A Digital Image Watermarking Scheme using *n*-level 2-dimensional DWT

Yong-Seok Lee
Dept. Electronic Materials Eng.
Kwangwoon University
Seoul, Republic of Korea
stucom12@kw.ac.kr

Young-Ho Seo
Sch. Liberal Arts
Kwangwoon University
Seoul, Republic of Korea
yhseo@kw.ac.kr

Dong-Wook Kim
Dept. Electronic Materials Eng.
Kwangwoon University
Seoul, Republic of Korea
dwkim@kw.ac.kr

ABSTRACT

This paper proposed embedding watermark data into all highest subband of n-level 2DDWT. Data embedding scheme is based on quantization index modulation of 2DDWT coefficient. The result of the image embedded watermark shows good visibility in near 40[dB] PSNR and robustness. This scheme is experimented in various resolution of image and attacked in various types filtering. The result shows developed performance in overall than the previous works, except certain types of attacks.

Ⅰ. Introduction

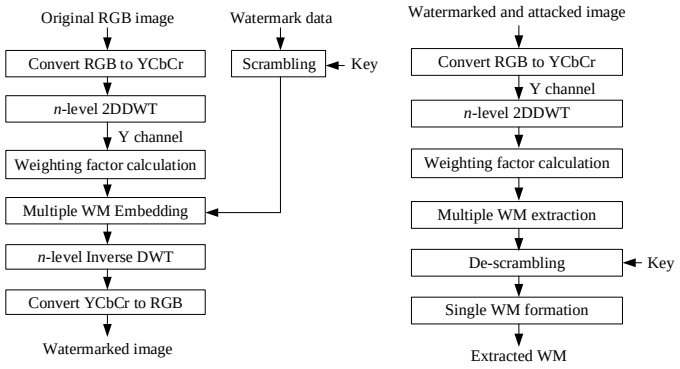
Recently, we live in various multimedia contents era. That contents spread out easily to communicate each other. So the ownership of the multimedia image is need to protected by watermark scheme. Watermark scheme is continuously developing for needs until now. Watermark scheme is commonly classified in two types; blind and non-blind type. We focused to research blind scheme. So using attacked image, we abstract watermark data, and propose watermark which is robust to several attacks and also visibility.

In this paper, we propose the spreading out multiple watermark data. Each subbands has different frequency, and each frequency has different robustness for attacks. Therefore, we attempt to embed watermark data in various frequency bands and extract the data by collecting and selecting.

Ⅱ. Watermark Embedding & Extracting

This paper proposes a digital watermarking scheme to protect the ownership of a digital image, which spreads out multiple watermark data into the subbands resulting from *n*-level 2-dimensional DWT of the host image.

This scheme is shown in Figure 1, which consists of watermark embedding scheme (a) and extraction scheme (b).



(a) (b)

Figure 1. The proposed watermarking scheme: (a) watermark embedding scheme, (b) watermark extraction scheme.

It uses only Y channel in YCbCr color format. So, RGB image should be converted into YCbCr image. Among them, Y channel is transformed to the frequency domain by *n*-level 2-dimensional (2DDWT), where *n* is determined by Eq. (1).

$$n = \min_{n}\left\{ pq < \frac{1}{4}\frac{PQ}{2^{2n}} \right\} \tag{1}$$

Where, $p \times q$ and $P \times Q$ are respectively the resolutions of watermark data and the host image. Then the weighting factors α_S for the n^{th} level subbands are calculated according to the kind of subband and the energy of the subband as Eq. (2), (3), and (4).

$$\alpha_{LL} = \begin{cases} T_{LL,l} & if \quad E_{LL} < T_{LL,l} \\ E_{LL} & if \quad T_{LL,l} \le E_{LL} \le T_{LL,h} \\ T_{LL,h} & if \quad T_{LL,h} < E_{LL} \end{cases} \tag{2}$$

$$\alpha_{HL} = \alpha_{LH} = \begin{cases} T_{M,l} & if \quad E_{M} < T_{M,l} \\ E_{M} & if \quad T_{M,l} \le E_{M} \le T_{M,h} \\ T_{M,h} & if \quad T_{M,h} < E_{M} \end{cases} \tag{3}$$

$$\alpha_{HH} = \begin{cases} T_{HH,l} & if \quad E_{HH} < T_{HH,l} \\ E_{HH} & if \quad T_{HH,l} \le E_{HH} \end{cases} \tag{4}$$

Where, $S \in \{LL, HL, LH, HH\}$ are the n^{th} level subbands. E_{LL}, E_M ($M = LH$ or HL), and E_{HH} are calculated by Eq, (5).

$$E_S = \frac{\beta_S}{MN} \sum_{i=1}^{MN} \left| c_i - \frac{1}{MN} \sum_{j=1}^{MN} |c_j| \right| \tag{5}$$

Where, $M \times N$ is the resolution of n^{th} level subband, c_i is the i^{th} coefficient of corresponding subband. β_S ($S \in \{LL, HL, LH, HH\}$) is the experimentally determined parameter. We have used $\beta_{LL} = 0.2$, $\beta_{LH} = \beta_{HL} = \beta_{HH} = 0.4$. Also $T_{LL,l}, T_{LL,h}, T_{M,l}, T_{M,l}$, and $T_{HH,l}$ are also experimental parameters. Here $T_{LL,l} = 1.5$, $T_{LLh} = T_{M,h} = T_{M,l} = T_{HH,l} = 2$ we used.

Meanwhile, the watermark data is scrambled using a *k*-stage linear feedback shift register (LFSR) in the manner that each coefficient is Exclusive-ORed to the serial output of the LFSR as Eq. (6). Here, the feedback characteristics and the initial values of the register are the security key.

$$w_i' = w_i \oplus f_{k,i} \tag{6}$$

Where, w_i and w_i' are respectively watermark data bit values before and after scrambling, and $f_{k,i}$ is the i^{th} serial output from k^{th} LFSR stage.

Then the scrambled watermark data is repeatedly inserted into each of the four n^{th} level subband as follows, where c_i' is the watermarked coefficient.

If $|c_i| < \alpha_S$, then

$$c_i' = \begin{cases} +0.5\alpha_S & if \quad w_i' = 0 \\ -0.5\alpha_S & if \quad w_i' = 1 \end{cases} \tag{7}$$

Else

$$c_i' = \alpha_S \left(\left\lfloor \frac{c_i}{\alpha_S} \right\rfloor + \gamma \right) \tag{8}$$

Where,

$$\gamma = \begin{cases} 0.5\dfrac{c_i}{|c_i|} & if \quad \left\lfloor \dfrac{c_i}{\alpha_S} \right\rfloor \bmod_2 = w_i' \\ 1.5\dfrac{c_i}{|c_i|} & if \quad \left\lfloor \dfrac{c_i}{\alpha_S} \right\rfloor \bmod_2 \ne w_i' \end{cases} \tag{9}$$

Then, the watermarked data is *n*-level inverse 2DDWTed and converted into RGB image to get the watermarked host image.

