PATCH-BASED ADAPTIVE IMAGE AND VIDEO WATERMARKING SCHEMES ON HANDHELD MOBILE DEVICES

M. N. Favorskaya [1,*], V. V. Buryachenko [1]

1 Reshetnev Siberian State University of Science and Technology, Institute of Informatics and Telecommunications, 31, Krasnoyarsky Rabochy ave., Krasnoyarsk, 660037 Russian Federation - favorskaya@sibsau.ru, buryachenko@sibsau.ru

Commission II, WG II/5

KEY WORDS: Video Watermarking Scheme, Patch-based Search, Relevant Region, Static Scene, Mobile Device

ABSTRACT:

Mobile devices provide a huge amount of multimedia information sending to the members of social groups every day. Sometimes it is required to authorize the sending information using the limited computational resources of smartphones, tablets or laptops. The hardest problem is with smartphones, which have the limited daily energy and battery life. There are two scenarios for using mobile watermarking techniques. The first scenario is to implement the embedding and extraction schemes using proxy server. In this case, the watermarking scheme does not differ from conventional techniques, including the advanced ones based on adaptive paradigms, deep learning, multi-level protection, and so on. The main issue is to hide the embedding and extracting information from the proxy server. The second scenario is to provide a pseudo-optimized algorithm respect to robustness, imperceptibility and capacity using limited mobile resources. In this paper, we develop the second approach as a light version of adaptive image and video watermarking schemes. We propose a simple approach for creating a patch-based set for watermark embedding using texture estimates in still images and texture/motion estimates in frames that are highly likely to be I-frames in MPEG notation. We embed one or more watermarks using relevant large-sized patches according to two main criteria: high texturing in still images and high texturing/non-significant motion in videos. The experimental results confirm the robustness of our approach with minimal computational costs.

1. INTRODUCTION

The rapid growth of a number of handheld mobile devices like smartphones, tablets and laptops leads to shearing multimedia information through Internet. Some information ought to be protected by authorized watermarks, visible and/or invisible, embedded in images and video sequences. Limited memory and significant battery consumption are the main challenges of watermark embedding and extraction using robust but complex modern watermarking schemes. Two scenarios with and without proxy server are available. We have interest to develop and study the watermarking algorithm without use of proxy server.

Conflicting constraints prevent the expectation of outstanding results. However, some expectations from the algorithmic solver are possible. As well-known, the robustness, imperceptibility and capacity are the main criteria of watermarking process, while the computational cost is additional criterion for mobile devices. In recent years, researchers provide a watermark security as an additional level of protection using scrambling algorithms. In this sense, the watermarking methods approximate the steganography methods.

Another problem inherent to all blind watermarking schemes, when the original image, frame or audio signal are not transmitted to a recipient through protected channels, connects with infinite number of possible attacks, one of the complete taxonomies of which was presented in (Zotin et al, 2020). Moreover, each type of attack can be done with different unknown parameters, and unknown combinations of several types of attacks can be applied. At present, there is no universal watermarking algorithm robust to the most types of attacks. As a rule, various watermarking schemes are more or less robust to the limited types of attacks.

In this paper, we propose a simple approach for creating a set of patch-based regions for watermark embedding using texture and texture/motion estimates in still images and frames that are highly likely to be I-frames in the MPEG notation.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. In Section 3, the texture analysis is presented, while the motion analysis is considered in Section 4. Section 5 discusses a technique for selecting relevant regions for watermark embedding. Experimental results are reported in Section 6, and Section 7 concludes this paper in the end.

2. RELATED WORK

At present, investigations in watermarking techniques are shifted to deep learning implementations (Hatoum et al., 2021). However, this approach is problematic in smartphone environment, and we propose an adaptive image and video watermarking approach based on texture or texture/motion analysis, which allows us to detect the relevant regions for embedding a watermark in image or frame, respectively. A wide range of digital watermarking algorithms has been investigated in order to find the efficient ones under the formulated constraints. Detecting the relevant regions for a video watermarking scheme as an extension of image watermarking scheme includes two types of analysis – texture and motion. Hereinafter, we discuss these issues.

