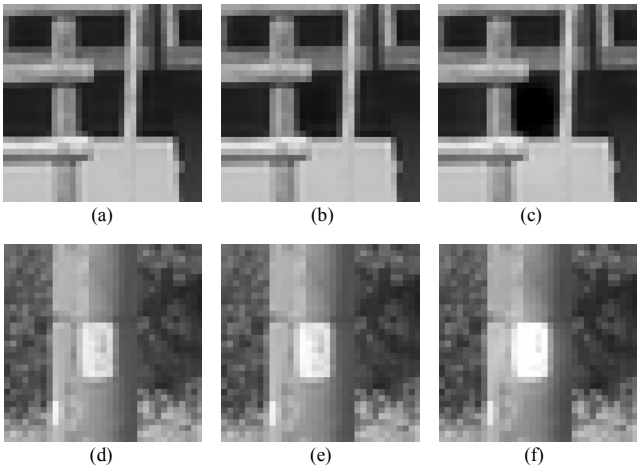

Figure 17: Two example images (3x zoomed), where (a, d) original, (b, e) blob manipulated using $\Delta = 20$, (c, f) blob manipulated using $\Delta = 70$.

To clarify the influence of blob overlapping (insertions/deletions) we tested the algorithm with and without re-synchronizing error correction. In the case of no usage of re-synchronization we simply embedded twice the number of bits and determined the resulting bit error rate for comparison. In Fig. 18, it can be seen that re-synchronization is necessary and yields strong performance improvements due to blob overlapping correction. But, although our new watermarking feature fulfils Kutter's localization property, cropping remains an open problem since the employed error correction scheme up to now is not able to handle it.

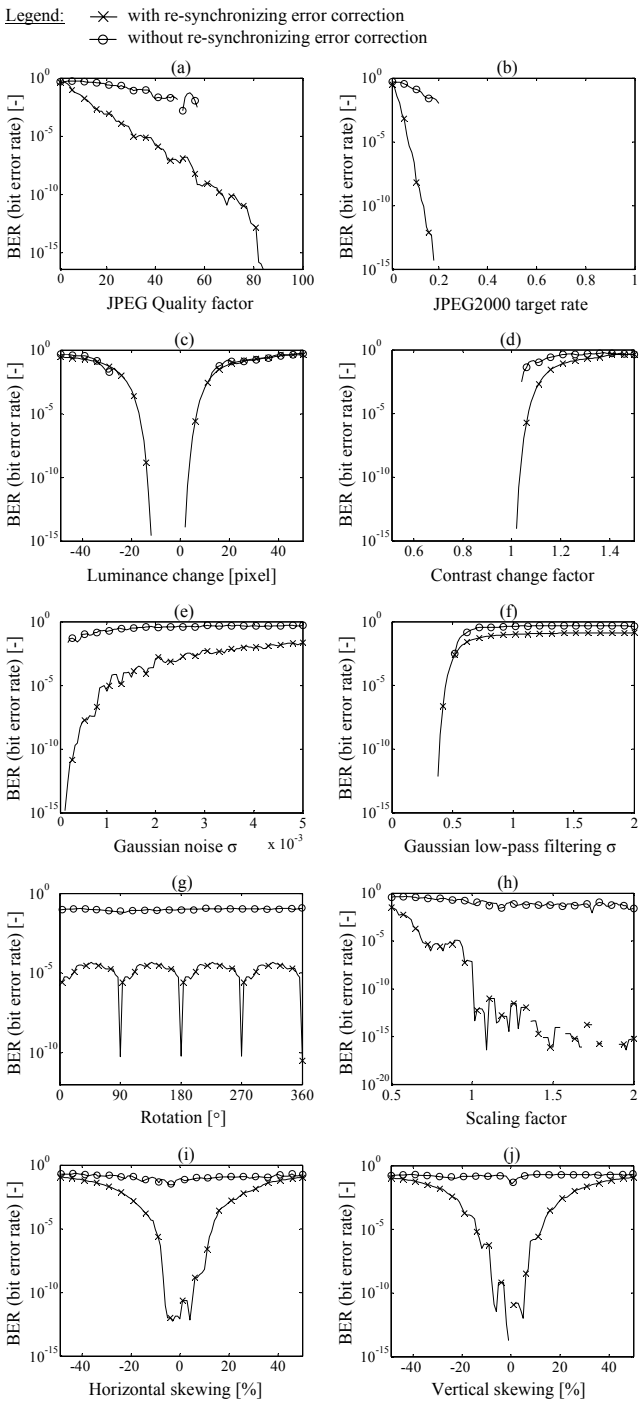

Figure 18: Results of robustness test ($\sigma_{min} = 3$, $\sigma_{max} = 5$, $P = 80$, $M = 32$): (a) JPEG compression, (b) JPEG2000 compression [target rate 0...1] (*JasPer*-codec), (c) luminance changes, (d) contrast changes, (e) Gaussian noise, (f) Gaussian low-pass filtering (mask size 3x3), (g) image rotation, (h) scaling, (i) horizontal skewing, and (j) vertical skewing.

All remaining blobs can be detected and the embedded bits

can be extracted correctly even after cropping followed by a scaling attack. But cropped blobs yield de-synchronization within the watermark sequence. The Viterbi decoder-based solution by Schlauweg *et al.* maybe can handle this kind of error after further research.