To extract the embedded watermark data, same procedure as embedding is performed by watermarked and attacked image with the key. The extracted data is de-scrambled with the same

Table 1. Experimental results for various images and various attacks

Attacks		512×512		1,024×1,024		640×480		1,280×960		1,920×1,080		3,840×2,160	
		PSNR [dB]	NCC	PSNR [dB]	NCC	PSNR [dB]	NCC	PSNR [dB]	NCC	PSNR [dB]	NCC	PSNR [dB]	NCC
Watermarked but no attack		40.21	-	40.14	-	39.33	-	39.37	-	39.48	-	39.43	-
Pixel value change attacks	JPEG quality 80/100	35.06	0.9991	36.37	1.0000	34.75	1.0000	35.73	1.0000	37.67	1.0000	38.16	0.9997
	JPEG quality 60/100	33.00	0.9328	33.99	1.0000	32.33	0.9869	33.43	1.0000	34.89	1.0000	35.82	0.9997
	JPEG quality 40/100	31.65	0.7157	32.46	0.9930	30.38	0.8165	31.93	0.9993	33.12	1.0000	34.27	0.9997
	JPEG quality 20/100	29.45	0.1988	29.93	0.7273	28.16	0.2800	29.45	0.8050	30.45	0.8851	31.58	0.9983
	sharpening	32.90	0.9808	34.11	0.9987	31.99	0.9598	33.20	0.9985	35.87	0.9995	37.58	1.0000
	Gaussian blurring (3×3)	33.47	0.9591	34.53	0.9927	32.70	0.9220	33.64	0.9939	36.25	0.9993	37.74	0.9997
	Gaussian blurring (5×5)	30.27	0.8445	30.66	0.8925	29.22	0.7358	29.86	0.9126	31.90	0.9758	33.58	0.9947
	Average blurring (3×3)	31.69	0.9358	32.74	0.9833	30.97	0.8792	31.85	0.9860	34.48	0.9993	35.98	0.9995
	Median filtering (3×3)	32.81	0.9623	34.08	0.9943	31.97	0.9443	32.85	0.9871	35.79	0.9995	37.44	0.9985
	Histogram equalization	18.73	0.5923	17.59	0.5816	19.08	0.6741	19.78	0.6199	18.73	0.6752	18.85	0.5500
	Gaussian noise 3%	30.37	0.5850	30.54	0.9774	30.53	0.7298	30.04	0.9956	30.44	0.9997	30.50	0.9997
	Salt&pepper noise 3%	34.28	0.9452	34.35	0.9997	34.38	0.9918	34.17	1.0000	34.33	1.0000	34.30	1.0000
	contrast (-20)	36.28	0.9972	26.66	0.8380	26.85	0.8847	27.59	0.8333	27.33	0.9396	26.17	0.8663
	Average	**31.54**	**0.8191**	**31.39**	**0.9214**	**30.25**	**0.8311**	**31.04**	**0.9332**	**32.40**	**0.9595**	**33.23**	**0.9543**
Geometric attacks	Shrink 0.8 fold	37.28	1.0000	39.81	1.0000	37.18	0.9997	38.91	1.0000	42.76	1.0000	43.55	0.9997
	Shrink 0.5 fold	32.52	0.9979	33.82	0.9997	32.04	0.9887	32.88	0.9997	35.82	1.0000	37.26	0.9997
	Shrink 0.25 fold	28.52	0.6748	28.91	0.9290	27.59	0.5528	28.21	0.9160	30.01	0.9929	31.78	0.9960
	Magnify 2 folds	44.65	1.0000	47.12	1.0000	44.46	1.0000	46.28	1.0000	49.95	1.0000	50.61	0.9997
	Rotation π/2	36.89	1.0000	37.69	1.0000	14.61	0.9979	13.21	0.9985	10.58	0.9977	10.91	0.9954
	Rotation π/3	13.19	0.9958	16.08	0.9962	15.09	0.9968	14.06	0.9985	11.39	0.9296	11.68	0.9230
	Rotation π/4	12.73	0.9950	15.75	0.9927	15.17	0.9950	14.16	0.9956	12.48	0.9699	12.63	0.9754
	Rotation π/6	13.15	0.9929	16.38	0.9933	15.68	0.9940	14.96	0.9948	13.64	1.0000	13.92	0.9995
	Cropping center 25%	11.31	0.9845	10.00	0.9750	9.64	0.9097	9.64	0.9030	10.63	0.9977	10.64	0.9910
	Average	**25.58**	**0.9601**	**27.28**	**0.9873**	**23.496**	**0.9372**	**23.59**	**0.9784**	**24.14**	**0.9875**	**24.78**	**0.9866**
Total average		**29.10**	**0.8767**	**29.70**	**0.9483**	**27.48**	**0.8745**	**27.99**	**0.9516**	**29.02**	**0.9709**	**29.77**	**0.9675**

Table 2. Comparison with existing methods

Attacks		Proposed				[1]	[2]	[3]	[4]	
		Visibility		WM		WM	WM	WM	Visibility	WM
		PSNR [dB]	SSIM	NCC	BER	NCC	NCC	BER	PSNR [dB]	BER
Pixel value change attacks	JPEG 80	36.29	0.9984	0.9998	0.00008	-	0.9559	0.0000	37.01	0.0087
	JPEG 60	33.91	0.9969	0.9865	0.0063	-	0.8040	0.0000	-	0.2187
	JPEG 40	32.30	0.9955	0.9207	0.0383	-	-	0.0020	-	-
	JPEG 20	29.84	0.9922	0.6491	0.1771	-	-	0.2058	-	-
	sharpening	34.27	0.9937	0.9895	0.0048	-	-	-	22.04	0
	Gaussian blurring (3×3)	34.81	0.9940	0.9870	0.0060	-	0.9912	0.0018	39.29	0.0530
	Average blurring (3×3)	32.95	0.9906	0.9638	0.0169	-	0.9618	-	28.26	0.2568
	Median filtering (3×3)	34.16	0.9925	0.9810	0.0089	-	0.8743	0.0984	32.03	0.2838
	Gaussian noise 0.5%	50.28	0.9999	0.9999	0.00001	-	0.846	-	-	-
	Gaussian noise 1%	39.06	0.9992	0.9999	0.00003	-	-	0.0585	-	-
	Gaussian noise 3%	30.46	0.9946	0.8812	0.0598	-	-	-	-	-
	Salt&pepper noise 1%	41.55	0.9995	0.9999	0.00001	-	0.9388	0.2604	-	-
	Salt&pepper noise 3%	34.30	0.9977	0.9894	0.0051	-	-	-	-	-
	contrast (-20)	28.48	0.9865	0.8932	0.0500	-	-	0.0426	-	-
	Average	**35.19**	**0.9950**	**0.9457**	**0.0026**	**-**	**0.9102**	**0.0743**	**31.72**	**0.1368**
Geometric attacks	Shrink 0.8 fold	39.92	0.9978	0.9999	0.00003	1.0000	-	-	-	-
	Shrink 0.5 fold	34.06	0.9925	0.9976	0.0010	0.9911	-	0.0240	-	-
	Shrink 0.25 fold	29.17	0.9789	0.8436	0.0774	0.6996	-	-	-	-
	Magnify 2 folds	47.18	0.9996	0.9999	0.00001	1.0000	-	-	-	-
	Rotation π/2	20.65	0.6829	0.9982	0.0008	1.0000	-	-	-	-
	Rotation π/3	13.58	0.6017	0.9733	0.0131	1.0000	-	-	-	-
	Rotation π/4	13.82	0.6201	0.9873	0.0060	0.9978	-	-	-	-
	Rotation π/6	14.62	0.6592	0.9957	0.0019	1.0000	-	-	-	-
	Cropping center 25%	10.31	0.5131	0.9601	0.0198	0.8965	-	0.1299	17.46	0.0390
	Average	**24.81**	**0.7828**	**0.9728**	**0.0133**	**0.9538**	**-**	**0.0769**	**17.46**	**0.0390**
Total average		**31.12**	**0.9120**	**0.9563**	**0.0214**	**0.9538**	**0.9102**	**0.0748**	**29.34**	**0.1228**