Many previous watermarking algorithms were based on rough set theory to solve some uncertainty problems that are related to the principles of human visual system (HVS) and affect the perceptual quality of host images. Color representations, variety

of grayscale values and brightness/darkness of the host image influence on imperceptibility of embedded watermarks. At the same time, the watermarking schemes in the transform domain slightly change the magnitudes of the frequencies. This causes visual artifacts. Thus, the selection of relevant regions for embedding a watermark with respect to HVS is important. In (Ni et al., 2007), the non-overlapped blocks were analyzed by means of fractal dimension and the feature blocks containing edges and textures were further classified by variance characteristics into three different parts: edges, weak textures and strong textures. Discrete cosine transform was applied to all blocks of the host image, and the formed watermark was embedded into their middle-frequency coefficients with different strength. A joint spatial texture analysis for robust watermarking technique to authenticate images was suggested in (Ghadi et al., 2017). The model used four features including skewness, kurtosis, entropy, and DC coefficient to analyze the texture in each partitioned block in the host image. The ranking all partitioned blocks based on their texture magnitude was implemented using technique for order preference by similarity to ideal solution. For embedding a watermark, 10% of highly textured blocks were selected. In (Favorskaya et al., 2017), the statistical and model-based methods were investigated as a trade-off between the computational costs and quality of the detected regions, where the embedded watermark might be most invisible for a human vision. It was shown that the gradient oriented local binary patterns (LBPs) provided better computational time with respect to fractal estimations. In (Ghadi et al., 2019), in order to enhance the imperceptibility and the robustness, four gray-scale histogram based-image features (DC, skewness, kurtosis, and entropy) were chosen as input data for designing association rules based on the Apriori algorithm. As a result, the ratios of imperceptibility, robustness and embedding rate with low execution time were obtained.

Estimating optical flow as the pixel-level motions is a fundamental problem in computer vision. It is worth noting that optical flow evaluation is realized through supervised and unsupervised deep learning (Ren et al., 2020) and, moreover, some approaches are based not on pixel-level motions but on patch-level motions. Traditionally, optical flow model is optimized under the assumption of brightness constancy and local smoothness constraint that usually requires small displacement between compared frames. The case with large displacements can be solved using the coarse-to-fine warping technique (Amiaz et al., 2007), the patch-based descriptor matching into the variational model (Brox and Malik, 2010), patch-match correspondence algorithm as an efficient approximate algorithm for finding the nearest neighbors of image patches between two related images (Barnes et al., 2010), and so on. In these studies, it was shown that patch-based correspondences are more robust and reliable but time-consuming regarding the pixel-based approach.

The original PatchMatch algorithm (PMA) estimates dense approximate nearest neighbor correspondences between patches of two image regions. The generalized PatchMatch correspondence algorithm was enhanced in three ways (Barnes et al., 2010): to find *k*-nearest neighbors, to search the patches across scales, rotations and translations and to match the patches using arbitrary descriptors and distances. The PatchMatch algorithm is successfully used as an approximate nearest neighbor algorithm on top of the learned descriptors after using Siamese CNN for optical flow computation (Gadot and Wolf, 2016).

Although block matching algorithm (BMA) based on full search was suggested in 1990s, its modifications are often used for motion estimation and video coding. There are three ways to improve the original BMA. The first way is to decrease the computational complexity of BMA (Huang et al., 2006):

1. Using a fixed pattern. The three step search (TSS), the simple and efficient TSS, the four step search and the diamond search are algorithmically considered as the fastest, but they cannot match the dynamic motion content.
2. Reducing the search points. This category includes the adaptive rood pattern search, the fast block matching using prediction, the block-based gradient descent search and the neighborhood elimination algorithm. These methods assume that the error-function behaves monotonically and well evaluate slow movement.
3. Decreasing the computational overhead for every search point. The matching cost is replaced by a partial or a simplified version under assumption that all pixels within each block move by the same finite distance. The new pixel-decimation, the efficient block matching using multilevel intra, the inter-sub-blocks and the successive elimination algorithm are in this category. Such methods are not immune to noise or illumination changes.

The second way is based on spatio-temporal correlation using neighboring blocks in the spatial and temporal domains in order to predict movements (Nisar et al., 2012). Such algorithms are able to avoid local minima, predicting the search center closer to the global minimum. However, the enhanced predictive zonal search and UMHexagonS algorithm cannot correctly detect motion of very small objects. The third way is to use evolutionary approaches such as the light-weight genetic block matching, the genetic four-step search and the particle swarm optimization block matching (Cuevas et al., 2013). However, the evolutionary approaches are characterized by large computational time.