In addition to the restriction in terms of image cropping, there is a second field for further research. Although most security aspects concerning the attackers ability of finding and manipulating the used blobs have been discussed and solved, in this paper, the position of the reference blob, B_0, remains a weak point. If an attacker finds this blob, a change would affect the order of all other blobs during extraction. Maybe, one can find another criterion to define the order of embedding/extraction, which is invariant to rotation, scaling, translation, and cropping as well.

6. CONCLUSION

This paper presents a new method for embedding information in the feature space. In the proposed method, a texture-based rotation-invariant feature is used, the so-called gray-level blob in Gaussian scale-space. Due to the feature detection in scale-space, the approach is invariant to image rotation and scaling. Since embedding at a fixed raster, e.g., the pixel grid, is a big problem in all first generation watermarking approaches, our new second generation approach does not work depending on any raster. Blob selection and information embedding only depend on the image content. As a consequence and due to re-synchronization in the case of blob insertions/deletions our new method is also robust to slight image cropping and translation. Further, it is robust against lossy compression, contrast as well as luminance enhancement, filtering, and noise adding.

7. REFERENCES

[1] M. Awrangjeb, M. Murshed, and G. Lu. Global geometric distortion correction in images. In *Proc. of IEEE Workshop on Multimedia Signal Processing*, pages 435–440, Sept. 2006.

[2] P. Bas, J.-M. Chassery, and B. Macq. Robust watermarking based on the warping of pre-defined triangular patterns. In *Proc. of SPIE Security and Watermarking of Multimedia Contents II*, pages 99–109, Jan. 2000.

[3] P. Bas, J.-M. Chassery, and B. Macq. Geometrically invariant watermarking using feature points. *IEEE Transactions on Image Processing*, 11(9):1014–1028, Sept. 2002.

[4] B. Chen and G. Wornell. Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory*, 47(4):1423–1443, May 2001.

[5] J. Dittmann, T. Fiebig, and R. Steinmetz. A new approach for transformation invariant image and video watermarking in the spatial domain: Ssp - self spanning patterns. In *Proc. of SPIE Security and Watermarking of Multimedia Contants II*, pages 176–185, Jan. 2000.

[6] J. J. Eggers, R. Bäuml, R. Tzschoppe, and B. Girod. Scalar costa scheme for information embedding. *IEEE Transactions on Signal Processing*, 51(4):1003–1019, Apr. 2003.

[7] M. Kutter, S. K. Bhattacharjee, and T. Ebrahimi. Towards second generation watermarking schemes. In *Proc. of IEEE International Conference on Image Processing*, pages 320–323, Oct. 1999.

[8] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):77–116, Nov. 1998.

[9] C.-S. Lu, S.-W. Sun, and P.-C. Chang. Robust hash-based image watermarking with resistance to geometric distortions and watermark-estimation attack. In *Proc. of SPIE Security, Steganography, and Watermarking of Multimedia Contents VII*, pages 147–163, Jan. 2005.

[10] M. J. J. J. B. Maes and van C. W. A. M. Overveld. Digital watermarking by geometric warping. In *Proc. of IEEE International Conference on Image Processing*, pages 424–426, Oct. 1998.

[11] F. Pérez-González, C. Mosquera, M. Barni, and A. Abrardo. Rational dither modulation: A novel data-hiding method robust to value-matric scaling attacks. In *Proc. of IEEE Workshop on Multimedia Signal Processing*, pages 139–142, Sept. 2004.

[12] D. Pröfrock, M. Schlauweg, and E. Müller. Content-based watermarking by geometric warping and feature-based image segmentation. In *Proc. of IEEE/ACM International Conference on Signal-Image Technology & Internet-Based Systems*, pages 572–581, Dec. 2006.

[13] M. Schlauweg, D. Pröfrock, and E. Müller. Soft feature-based watermark decoding with insertion/deletion correction. In *Proc. of Information Hiding Workshop*, pages 236–250, June 2007.

[14] M. Schlauweg, D. Pröfrock, and E. Müller. Correction of insertions and deletions in selective watermarking. In *Proc. of IEEE/ACM International Conference on Signal-Image Technology & Internet-Based Systems, (to appear)*, Nov. 2008.

[15] J. Seo and C. D. Yoo. Image watermarking based on scale-space representation. In *Proc. of SPIE Security, Steganography, and Watermarking of Multimedia Contents VI*, pages 560–570, Jan. 2004.

[16] V. Solachidis, N. Nikolaidis, and I. Pitas. Watermarking polygonal lines using fourier descriptors. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1955–1958, June 2000.

[17] K. Solanki, N. Jacobsen, U. Madhow, B. S. Manjunath, and S. Chandrasekaran. Robust image-adaptive data hiding using erasures and error correction. *IEEE Transactions on Image Processing*, 13(12):1612–1639, Dec. 2004.

[18] C.-W. Tang and M.-H. Hang. A feature-based robust digital image watermarking scheme. *IEEE Transactions on Signal Processing*, 51(4):950–959, Apr. 2003.

[19] J. Weinheimer, X. Qi, and J. Qi. Towards a robust feature-based watermarking scheme. In *Proc. of IEEE International Conference on Image Processing*, pages 1401–1404, Oct. 2006.

[20] D. Zheng, Y. Liu, J. Zhao, and S. E. Saddik. A survey of rst invariant image watermarking algorithms. *ACM Computing Surveys*, 39(2), 2007.