key as the embedding process. The result is more than 16 sets (more than 4 sets of each subband) of watermark data. The value of each location of watermark data is the one appearing most frequently.

Ⅲ. Experiments and Results

We used the 1024bit binary logo image for watermark embedded to host image, as shown in Figure 2.

Figure 2. Watermark data used(32x32)

This scheme is experimented for various kinds of images for various kinds of attacks, and the results of which are shown in Table 1. PSNR after embedding watermark shows near 40[dB] at all image of resolution and noticeable visibility. Robustness for the attacks is shown as NCC value, and shows high robustness for overall resolution. But, the image which watermark is embedded in high resolution, high level in 2DDWT shows better result. Watermark size is fixed as 32x32 in this experiment, so at 512x512 shows 3 level 2DDWT, 3840x2160 shows 5 level 2DDWT according to the image size.

Also we have compared our scheme to four previous works, which are shown in Table 2. From Table 1 and 2, our scheme can be said to have both enough invisibility of the watermark data and enough robustness for more general kind of attacks.

The compared papers below is selected mainly for watermark algorithm which is proposed recently. Watermark algorithm is continuously developing, and considered the recent papers as the most developed method.

Ⅳ. Conclusion

In conclusion, watermark embedding scheme proposed in this paper is color digital image watermark using all subband of 2DDWT. This paper has experimented visibility and robustness of watermark in various resolution, and shows necessity of performing various attacks such as global filtering, compression, and diverse geometric attacks to measure general watermark robustness. By the result of this paper, it suggests the direction of future research of watermark scheme.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (NRF-2013R1A1A2057798)

REFERENCES

[1] Y. Xueyi, et al. "A SIFT-based DWT-SVD blind watermark method against geometrical attacks," IEEE, Intl. Cong. CISP, pp. 323-329, Oct. 2014.

[2] J. Ouyang, G. Coatrieux, B. Chen and H. Shu, “Color image watermarking based on quaternion Fourier transform and improved uniform log-polar mapping,” Computers & Electrical Engineering, http://dx.doi.org/10.1016/ j.compeleceng.2015.03.004.

[3] H-T. Hu, Y-J. Chang and S-H. Chen, “A progressive QIM to cope with SVD-based blind image watermarking in DWT domain,” IEEE China SIP, pp. 421-425, Jul. 2014.

[4] R. Mehta, V. P. Vishwakarma and N. Rajpal, “Lagrangian support vector regression based image watermarking in wavelet domain,” Signal Processing and Integrated Networks, pp.854-859, Feb. 2015.

A Highly-Secured Chaos-Based Cryptographic Algorithm on Android for Private Image Storage and Transmissions

Sivapong Nilwong and Wimol San-Um
Intelligent Electronic Systems (IES) Research Laboratory
Faculty of Engineering, Thai-Nichi Institute of Technology (TNI)
1771/1 Pattanakarn 37 Rd., Suanluang, Bangkok, Thailand, 10250. E-mail: sivapong@tni.ac.th

***Abstract*—This paper presents a chaos-based cryptographic algorithm on Android for private image storage and transmissions. The robust chaotic map based on absolute value nonlinearity is presented and analyzed before employing in the encryption process. Dynamic behaviors of the chaotic map are described in terms of Equilibria, Jacobian matrix, bifurcation diagram, and Lyapunov Exponent spectrum. The designed encryption process and decryption process were based on XOR operations of pixel values with the random bit signals generated from the chaotic map. The keys were designed based on a 16 alphanumeric characters password for the initial condition and control parameter. The implemented Android application was developed using Java programming language on the Android studio. The generally used LENA image with the size of 256×256 and 512×512 were employed for demonstrations of encryption and decryption procedures. Encryption qualitative performance were evaluated through pixel density histograms and correlation plots. Encryption quantitative performances were evaluated through encryption speed, correlation coefficients, Net Pixel Change Rate (NPCR), and Unified Average Changing Intensity (UACI). Results from actual Android application on a smart phone with both correct and wrong keys are also included.**

Keywords—Chaos, Cryptographic Algorithm, Android, Image Storage, Image Transmission.

I. INTRODUCTION

Recent advances in communications, especially the internet, have led to great demands of secured data storage and transmissions. Security and reliability of data are needed in various applications such as medical, industrial, military, and internet of things [1]. From recent research, image data is one of the most used data types in data cryptography [2-5]. Various cryptographic approaches were suggested for both software and hardware, involving chaos-based approaches. Chaos-based cryptography exploits the properties of chaotic systems, including sensitivity to initial conditions and parameters of the system. Chaotic maps which are iterative functions in discrete-time domain that can exhibit behaviors, were frequently implemented in chaos-based cryptography [3-4]. For example, the image encryption and decryption algorithm using the logistic map and two-dimensional cat map [3], and the colored image encryption algorithm using the logistic map and characteristics of plain image [4]. However, chaotic maps implemented recently were not robust and led to complex design of the cryptographic system. Despite the fact that, an implementation of cryptographic systems on mobile platforms provide convenience in image encryption and decryption, only few implementations of chaos-based cryptography using chaotic maps on mobile device were reported, such as the chaos-based secret image transmission and reception on Android [5].

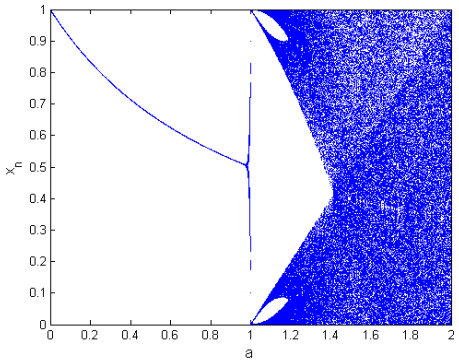


Fig. 1. Bifurcation diagram of (1) where the value of control parameter *a* is in the range of [0, 2].