3. TEXTURE ANALYSIS

The texture classification, texture segmentation, texture synthesis, and texture shape are the main issues of texture analysis in spatial domain. Our task is close to texture segmentation under criterion of high texturing. We can enforce texture segmentation by saliency analysis, removing regions which attract a human attention. Generally, methods of texture analysis are usually classified into four categories: statistical, structural, model-based and transform-based methods (Armi and Fekri-Ershad, 2019). Statistical methods are based on the moments of distribution functions of pixels' intensities. Calculation of statistical moments is one of the simplest approaches to evaluate the texture features. Central moment of order *n* is calculated using following equations:

$$\mu_n(z) = \sum_{i=0}^{L-1} (z_i - m)^n \, p(z_i) \tag{1}$$

$$m = \sum_{i=0}^{L-1} z_i \, p(z_i) \tag{2}$$

where z_i = the random intensity value

$p(z_i)$ = the number of pixels that have intensity values equal to z_i

L = the number of intensity levels, $L > 1$

To estimate a degree of high texturing, we use three measures of homogeneity U

$$U = \sum_{i=0}^{L-1} p^2(z_i) \tag{3}$$

smoothness R

$$R = 1 - \frac{1}{1 + \sum_{i=0}^{L-1} (z_i - m)^2 \, p(z_i) \Big/ (L-1)^2} \tag{4}$$

and entropy E

$$E = -\sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i) \tag{5}$$

Also three modified texture features as normalized homogeinity U_n, relative smoothness R_m and normalized entropy E_n can be used:

$$U_n = U / \log_2 L \tag{6}$$

$$R_m = \begin{cases} -\log R & \text{if} \quad R > 0 \\ 10 & \text{if} \quad R = 0 \end{cases} \tag{7}$$

$$E_n = E / \log_2 L \tag{8}$$

If parameter $R = 0$, then we forcibly maintain a relative smoothness $R_m = 10$ (small empirical value differing from 0). Normalized entropy E_n indicates some equalization effect in dark and bright areas of an image. We can speak about high texturing region if its homogeneity $U_n \to 1$, smoothness $R_m \to 0$ and $En \to 1$.

Another possible approach is to estimate a degree of high texturing using a special type of LBPs in a manner proposed in (Favorskaya and Buryachenko, 2020a). The LBP provides a technique of a unique encoding of the central pixel with position c regarding to its local pixel neighbourhood (number of neighbours P) using a predefined radius value R. A uniformity measure U shows the number of bitwise 0/1 and 1/0 transitions in LBP. The LBP is considered as a non-uniformed if $U > 2$. In order to extract the neighbour's gray values, a rotation-invariant variance measure VAR was introduced in (Ojala et al., 2002):

$$VAR_{P,R} = \frac{1}{P} \sum_{p=0}^{P-1} \left(g_p - \mu \right)^2 \tag{9}$$

$$\mu = \frac{1}{P} \sum_{p=0}^{P-1} g_p \tag{10}$$

where $g(\cdot) \in [0 \ldots 255]$ = the gray-scale value of pixel

Thus, the gradient magnitude can be estimated in a following manner:

$$G\left(LBP_{P,R}\right) = \begin{cases} \sum_{r=R_1}^{R_n} \sqrt{VAR_{P,r}} & \text{if } LBP_{P,R} \text{ is uniform} \\ 0 & \text{else} \end{cases} \tag{11}$$

In this study, we compare the results of texture estimation based on moments and LBPs in terms of smartphone performance. Images received from the cameras of modern smartphone are high-resolution, while the screen resolution is not so high.

Adaptive watermarking algorithm detects the best regions for embedding in the host image as a set of patches. Unfortunately, a set of patches does not usually form a regular structure in an image. We need to find the close patches joining them into a larger structure, which coordinates include in a secret key. We estimate a degree of high texturing at different scales 32×32, 16×16 and 8×8. The embedding and extraction algorithms are based on discrete wavelet transform and the Cox-Zhao scheme.

