Improved implementation of method for video-watermark detection using smart device

Takaaki Yamada
Center for Technology Innovation
Hitachi, Ltd.
Yokohama, Japan

Motoki Kamitani
Contents Solution Division
Hitachi Solutions, Ltd.
Tokyo, Japan

Haruyuki Nakamura Tsuyoshi Ohta
News Technical Center
Japan Broadcasting Corporation
Tokyo, Japan

***Abstract*—The implementation of a previously developed method for rough image analysis for detecting watermarks in displayed video content using a smart device has been improved. The computational load involved in processing a full-size image was reduced by using Gaussian filtering to analyze the complexity of the image regions. Watermarks are assumed to be embedded in those regions with higher complexity. Regions are then checked for watermarks in descending order of complexity. Quick and reliable watermark detection is achieved by adjusting the number of regions checked so that the computational load matches device-hardware performance. Evaluation demonstrated that the improved implementation can detect watermarks at 18 fps compared with 10 fps for the previous implementation.**

Keywords—smart device; video watermark; reader application

I. INTRODUCTION

A smart device with applications can provide various human-computer interactions to the user by analyzing sensor inputs, estimating the situation, and accessing corresponding network services [1]. If a smart device with a reader application is held over optically marked content, the application can read the mark and process the information it conveys. Such applications can be used, for example, to speed up the inventory process and to promote sales. In addition to visible barcode and augmented reality marks, invisible digital watermarks also can be used [2].

Digital watermarking is used to embed information in content by modifying the content slightly. Many algorithms have been developed for general-purpose image watermarking in both the pixel and frequency domains [3, 4]. Various conventional methods create watermarks with sufficient robustness against image processing during recapture [5, 6]. If a video is watermarked using such a method, the watermarks can be detected when the video is output to a display.

We are developing a watermark application that will enable services and products to be promoted along with video content distribution, as shown in Fig. 1. A person viewing the video who is interested in the product could then use a smart device equipped with our reader application to capture frame images. The application would then be used to detect the watermarks in a frame image and extracts the ID number. The reader application could then access the corresponding information by using the number. This watermark detection requires geometrical correction of an image captured with a hand-held camera. Methods for efficient geometrical correction include ones that use visible marks [7] and ones that do not use them [8]. Although visible marks such as barcodes in a watermarked image can help with geometrical correction, they create image noise. We thus use a full-search method [8] without visible marks in our application, which means that the computational load is heavier.

One way to reduce the load is to quickly focus on those regions in the image where watermarks are likely detectable. Since watermarks are assumed to be embedded strongly in complex regions, analyzing the complexity of the regions comprising a camera image is an effective method for focusing on the detectable regions. Although features obtained with conventional methods can be used for identifying image features [9, 10], a database containing all features would be huge and redundant. Therefore, the use of these methods is not a promising approach to reducing the computational load. The quickness and reliability of watermark detection are in a trade-off relationship. This trade-off should be balanced by clarifying how the analysis method is to be used for calculating image complexity from the viewpoint of the target application.

The work reported here extends our previous work [8] that the use of the rough image analysis improves watermark-detection performance. The complexities of the regions in a camera image can be roughly analyzed by using a Gaussian filter. Because the full image is not used, a smart-device application using this method can be light weight. We have now improved its implementation and demonstrated that the method performs no worse than alternative basic methods.

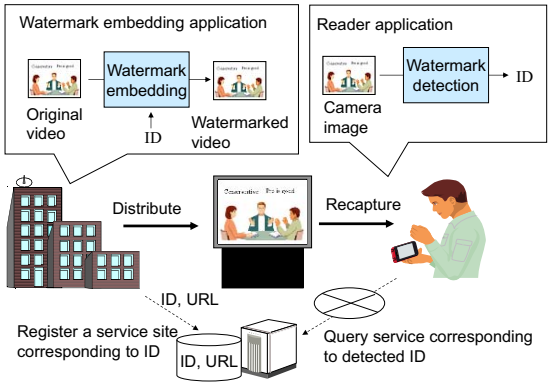

Fig. 1. Illustration of target application.

978-1-4799-9958-3/15 $31.00 © 2015 IEEE
DOI 10.1109/IIAI-AAI.2015.164

II. METHOD FOR VIDEO WATERMARK DETECTION

The watermark-detection process consists of a pre-process and a post-process, as shown in Fig. 2. The pre-process identifies the regions where watermarks are expectedly embedded by roughly analyzing the complexity of the input image. The post-process searches for watermarks only in those regions, not in the full image. Integers x, k, and q are input parameters determined beforehand.