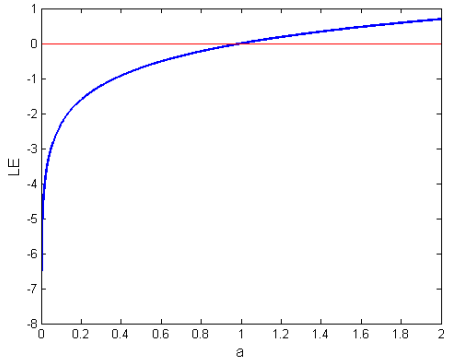


Fig. 2. LE spectrum of (1) where the value of control parameter *a* is in the range of [0, 2].

According to the report in 2015 by the Internet Society, Android has a significant portion of mobile devices share [6]. Therefore, this paper proposed the cryptographic algorithm on Android mobile platform. The proposed cryptographic algorithm is based on the second case of chaotic maps with absolute value nonlinearity and the XOR operation. The implementation of chaotic maps with absolute value nonlinearity on Android was experimented in previous research by generalizing the first case of chaotic maps with

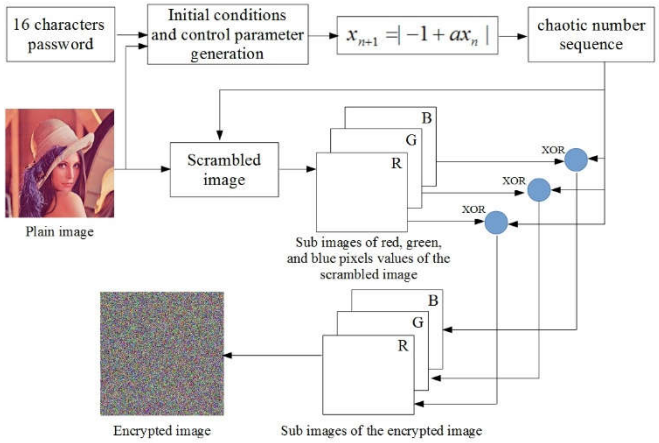


Fig. 3. Block diagram of encryption procedure.

absolute value nonlinearity on Android [7]. Results from the experiment on Android stated that chaotic maps with absolute value nonlinearity can be implemented on Android.

II. THE SECOND CASE OF CHAOTIC MAPS WITH ABSOLUTE VALUE NONLINEARITY

The second case of chaotic maps with absolute value nonlinearity which implemented in this paper is one of four robust chaotic maps proposed in the recent research [8]. All cases of chaotic maps with absolute value nonlinearity were tested for their robustness by testing random bit sequences which consist of 1,000,000 bits generated from the chaotic maps. Random bit sequences generated from chaotic maps with absolute value nonlinearity passed all 15 standard tests of the National Institute of Standards and Technology (NIST) statistical test suite from 800-22rev1a special publication. The second case of chaotic maps with absolute value nonlinearity can be mathematically described as

$$x_{n+1} = |-1 + ax_n| \tag{1}$$

where a is the control parameter of the chaotic map which has the value in the range of [0, 2] and x_n is the state variable of the chaotic map which has the value in the range of [0, 1]. Additionally, the Jacobian matrix of (1) is calculated as

$$J = [a(sign(-1 + ax))] \tag{2}$$

, and the fixed points of (1) are indicated as

$$x = \frac{1}{a \pm 1} \tag{3}.$$

Chaotic behaviors of (1) are also analyzed in this paper, qualitatively through the bifurcation diagram, and quantitatively through the Lyapunov Exponent (LE) spectrum.

Fig. 4. a. The plain image, b. The encrypted image, c. The decrypted image using the correct key, and d. The decrypted image using the wrong key

The bifurcation diagram indicates possible long-term values as a function of a bifurcation parameter. Fig. 1 shows the bifurcation diagram of (1) from MATLAB. From the bifurcation diagram, chaotic behaviors of (1) occurred when the control parameter a is in the range of [1, 2] in which the number of blue dots that refer to possible values in time domain of (1) is enormous in this region. The LE quantifies the mean rate of divergence of trajectories which start infinitesimally close to the reference and also provides a measurement of the instability of the system. The LE can be expressed mathematically as [9]

$$\lambda = \lim_{n\to\infty}(\frac{1}{N})\sum_{n=1}^{N}\ln f'(x_n) \tag{4}$$

where N is the number of iterations, λ is the LE value, and $f'(x_n)$ is the first derivative of the chaotic map, Fig. 2 shows the LE spectrum of (1) from MATLAB. As seen from the LE spectrum, the LE value is greater than 0 when the value of control parameter a is in the range of [1, 2], that is, equation (1) exhibits chaotic behaviors in this region, the same region as the bifurcation diagram of (1).

III. PROPOSED SECURED CHAOS-BASED CRYPTOGRAPHIC ALGORITHM

The proposed chaotic-based cryptographic algorithm for Android in this paper is based on XOR operation of the pixel values in each separated color planes of the plain image with numbers in a chaotic sequence generated from (1).

Fig. 3 shows the block diagram of the encryption process in the proposed chaos-based cryptographic algorithm. The encryption process starts by the acquisition of the plain image and the password which is a set of 16 alphanumeric characters. The acquired password is transformed into a set of ASCII number which each alphanumeric character is transformed into 8-bit ASCII number, resulted in a 128-bit numeric password. The transformed password and average pixel values of each color plane of the plain image are used to calculate the initial condition and the control parameter for the chaotic map in (1). The generated initial condition and control parameter are then employed in (1) to generate a chaotic sequence of numbers

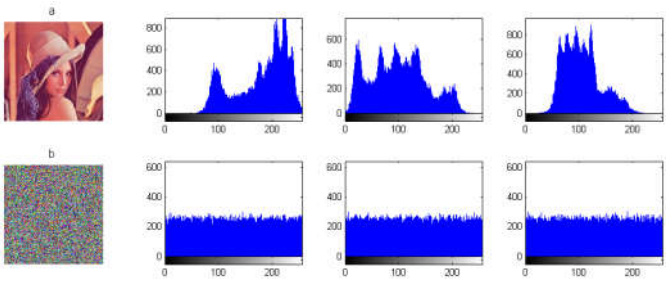


Fig. 5. a. Histogram of the plain image, b. Histogram of the encrypted image in red, green, and blue color planes, respectively