4. MOTION ANALYSIS

In video watermarking schemes, the task is extended by embedding a watermark into frames of video sequence. Due to limited computational resources, it is difficult to develop a system that is invariant to MPEG noise quantization and MPEG compression. However, we can examine a set of frames on the candidates of I-frame that are fully transmitted with respect to B-frames and P-frames (Favorskaya and Buryachenko, 2020b). To reduce computational costs, we propose to find motion in a scene, first, using rough algorithms (for example, background subtraction) and, second, estimating moving regions applying BMA and/or PatchMatch algorithm (Barnes et al., 2010). The matter is that local search using BMA can only identify small displacements, while large arbitrary motions of small objects can be estimated using PMA but with less smoothness.

There are two advantages for solving the watermark problem. We do not need to estimate motion in a static scene accurately because the moving regions will be removed from the set of relevant regions for embedding a watermark. Also we do not need to do additional salient analysis because usually moving objects are salient objects, and such salient regions should also be removed from the set of relevant regions. Embedding a watermark in each of the consecutive frames is not critical because smartphone videos are usually short.

In BMA, the consecutive frames are divided into blocks. For each block in the current frame, BMA calculates the best matching block inside a region of the previous or following frame, aiming to minimize the sum of absolute differences (SAD), sum of squared differences (SSD) or mean of squared differences (MSD):

$$SAD\left(d_x, d_y\right) = \sum_{x=1}^{n} \sum_{y=1}^{n} \left| I_{t+1}(x,y) - I_t\left(x + d_x, y + d_y\right) \right| \tag{12}$$

$$SSD\left(d_x, d_y\right) = \sum_{x=1}^{n} \sum_{y=1}^{n} \left(I_{t+1}(x,y) - I_t\left(x + d_x, y + d_y\right) \right)^2 \tag{13}$$

$$MSD\left(d_x, d_y\right) = \frac{1}{n \times n} \sum_{x=1}^{n} \sum_{y=1}^{n} \left(I_{t+1}(x,y) - I_t\left(x + d_x, y + d_y\right) \right)^2 \tag{14}$$

where (d_x, d_y) = the displacement vector
$I(x, y)$ = the intensity of pixel with coordinates (x, y)
n = the size of the analyzed block

In our experiments, we applied the simple and efficient TSS modification for relatively slow movement of large areas. In order to evaluate small moving areas, we offer the PMA modification.

The original PMA is based on a nonparametric patch sampling method which executes a repeated search of all given patches in one image for the most similar patch in another image under assumption that both images are close (i.e. stereo images or consecutive frames. The initial procedure fills the nearest-neighbor field (NNF) with either random offsets or prior information. The iterative update procedure propagates good patch offsets in NNF to adjacent pixels and then randomly searches the best offset in the neighborhood.

We adapt the original PatchMatch algorithm exploiting the idea that the initial random procedure can be replaced by a directed search based on a coarse interframe difference. Also we can suppose that the adjacent patches are cooperatively shifted in a part of the frame. Let $\{D\}$ be a set of interframe differences in the form of patches with predefined sizes, and $\{A\}$ be a set of patches in the extended neighborhood of set $\{D\}$ in frame t. Thus, the nearest-neighbor field is initialized with set $\{A\}$, and the patch correspondences should be computed in frame t+1 using set $\{D\}$ and generating a set of corresponding patches $\{B\}$. Since frame differences are small, we can only consider the offsets, simplifying the iterative update procedure.

The iterative procedure examines the joint set $\{A\}\bigcup\{D\}$ in scan order (from left to right and top to bottom) by random search. Note that since a random search is performed in a limited joint set, the random search for next patches is reduced and can be directed into the pipeline of the previous corresponding patches. Thus, the first correspondences can be considered as a seed for the next correspondences.

During the iterative update procedure, two operations called as propagation and random search alternate at the patch level. Propagation operator can be implemented in both the original PMA and the generalized PMA, which differ in the offset values $(d_x, d_y) = \{(1, 0), (0, 1)\}$ and by collecting k nearest neighbors for each patch, respectively. Random search operator attempts to improve function $I(x, y)$ by testing a sequence of candidate offsets at an exponentially decreasing distance:

$$\mathbf{U}_s = \mathbf{U}_0 + w\alpha^s\mathbf{R}_s \qquad (15)$$

where s = the iterative step, s = 0, 1, 2,...
$\mathbf{U}_s$ = the sequence of candidate offsets at step s
$\mathbf{U}_0$ = the initial sequence of candidate offsets
w = the large maximum search "radius"
α = the fixed ratio between search window sizes
$\mathbf{R}_s$ = the uniform random in $[-1, 1] \times [-1, 1]$ at step s

In PMA, the patches are examined for s = 0, 1, 2,… until the current search radius $w\alpha^s$ is below 1 pixel, α = 1/2. It is recommended to use the halting criteria as a fixed number of iterations, no more than 4-5.