<Pre-process>

Step 1 (Input): Receive frame image $\boldsymbol{Y}$.

Step 2 (Scaling): Scale down $\boldsymbol{Y}$ to scaled image $\boldsymbol{S}$ at a scaling ratio of $1/x$ so that computing load is reduced in rough image analysis.

Step 3 (Estimation): Divide $\boldsymbol{S}$ into k regions. For all integers i from 1 to k, iterate to create Gaussian image $\boldsymbol{G}_i$ from i-th region image $\boldsymbol{B}_i$ and calculate i-th metric value D_i as the sum of the absolute distance between each pixel value in $\boldsymbol{G}_i$ and the corresponding value in $\boldsymbol{B}_i$. Note that D_i roughly indicates the complexity of the corresponding region in $\boldsymbol{Y}$.

Step 4 (Listing): Take q estimation values from the k metric values sorted in descending order. List region numbers $\boldsymbol{R}$ ($r_1..r_q$) corresponding to the selected metric values. That is, q is the number of selected regions. The list order indicates the priority for detecting watermarks in the corresponding regions.

<Post-process>

Step 5 (Output): For each region number in $\boldsymbol{R}$, try repeatedly to detect watermarks in the corresponding region in the input image by using a correlation-based algorithm. If watermarks are detected, extract the embedded information and end the iteration. If all watermark detections for q regions fail, discard $\boldsymbol{Y}$ and return to Step 1 to receive the next input.

Note that the method for rough image analysis does not directly depend on the post-process (watermarking algorithm).

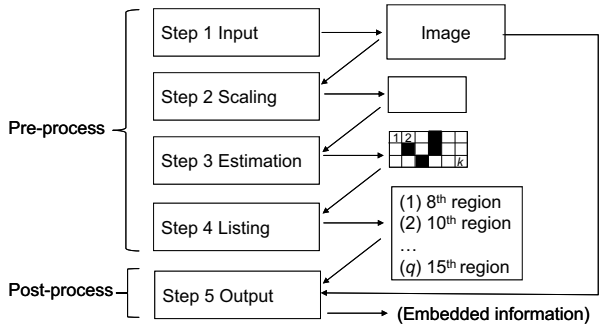

Fig. 2. Watermark detection using image analysis.

III. EVALUATION OF IMPROVED IMPLEMENTATION

A. Implementation description

The algorithm used for watermark detection was implemented as a reader application in a smart device. The information payload was 16 bits, which is practically required to identify content. The specifications of the smart device used are listed in Table 1. The specifications for the CPU clock frequency and memory size were the same as those for the previous implementation [8].

TABLE I. SMART DEVICE USED

Component	Specifications
Memory	RAM 2 GB
CPU	Quad-core, 1.5 GHz
Camera	5 M-pixels (HDTV)
Display	7 inch

B. Response time

A smart device with the improved implementation of our reader application and held in front of a displayed watermarked video could detect the watermarks in the recaptured video, as shown in Fig. 3. Five practical videos were used as samples. Watermark detection took about one second for an experienced user. It sometimes took several seconds for an inexperienced user despite the user having received instructions regarding the search range. These results demonstrate that the reader application is practical to some degree under the constraint that the camera be positioned within a certain search range with respect to the displayed video.

Fig. 3. Experiment scene with reader application.

C. Performance

The use of metric value D_i (obtained by roughly analyzing the complexity of the image) to determine the priority of regions in the input image results in the quick discard of ill-conditioned images and quicker processing of the next frame image.

The use of q to constrain the number of watermark-detection attempts means that watermark detection is done only for limited regions of the input image, which greatly reduces the computational load. This enables watermark detection to be performed in real time. Although we can set q smaller than in the previous implementation so as to improve frame-rate, doing so may degrade the reliability of watermark detection. The parameters were set such that the response times for the two implementations were subjectively almost the same.

The average total time for watermark detection for one frame with the improved implementation was 56 ms, as shown in Table 2. This means that the frame rate for watermark detection was 18 frames per second (fps) on average, compared

with 10 fps for the previous implementation [8]. This higher rate is closer to the 30 fps of full-frame real-time processing in a mobile camera. Note that the performance time includes the time for overhead processes such as displaying the input images in the pre-process.

TABLE II. AVERAGE TIME FOR WATERMARK DETECTION

Pre-process	Post-process	Total
14 ms	42 ms	56 ms

D. Comparison with other methods

We are unaware of any previously reported methods for calculating image features for real-time watermark detection that are adaptable to a smart device application. Since quickness and reliability of watermark detection are in a trade-off relationship, we compared the improved implementation (B) for analyzing image complexity with a simpler method (A) and a more complex method (C).