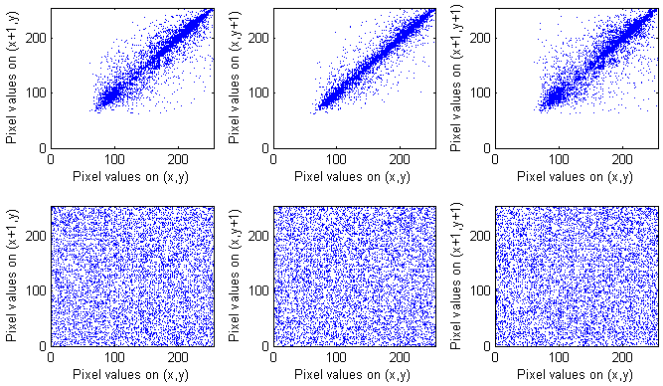


Fig. 6. Upper, pixel correlation diagram of the plain image, and lower, pixel correlation of the encrypted image in red color plane to adjacent pixels in horizontal, vertical, and diagonal directions, respectively

with the number of iterations used equals to number of pixels+2. After the chaotic sequence is generated, the acquired plain image is then scrambled using the generated chaotic sequence to swap pixel values of each pixel in all color planes with the pixel indicated by the number of each element of the chaotic sequence multiplied by the number of pixels-1, starts from the top-leftmost pixel of the plain image and the first element of the chaotic sequence. The scrambled image is then processed through the XOR operation by XOR pixel values in each color plane with numbers in the generated chaotic sequence from (1) multiplied by 255, starts from the top-leftmost pixel of the scrambled image. Additionally, XOR operation of pixel values in red color plane starts by XOR the multiplied number of the first element of the chaotic sequence with the first pixel value in red color plane, while the XOR operation of green and blue color planes starts by XOR multiplied numbers of the second and the third elements of the chaotic sequence with the first pixel values in green and blue color plane, respectively. After the XOR process was completed, the processed or encrypted image is displayed and prompt users to export the encrypted image. The encrypted image can be exported as .PNG image file along with average pixel values in each color plane as .txt text file. The exported .txt text file which contains average pixel values is required in the decryption process.

The decryption process is similar to the encryption process. However, there are some differences which are, first, the decryption process requires average pixel values in each color plane of the original plain image which contained in the text

TABLE I. CORRELATION COEFFICIENTS BETWEEN THE PLAIN IMAGE AND THE ENCRYPTED IMAGE AT THE SIZE OF 256×256 AND 512×512

Correlation coefficients (*C*)	**256×256**	**512×512**
C_{RR} (Red-Red)	0.0036	0.0008
C_{RG} (Red-Green)	-0.0027	-0.0016
C_{RB} (Red-Blue)	-0.0034	-0.0008
C_{GR} (Green-Red)	0.0026	0.0010
C_{GG} (Green-Green)	-0.0002	0.0003
C_{GB} (Green-Blue)	-0.0022	-0.0009
C_{BR} (Blue-Red)	0.0046	0.0017
C_{BG} (Blue-Green)	-0.0008	0.0018
C_{BB} (Blue-Blue)	0.0001	-0.0007

TABLE II. NPCR AND UACI OF THE ENCRYPTED LENA IMAGE AT THE SIZE OF 256×256 AND 512×512

	256×256	**512×512**
NPCR (Red)	99.5697	99.5975
NPCR (Green)	99.6414	99.6250
NPCR (Blue)	99.6124	99.6143
UACI (Red)	33.5511	33.4609
UACI (Green)	33.4863	33.5438
UACI (Blue)	33.5433	33.5533

file that exported from the encryption process to generate initial conditions and control parameters for (1). Second, the XOR operation is operated before the image unscrambling. The last difference is that, the image unscrambling process which has the same mechanism as the scrambling process, starts from the pixel on the bottom-rightmost of the image instead of the top-leftmost pixel.

IV. EVALUATION OF THE PROPOSED CHAOS-BASED CRYPTOGRAPHIC ALGORITHM

The implemented Android application was developed on the Android Studio Integrated Development Environment (IDE) which is an official IDE for Android application development. Despite the fact that the Android Studio has an emulator which can be used to test the developed application on virtual device, the implemented application in this paper was tested on an actual Android device, the SONY Xperia M2 smartphone. Fig. 4 displays the results on the smartphone which operating the implemented application, where (a) shows the plain image before the encryption, (b) shows the encrypted image, shows the decrypted image which used the same key as the encrypted key, and (d) shows the decrypted image using the wrong key. The average computation time used in the encryption process when encrypt the LENA image in sizes of 256×256 and 512×512 pixels, 5 times for each image, were measured. The experiment on the smart phone shows that

average computation time of the 256×256 pixels LENA image is 0.837 seconds, and average computation time of 512×512 pixels LENA image is 3.591 seconds.

The result from encryption process of the implemented Android application which used the 256×256 LENA image as the plain image, was analyzed using MATLAB 2013a quantitatively and qualitatively. Quantitative analysis was achieved through the correlation coefficients, the Number of Pixel Changing Rate (NPCR), and the Unified Averaged Changed Intensity (UACI). Besides, qualitative analysis was achieved through the pixel density histograms and the pixel correlation diagrams. Fig. 5 shows the pixel density histograms of the plain image and the encrypted image. The pixel density histograms show that the encrypted image has uniformly distributed histograms in all color planes, i.e. red, green, and blue color planes, while the pixel density histograms in all color planes of the plain image are not. Fig. 6 illustrates the pixel correlation diagrams of the plain image and the encrypted image in red color plane. Pixel correlation diagrams depict the pixel values of each pixel compared to adjacent pixels in horizontal, vertical, and diagonal directions. The correlation diagrams of the plain image reveal that pixel values of each pixel are similar to adjacent pixels. On the other hand, pixel values of each pixel in the encrypted image are significantly differ to their adjacent pixels, the correlation diagrams of the encrypted image are scattered in all directions.

Table 1 depicts the correlation coefficients between the plain image and the encrypted image in all color planes. The correlation coefficients were achieved through the correlation coefficient of pixels, comparing each color plane of the plain image to all color planes of the encrypted image. As described in table 1, all correlation coefficients between the plain image and the encrypted image were close to zero. Table 2 shows the NPCR and UACI value of the encrypted 256×256 pixels and 512×512 pixels LENA image. The NPCR is used to measure the changing rate of pixels in encrypted images, and the UACI is used to measure changes in pixel values of the encrypted images, where their corresponding original plain images have some slight changes, i.e. only one pixel changed [10]. The NPCR and UACI can be described as

$$NPCR = \frac{\sum_{i,j} D(i,j)}{T} \times 100 \qquad (6)$$

$$UCAI = \frac{\sum_{i,j} C_1(i,j) - C_2(i,j)}{T \times P} \times 100 \qquad (6)$$

where C_1 and C_2 denote the pixel in encrypted image 1 and 2 which has one pixel changed, D indicates if the pixel value is changed which is 0 if $C_1 = C_2$, and 1 if $C_1 \neq C_2$, i and j is the horizontal and vertical position of pixels in images, T is number of pixels, and P is maximum value of the image format, in this case is 255. The NPCR and UACI in table 2 is close to 99% and 33% for both 256×256 and 512×512 images, which were close to theoretically values.