In such manner, we form a set of frames with small interframe differences, exclude all detected moving patches from each frame of this set and create a motionless map for this set. A hypothesis is that one of such frames will be I-frame.

5. SELECTING RELEVANT REGIONS

In the case of an image watermarking scheme, we only apply texture analysis, selecting highly textured regions. For video watermarking scheme, spatio-temporal analysis should be utilized. Since a video sequence is compressed by an unknown type of codec when transmitted through unprotected channels, we ought to detect a set of frames, one from which will be I-frame. To do this, first, we detect scene changes (Favorskaya and Buryachenko, 2020a), which are excluded from a watermarking process but after that a set of I-frames can be detected. Second, we apply motion and texture analysis for these frames. These processes are clarified in Figures 1 and 2.

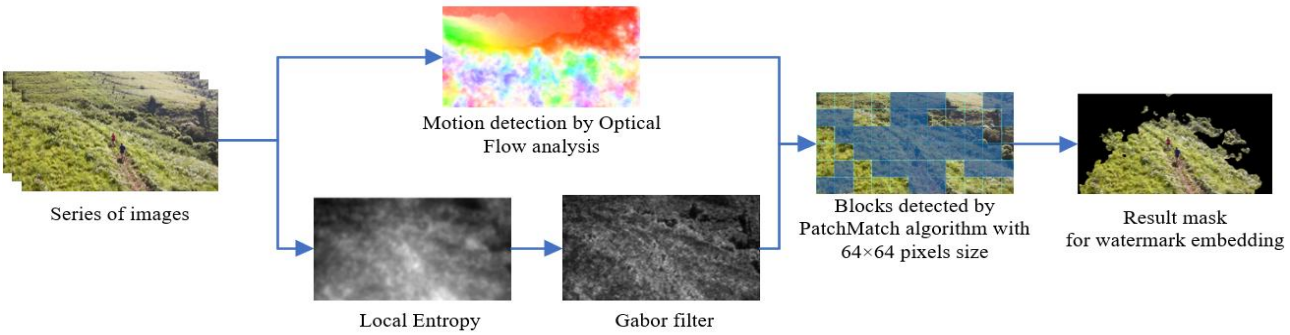

Figure 1. Selecting the relevant regions in the frame set. The main stages of the algorithm are texture analysis and motion analysis. After that, we use the PatchMatch algorithm to select the high-textured blocks with less motion. The size of blocks varies depending on the frame size and the desirable resource cost.

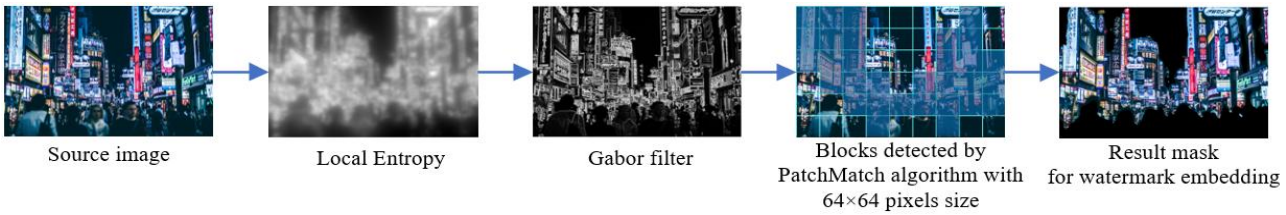

Figure 2. Selecting the relevant regions in the still image. We exclude motion analysis stage and use the PatchMatch algorithm mainly to select the high-textured blocks.