- *(A) Calculating gradient method:* The absolute horizontal gradient value is used as a metric for image complexity at the target pixel in a region of a downscaled image. It is quickly calculated by subtracting the pixel value of the target pixel from that of the horizontally next pixel and setting the sign of the reminder to positive.
- *(B) Improved implementation using Gaussian filter:* The summation of the absolute differences between a Gaussian image filtered using a 3×3 matrix and the original image is used as a metric, as described in Step 3 of the watermark-detection procedure.
- *(C) Calculating variance method:* The variance in the neighborhood is used as a metric. It is calculated by taking the differences between pixel values in the 5×5 neighborhood and the mean, squaring the differences, and dividing the sum of the squares by the number of pixels in the neighborhood.

<Reliability of watermark detection>

To evaluate the reliability of watermark detection with the three methods, we used the correlation value.

Consider a basic schema for watermark embedding. Original image $\boldsymbol{Y}$ is divided into regions that do not overlap. The i-th region of the watermarked image is obtained by adding a watermark pattern $\boldsymbol{P}$, which is an array of pseudo random numbers comprising $\{-1, +1\}^n$ multiplied by the positive watermark-strength, to the corresponding region of the original image:

$$\boldsymbol{Y}_i' = \boldsymbol{Y}_i + \boldsymbol{P} \tag{1}$$

In the corresponding schema for watermark detection, a correlation value is calculated on the basis of (2). The first term is near zero due to the randomness of P_k. The positive second term determines the correlation value independent of the image content. A watermark can be found by comparing the correlation value c_i with a positive threshold value.

$$c_i = \frac{1}{n}\sum_k P_k \cdot Y'_{i,k} = \frac{1}{n}\sum_k P_k (Y_{i,k} + p_k)$$
$$= \frac{1}{n}\sum_k P_k \cdot Y_{i,k} + \sum_k {p_k}^2 \tag{2}$$

In our target application, a watermarked image undergoes various image processings. Therefore, the correlation value is calculated using $f(\boldsymbol{Y}_i)$ rather than the $\boldsymbol{Y}_i$ in (2) when a function for image processing given to the watermarked image is defined. When image processing is given, watermarks embedded in a plane region are degraded worse than those in a complex one. Moreover, watermarks in such a region simply degrade image quality. To avoid such embedding, watermark strength is adjusted while applying the watermark pattern.

The three methods defined above were applied to the pre-process of the watermark detection procedure described in Section II. The test procedure for calculating the correlation values comprised five steps:

Step 1: Embed watermarks in original image.
Step 2: Scale down watermarked image.
Step 3: Divide scaled image into regions, and use method (A), (B), or (C) to calculate metric value for each region.
Step 4: Use metric values to identify regions likely containing watermarks.
Step 5: Calculate correlation value for each region in full watermarked image.

All parameters used in both the watermark-embedding and detection processes were set the same except for the methods used for rough image analysis. We used five 15-s HDTV video samples (2, 8, 20, 23, and 46) taken from the standard video set [11], as shown in Fig. 4.

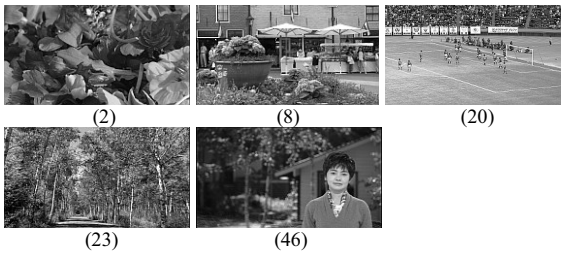

Fig. 4. Samples used.

The maximum correlation values for the identified regions are summarized in Table 3. The values for each sample were normalized between 0 and 1. A large correlation value obtained for a region indicates that watermarks have been strongly embedded in that region. Watermark detection in such a region is expectedly reliable under the assumption that strong watermarks can survive image processing. That is, a method with a correlation value higher than that of another method for an identified region would detect watermarks in that region more reliably. Note that method for calculating the correlation value for each region is independent of the method used for rough image analysis.

The improved implementation performed no worse than method (A). While the samples we used are adequate for embedding watermarks in many regions, practical content may not be so. However, since the value with method (A) was close to that of the improved implementation, if smart-device-hardware performance is a concern, method (A) may be a viable alternative to the proposed method, although method (A) is theoretically unreliable because the vertical aspect of an image is not analyzed.

Since the correlation values depend on the content, we cannot conclude that the improved implementation performed better than method (C) on the basis of our results. Nevertheless, the improved implementation should perform almost as well as method (C) in terms of watermark detection reliability.