CONCLUSIONS

This paper has proposed a chaos-based cryptographic algorithm based on the chaotic map with absolute value nonlinearity. The proposed cryptographic algorithm can be implemented on actual Android device, and was tested on a smart phone. The test results on smart phone reveal that the proposed cryptographic algorithm can encrypt the test image which has the size of 256×256 pixels in less than 1 second. The result from the encryption process of the proposed cryptographic algorithm were analyzed qualitatively through pixel density histograms and pixel correlation diagrams, and quantitatively through the correlation coefficients, NPCR, and UACI on MATLAB. Even though the satisfied results are acquired from the analysis, the proposed image cryptographic algorithm need improvements with more types of tests. However, the proposed cryptographic algorithm provides an alternative algorithm to be implemented on mobile platforms.

REFERENCES

[1] M. B. Barcena and C. Wueest, *Insecurity in Internet of Things*, version 1.0, March 2015.

[2] Ch.K. Volos, I.M. Kyprianidis, and I.N. Stouboulos, "Image encryption process based on chaotic synchronization phenomena," *Signal Processing*, vol. 93, pp. 1328–1340, 2013.

[3] M. Ahmad et al, "A New Algorithm of Encryption and Decryption of Images Using Chaotic Mapping," *International Journal on Computer Science and Engineering*, vol. 2, pp. 46–50, 2009.

[4] M. A. Murillo-Escobar et. al, "A RGB image encryption algorithm based on total plain image characteristics and chaos," *Signal Processing*, vol. 109, pp. 119–131, 2015.

[5] G. Savithri and K.L. Sudha, "Android Application for Secret Image Transmission and Reception Using Chaotic Steganography," *International Journal of Innovative Research in Computer and Communication Engineering.*, vol. 2, issue 7, pp. 5107-5113, 2014.

[6] Internet Society, *Internet Society Globas Internet Report 2015*, 2015.

[7] W. San-Um and S. Nilwong, "The Development of an Android Application for The Chaotic Map with Absolute Value Nonlinearity", *MITicon 2014,* pp. 112-115, 2014.

[8] W. San-Um and P. Ketthong, "The Generalization of Mathematically Simple and Robust Chaotic Maps with Absolute Value Nonlinearity", *TENCON 2014-2014 IEEE Region 10 Conference,* pp. 1–4, 2014.

[9] M. Cencini, F. Cecconi, and A. Vulpiani, "Characteristic Lyapunov exponents," *CHAOS From Simple Models to Complex Systems,* pp. 111-126, 2010.

[10] Yue Wu, "NPCR and UACI Randomness tests for Image Encryption", *Journal of Selected Areas in Telecommunication (JSAT),* April Edition, pp. 31-38, 2011

Implementation of Hologram Generation from Integral Photographic Image with Wide Viewing-Zone Angle

Tomohiro Suzuki[*,**], *Yasuyuki Ichihashi*[**], *Ryutaro Oi*[**], *Kenji Yamamoto*[**], *Takashi Kakue*[*], *Tomoyoshi Shimobaba*[*], *Tomoyoshi Ito*[*]

[*]Graduate School of Engineering
Chiba University
Chiba, Japan

Universal Communication Research Institute
NICT
Koganei, Japan

***Abstract*—We implemented a computational processing for enlarging a viewing zone of holographic images to a GPU with the aim of realizing real-time reconstruction of wide-viewing-zone holographic images. We achieved 50 times faster-calculation by a GPU than a CPU.**

***Keywords*—Holography; Integral Photography; Three Dimensional Imageng; GPU; CUDA**

I. INTRODUCTION

We have proposed and developed a real-time reconstruction system for 3D live scenes, which uses integral photography (IP) and electro-holography for recording and reconstructing 3D live scenes respectively [1]. However, small viewing-zone angle of reconstructed image is one of remaining problems of 3D displays based on electro-holography. Various methods have been proposed for enlarging the viewing-zone angle of reconstructed 3D images [2, 3]. We have focused on the wider viewing-zone-reconstruction system according to Senoh's method [3] which uses time-division multiplexing of three different horizontal viewing-zone-angle holograms.

In order to apply the method to our real-time reconstruction system, we devised a hologram-generation algorithm which generates three 8K hologram, which have different viewing-zone angle in horizontal direction respectively, from 4K integral photographic image (IP image) [4]. However, we have not yet realized real-time reconstruction of 3D scenes with the wider viewing-zone angle, because the calculation cost of the devised algorithm is too high to be performed in real time. In this paper, we show that the processing time for the calculation of hologram generation by the algorithm can be reduced by using a Graphic Processing Unit (GPU) with CUDA developed by NVIDIA Corporation.

II. ALGORITHM OF WIDE VIEWING-ZONE HOLOGRAM GENERATION

Integral photography is a technique capable of capturing ray information of objects as an IP image by using a microlens array as shown in Fig. 1 [5]. An IP image consists of a lot of micro images (elemental images) made by each microlens. Each elemental image contains ray information of objects from a different direction pixel by pixel by microlenses as shown in Fig. 2. Therefore, we can reconstruct 3D image from IP images by using microlens array which is used in recording processing. The viewing-zone angle of the 3D image reconstructed from the IP image and the resolution of the reconstructed 3D image depend on the specifications of microlenses (the size of elemental images, the number of elemental images, etc.) In order to reconstruct 3D images with high resolution and wide viewing-zone angle from IP image, we need to capture IP images with a higher-resolution camera as possible. Although we have an 8K reconstruction system [3], we have no camera which can capture 8K images. In this paper, we capture a 4K IP image and generate an 8K IP image from the captured IP image for hologram generation by using Bicubic interpolation method.