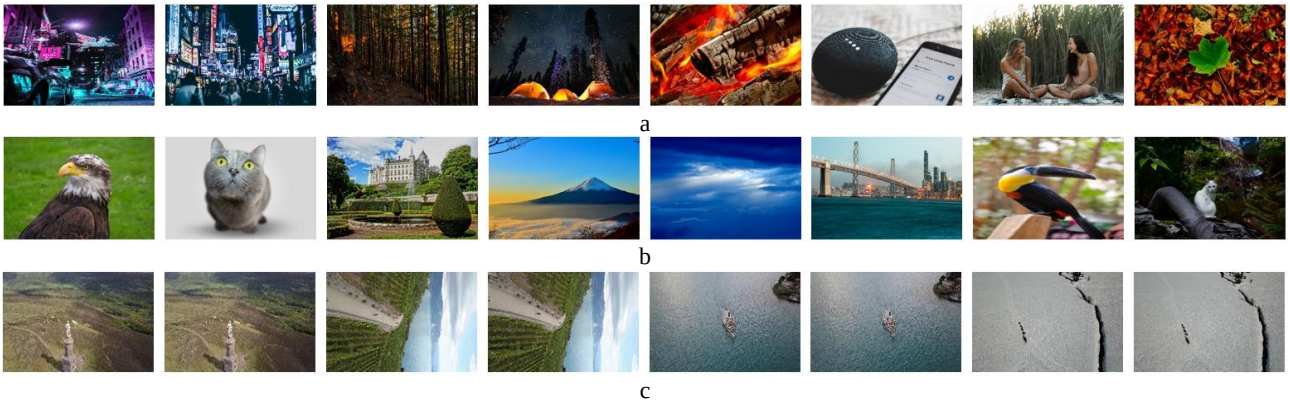

Figure 3. Datasets for embedding watermarks: a images from the CLIK2019Professonal dataset, b images from the Felicepollano watermarked/not-watermarked-images dataset, c frames from the Drone_Videos dataset.

Image	Image resolution, pixels	Size of PatchMatch, pixels	Time of embedding algorithm, ms	Information capacity of a watermark, kb	Watermarked image quality, PSNR	Textural complexity
0d870f5fa97.jpg	2048×1365	8×8 16×16 32×32	1.151 0.985 0.881	21.4 34.7 56.3	34.81 34.78 34.79	Low
1c09b4c54e.jpg	2048×1365	8×8 16×16 32×32	1.192 0.980 0.899	16.2 25.6 31.0	38.51 38.48 38.59	Medium
whdof34n21.jpg	2048×1365	8×8 16×16 32×32	1.238 0.976 0.875	11.1 16.2 20.8	36.10 35.96 35.94	High

Table 1. Assessment of the watermark information capacity.

To reduce computational costs, we cannot employ full salient analysis. However, we can apply a weight function to selected highly textured regions. A weight function assigns weight values from the interval [0…1] to the selected highly textured regions according to their remoteness from a center of the image. The weight function can be uniformly or elliptically distributed function, depending on the content of the image. Thus, we consider possible salient objects which are usually located in the center of the image in order to avoid embedding a watermark in such visual regions.

6. EXPERIMENTAL RESULTS

We tested several hundred of images and dozens of videos received from smartphone cameras, first, by selecting the relevant regions and, second, by embedding watermarks in the form of small images or short text messages. Experiments show that adaptive watermarking schemes provide better robustness and invisibility than conventional watermarking schemes.

The experiments were conducted using several datasets. The CLIK2019Professonal dataset (CLIC 2019, 2021) contains high quality images, including complex textures and foreground objects, which allows us to evaluate the loss of quality when embedding watermarks. The dataset, namely Felicepollano watermarked/not-watermarked-images (Pollano, 2021), includes data for training the algorithm with different types of the watermarks. Most images contain different levels of texturing and vary in quality and resolution. We also used high quality video sequences obtained from the Drone_Videos dataset (Drone Videos, 2021), characterized by complex camera movements and natural textures. Examples of such images are depicted in Figure 3.

The process of selecting regions for embedding watermarks includes several steps, depending on the availability of the sequence of images for texture/motion estimation, as well as the available computing resources. To estimate the level of texturing, a combination of different approaches (the first way) was used: Local Entropy, Local Standard Deviation, and Local Range, on the basis of which a texture mask was built taking into account the expansion operation. The second way was based in the LBP calculation with following classifying by the Kullback–Leibler divergence. This approach is the fastest, but the most inaccurate. The third way for building a texture mask was performed using Gabor filters and required 2-4 times more for the same image resolution.