TABLE III. MAXIMUM CORRELATION VALUES FOR IDENTIFIED REGIONS.

Sample	(A)	(B)[a]	(C)
2	0.8	0.9	0.6
8	0.7	0.7	0.7
20	0.4	0.4	0.8
23	1.0	1.0	0.9
46	0.7	0.9	0.9
(Average)	0.7	0.8	0.8

[a.] (B) is improved implementation

[b.]

<Quickness of watermark detection>

The average metric value calculated during rough image analysis was used. The number of surrounding pixels accessed for evaluating image complexity at a target pixel with methods (A), (B), and (C) was respectively 2, 9, and 25. The performance time for each method was measured using a PC (not a smart device) to enable the computational load to be compared. The execution time of a program running on a PC was almost proportional to that on a smartphone in our development environment. As shown in Table 4, method (A) was the fastest (5 ms), while method (C) was the slowest (94 ms).

The improved implementation (B) was slightly slower (16 ms) than (A) and much faster than (C) (94 ms). Therefore, in addition to its reliable watermark detection, the improved implementation is an adequate solution to the trade-off problem between quickness and reliability in this experiment.

TABLE IV. PERFORMANCE TIME

Sample	(A)	(B)[a]	(C)
Number of surrounding pixels	2	9	25
Performance time (ms)	5	16	94

[a] (B) is improved implementation

IV. SUMMARY

Our aim is a watermark application that will enable services and products to be promoted along with video content distribution. We have improved the implementation of a previously developed method for detecting watermarks using a smart device so that the computational load is lower. The frame rate with this implementation is 80% higher than the previous one (18 vs. 10 fps). We have demonstrated that improved implementation performs no worse than alternative basic methods.

This improved implementation roughly analyzes the regions in an input image and then searches for watermarks only in the complex regions, where watermarks are more likely to be embedded. As a result, less time is taken for watermark detection. A balance between quickness and reliability is achieved by adjusting the number of regions searched. Testings demonstrated that a smart device with a reader application using this implementation and held over a watermarked video display can detect watermarks in real time. Our reader application is thus practical to some degree.

Future work includes improving the usability of our application, such as by taking into account projection transformation and skewing.

Parts of the improved implementation will be used in a commercial product and service.

REFERENCES

[1] D. Roggen, G. Troster, P. Lukowicz, A. Ferscha, J. del R. Millan, and R. Chavarriaga: “Opportunistic human activity and context recognition,” IEEE Computer, Vol. 46, No. 2, 2013, pp. 36–45.

[2] K. Kamijo, N. Kamijo, and Z. Gang,: “Invisible barcode with optimized error correction,” in Proc. of IEEE Int'l Conf. on Image Processing (ICIP), 2008, pp. 2036–2039.

[3] M. Wuand and B. Liu: Multimedia data hiding, Springer-Verlag, New York, 2003.

[4] V. M. Potdar, S. Han, and E. Chang: “A survey of digital image watermarking techniques,” in Proc. of IEEE Int'l Conf. on Industrial Informatics (INDIN05), 2005, pp. 709–716.

[5] Y. Goto and O. Uchida: “Digital watermarking method for printed materials embedding in hue component,” in Proc. of Int'l Conf. on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP09), 2009, pp. 148–152.

[6] S. Gohshi, H. Nakamura, H. Ito, R. Fujii, M. Suzuki, S. Takai, and Y. Tani: “A new watermark surviving after re-shooting the images displayed on a screen,” Knowledge-Based Intelligent Information and Engineering Systems, Springer LNCS 3682, 2005, pp. 1099–1107.

[7] T. Nakamura, A. Katayama, R. Kitahara, and K. Nakazawa: “A fast and robust digital watermark detection scheme for cellular phones,” NTT Technical Review, 4, 3, 2006, pp. 57–63.

[8] T. Yamada, M. Kamitani, H. Nakamura, and T. Ohta: “Use of Rough Image Analysis in Smartphone Application for Video Watermark Detection”, in Proc. of 2014 Int’l Conf. on Infor-mation Science, Electronics and Electrical Engineering (ISEEE 2014), pp. 474–478 (2014).

[9] D.-G. Lowe: “Distinctive image features from scale-invariant keypoints,” Int’l Journal of Computer Vision, 60, 2, 2004, pp. 91–110.

[10] W. Burger and M.-J. Burge: SIFT, Principles of Digital Image Processing, Springer, 2013, pp. 229–296.

[11] ITE/ARIB HDTV Test Materials, Vol. 1, http://www.ite.or.jp/en/p_t/test_chart/.