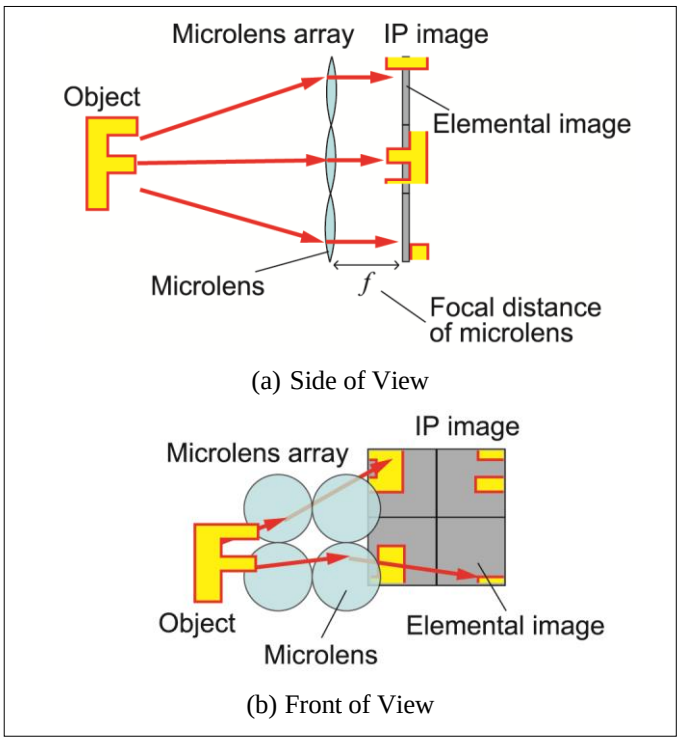


Fig. 1. Capture System of Integral Photography

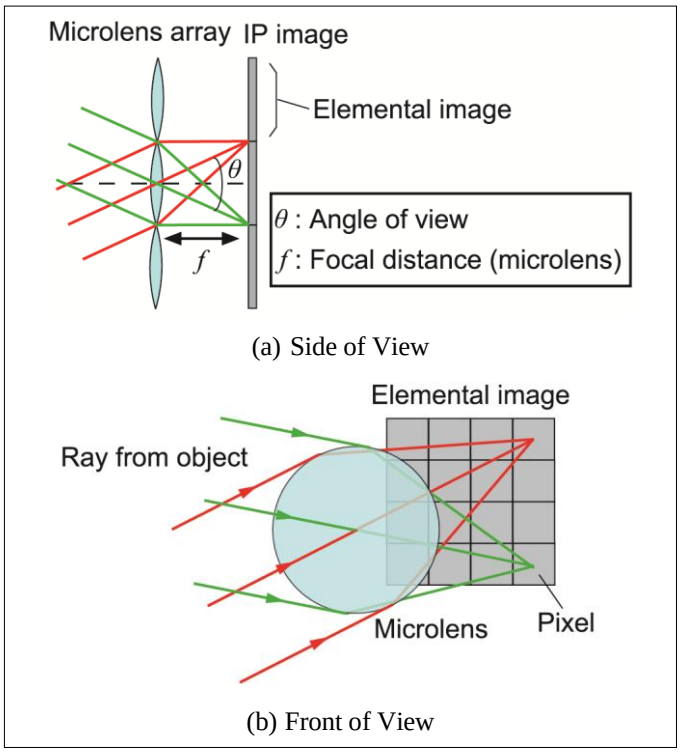


Fig. 2. Viewing-Zone Angle of IP

We devised a method of generating three holograms [4], in order to use time-division multiplexing system [3]. However, in order to reconstruct wider viewing-zone-angle 3D image by the time-division multiplexing system, we needs multiple holograms which contain different viewing zone in horizontal direction. Because we generate a single hologram from a single IP image, each IP image, which has ray information of each viewing zone, has to be prepared for the system. Based on the devised hologram-generation method, we generate three 8K IP images from a single 4K IP image as described below. After generating three IP images, we perform hologram generation with the three 8K IP image. Each pixel in an elemental image has different ray information from a different direction as shown in Fig. 2. We can generate new elemental images that have ray information of the desired viewing zone by selecting appropriate pixels from original elemental images as shown in Fig. 3, and interpolating the vacant pixels between the selected pixels. Then, we generate new three IP images for wide viewing-zone-angle hologram generation by executing the processes for all elemental images.

We execute hologram generation by simulation of light propagation as shown in Fig. 4. In the simulation, we record a reproduction light of each IP image in holograms. However, we need to add random phases to the reproduction light of each IP image in order to diffuse the reproduction light and prevent the concentration of the reproduction lights on the holograms [6]. In this paper, we used a method of approximating the simulation of the light propagation in Fig. 4 to only one FFT and reduce the calculation cost of the simulation by using a capture system under a specific constraint [7].

III. Implementation to a GPU

In this paper, we implemented the above image processes and calculations to a GPU with CUDA 6.0. Fig. 5 shows the flowchart of the processes.

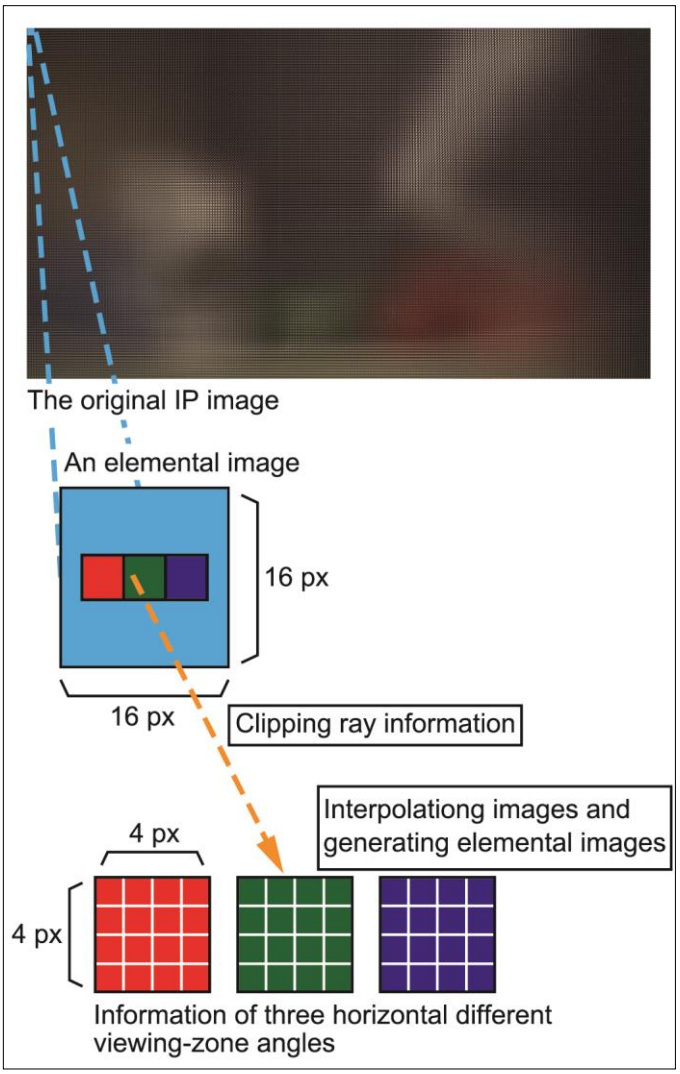


Fig. 3. Algorithm for Generating IP Images

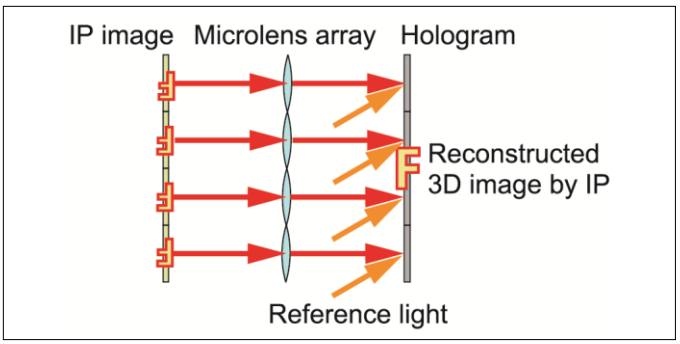


Fig. 4. Hologram Generation from an IP Image

First, a GPU selects pixels that have desired ray information from an original IP image captured with a 4K

camera as shown in Fig. 3 (Process 1 in Fig. 5) for hologram generation. The GPU generates three new 4K IP images that have ray information from different viewing zones by interpolating the selected pixels (Process 2 in Fig. 5). Then, the GPU enlarge the size of the three 4K IP images to 8K size, which corresponds to the resolution of display for reconstructing holograms (Process 3 in Fig. 5). In the process of enlarging, the number of elemental images is increased in order to improve the resolution of reconstructed 3D image. Finally, the GPU generates three holograms from each new IP image by the optical simulation as shown in Fig. 4 (Process 4 in Fig. 5).