In the case of videos, motion is evaluated in several consecutive frames. Initially, it is required to estimate the probability of a scene change in order to find the I-frame, which is least prone to data compression distortions during encoding and transmission over unprotected channels. To determine the regions relevant for watermarking, we do not need a clear knowledge of the position of objects and feature points: we only need a basic understanding of the level of movement in different areas of the frame. Therefore, it is reasonable to use the BMA and the PMA. An alternative way is to estimate the optical flow, which also gives a good representation of the global movement of the scene and allows foreground objects to be detected. The examples of using the PMA and optical flow estimation are shown in Figures 1, 2 and 4.

The next step is to combine the texture mask and the global motion mask in order to exclude low-textured regions, while considering the position of pixel blocks: the closer to the center of the image, the less priority is given to this region for embedding a watermark (Figure 4).

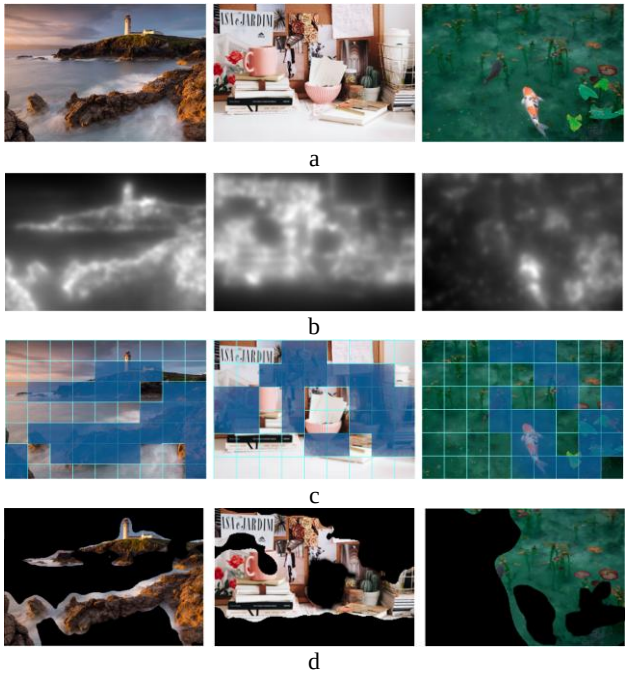

Figure 4. Examples of watermarking process of the CLIK2019Professonal dataset: a original images, b entropy masks, c patch-based motion masks with 64×64 patch size, d regions for embedding a watermark after excluding fast motion and salient objects.

We also estimated the information capacity of images when embedding a watermark depending on the size of the PMA blocks (Table 1). The quality of the watermarked images was estimated by well-known peak signal-to-noise ratio (PSNR) and mean square error (MSE) metrics:

$$PSNR = 10\log_{10}\left(\frac{\max(I(i,j))^2}{MSE}\right) \quad (16)$$

The MSE between the original and the watermarked images is calculated by the following expression:

$$MSE = \frac{1}{m \times n}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(I(i,j) - I_w(i,j)\right)^2 \quad (17)$$

where m, n = width and height of the images
I = the source frame
I_w = frame with watermark embedded

Depending on the size of the image, we can get a big difference in the amount of the embedded information. The amount of such information is tens of kilobytes for an HDTV format and increases with an increase in the block size, which is due to the fact that the block motion estimation algorithm becomes less sensitive to small objects and objects with complex texture and contour structure. This is not a disadvantage for embedding a watermark and practically does not lead to degradation of image quality.

7. CONCLUSIONS

The paper proposes an algorithm for embedding a watermark for handheld mobile devices. The main idea is to improve the speed of embedding using faster methods. We also evaluate the relevant regions for embedding a watermark based on texture/motion estimates and exclude foreground objects or visible objects closer to the center of the frame. This allows us to improve the quality of watermarked images and avoid embedding a watermark in those areas of the frame that can be attacked and distorted when transferring data over unprotected channels. Evaluation of the efficiency of the algorithm on existing databases shows high speed and robustness to distortion, regardless of the complexity of the image.

ACKNOWLEDGEMENTS

The reported study was funded by the Russian Fund for Basic Researches according to the research project no. 19-07-00047 a.

REFERENCES

Amiaz, T., Lubetzky, E., Kiryati, N., 2007. Coarse to over-fine optical flow estimation. *Pattern Recognit*., 40(9), 2496-2503.

Armi, L., Fekri-Ershad, Sh., 2019. Texture image analysis and texture classification methods - A review. *International Online Journal of Image Processing and Pattern Recognit*., 2(1), 1-29.

Barnes, C., Shechtman, E., Goldman, D.B., Finkelstein, A., 2010. The generalized PatchMatch correspondence algorithm. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) *Computer Vision – ECCV 2010. ECCV 2010*. LNCS, vol. 6313, pp. 29-43. Springer, Berlin, Heidelberg.

Brox T., Malik J., 2010. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Trans. Pattern Anal. Mach. Intell*., 33(3), 500-513.

CLIC 2019: CVPR 2019 - Workshop and Challenge on Learned Image Compression. Available at: http://clic.compression.cc/2019/ (7 March 2021).

Cuevas, E., Zaldívar, D., Pérez-Cisneros, M., Sossa, H., Osuna, V., 2013. Block matching algorithm for motion estimation based on Artificial Bee Colony (ABC). *Applied Soft Computing Journal*, 13(6), 3047-3059.

Drone Videos DJI Mavic Pro Footage in Switzerland. Available at: https://www.kaggle.com/kmader/drone-videos (7 March 2021).

Gadot, D., Wolf, L., 2016. PatchBatch: A batch augmented loss for optical flow. *CoRR ArXiv Preprint*, arXiv:1512.01815v2, 1-10.

Ghadi, M., Laouamer, L., Nana, L., Pascu, A., 2017. A joint spatial texture analysis/watermarking system for digital image authentication. In: *Proceedings of the 12th 2017 IEEE*

International Workshop on Signal Processing Systems (SiPS), pp. 1-6. IEEE, Lorient, France.

Ghadi, M., Laouamer, L., Nana, L. Pascu, A., 2019. A blind spatial domain-based image watermarking using texture analysis and association rules mining. *Multimed. Tools Appl.* 78, 15705-15750.

Hatoum, M.W., Couchot, J.F., Couturier, R., Darazi, R., 2021. Using deep learning for image watermarking attack. *Signal Processing: Image Communication*, 90, 116019.1-116019.12.

Huang, T., Chen, C., Tsai, C., Shen, C., Chen, L., 2006. Survey on block matching motion estimation algorithms and architectures with new results. *Journal of VLSI Signal Processing*, 42, 297-320.

Favorskaya, M., Pyataeva, A., Popov, A., 2017. Texture analysis in watermarking paradigms. *Procedia Computer Science*, 112, 1460-1469.

Favorskaya, M., Buryachenko, V., 2020a. Detecting relevant regions for watermark embedding in video sequences based on deep learning. In: Czarnowski, I., Howlett, R.J., Jain, L.C. (eds.) *Intelligent Decision Technologies 2020 – Proceedings of the 12th KES International Conference on Intelligent Decision Technologies (KES-IDT-20)*, SIST, Vol. 193, pp. 129-139. Springer Nature Singapore.

Favorskaya, M.N., Buryachenko, V.V., 2020b Authentication and copyright protection of videos under transmitting specifications. In: Favorskaya, M.N., Jain, L.C. (eds.) *Computer Vision in Control Systems–5*, ISRL, vol. 175, pp. 119-160. Springer, Cham.

Ni, R., Ruan, Q., Lu, J., 2007. Adaptive watermarking and performance analysis based on image content. *International Journal of Wavelets, Multiresolution and Information Processing*, 5(1), 173-185.

Nisar, H., Malik, A.S., Choi, T.-S., 2012. Content adaptive fast motion estimation based on spatio-temporal homogeneity analysis and motion classification. *Pattern Recognition Letters*, 33, 52-61.

Ojala, T, Pietikäinen, M, Mäenpää, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans Pattern Anal. Mach. Intell.*, 24(7), 971-987.

Pollano, F. A suite of images with and without a random watermark by Felice Pollano Watermarked / Not watermarked images. Available at: https://www.kaggle.com/felicepollano/watermarked-not-watermarked-images (7 March 2021).

Ren, Z., Yan, J., Yang, X., Yuille, A., Zha, H., 2020. Unsupervised learning of optical flow with patch consistency and occlusion estimation. *Pattern Recognit.*, 103, 107191.1-107191.10.

Zotin, A., Favorskaya, M., Proskurin, A., Pakhirka, A., 2020. Study of digital textual watermarking distortions under Internet attacks in high resolution videos. *Procedia Computer Science*, 176, 1261-1270.