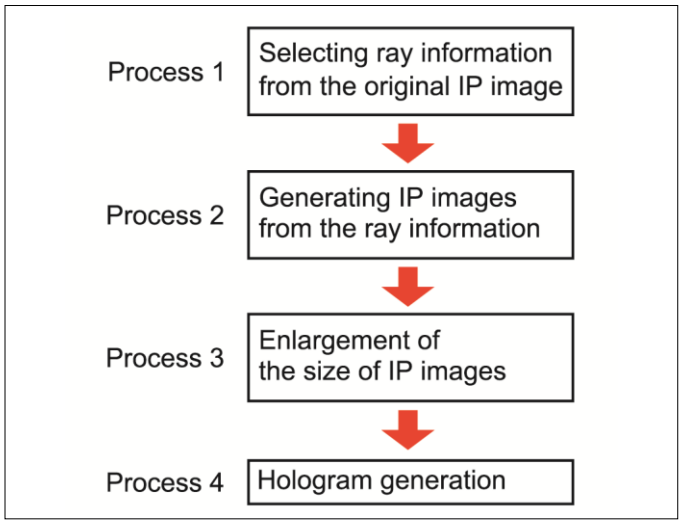


Fig. 5. Flowchart of Wide Viewing-Zone Hologram Generation

Many core computing is effective to accelerate the above processes because the processes by pixels are independent of each other. With CUDA, we can use the texture unit in a GPU and implement fast calculation of enlarging IP images (Process 3 in Fig. 5) easily. However, we need to take care of implementation to execute efficient memory access. A GPU computes micro and many array of pixels in the process. We can use cuFFT, which is the library for computing FFT with CUDA for accelerating hologram generation (Process 4 in Fig. 5).

Table I shows the processing times of the processes in Fig. 5 by using a CPU or a GPU. We use an Intel Core i7 2600K (using only single core) as the CPU, and a Tesla K20c as the GPU.

The processing time for Process 1 in Fig. 5 by the GPU is 25 times faster than that by the CPU. The processing time for Process 2 in Fig. 5 by the GPU is 238 times faster than that by the CPU. The processing time for Process 3 in Fig. 5 by the GPU is 12 times faster than that by the CPU, and the processing time for Process 4 in Fig. 5 by the GPU is 64 times faster than that by the CPU. For the entire processing, the calculation by the GPU is 50 times faster than that by the CPU. The accelerating rates of Process 1, Process 3, and Process 4 are lower than that of Process 2. In Process 1, the efficiency of memory access is low because a GPU have to access many micro dotted area of the memory when the GPU selects pixels that have ray information from the desired viewing zone. In Process 3, register spilling resulted in the large overhead. In Process 4, a GPU spends a long time to calculate random phases which is used when a GPU simulates light propagation of IP images. We need to resolve these problems to accelerate hologram generation more.

TABLE I. PROCESSING TIMES BY A CPU AND GPU

		CPU	***GPU*** (accelerating rate)
Processing Time [ms]	Process 1	10	0.4 (25 times)
	Process 2	476	2 (238 times)
	Process 3	443	36 (12 times)
	Process 4	26114	407 (64 times)
	Entire Processing	27061	546 (50 times)

IV. CONCLUSION

We successfully reduced the processing time required for wide viewing-zone-angle hologram generation with IP by 50 times when using a GPU rather than a CPU. In the future, we plan to accelerate more the processes by resolving a register spill in the processes, reducing a calculation cost about random phases for the light propagation, etc.

REFERENCES

[1] R. OI, *et al*., *IEICE Technical Report*. Image engineering **105** (IE2005 11-17), 31-35 (2005) (in Japanese)
[2] Y. Takaki and N. Okada, *Opt. Express* **18**, 11327-11334 (2010)
[3] T. Senoh, *et al., J. Display Technol.* **7**, 382-390 (2011)
[4] T. Hayashi, *et al., ITE Trans. MTA* **69**, 177-179 (2015) (in Japanese)
[5] J. Arai, *et al., Opt. Express* **21**, 3474-3485 (2013)
[6] A. W. Lohmann and D. P. Paris, *Applied Optics* **6**, 1739-1768 (1967)
[7] R. Oi, *et al., ITE Trans. MTA* **61**, 198-203 (2007) (In Japanese)

Adaptive Feature Tracking for 3D Ego-Motion Estimation

Yu-Hsiang Chang[1], Ting-Hsiang Huang[1],
Chia-Yen Chen[1], Chen-Chi Chunang[1], Yu-Mei Hong[1]
[1]Dept. Computer Science & Information Engineering
National University of Kaohsiung
Kaohsiung, Taiwan
ayen@nuk.edu.tw

Chia-Chen Kuo[2]
[2]National Center for High-performance Computing
National Applied Research Laboratories
Hsinchu, Taiwan
cckuo@narlabs.org.tw

***Abstract* — Tracking of features in a video sequence is a critical task in visual odometry and has significant influence on the accuracy of ego-motion estimation. The paper proposes a novel method to improve the accuracy of tracking by adaptively constraining features matches based two criteria. A scale constraint for the disparity of within the binocular images and an angular constraint associated with epipolar geometry. Experimental results performed using benchmark stereo image sequences demonstrated the effectiveness of the proposed method and its improvements over existing approaches.**

Keywords : egomotion estimation, binocular stereo, tracking, feature matching, epipolar geometry

I. INTRODUCTION

Research on visual odometry began in the 80's with a representative work on the vision-guided rover [1]. The work starts a series of research associated with the theories and practice toward solving ego-motion estimation problem in fields of visual odometry and simultaneously localisation and mapping (SLAM). In visual odometry, the environment can often be quite complex with many uncontrolled variables which makes accurate tracking a difficult task. Nevertheless, to obtain more correct egomotion estimation, tracking is essential and critical. To improve tracking, the number and distribution of tracked features are important factors.

In this work, we propose a generic mathematical framework for effective feature tracking for binocular stereo image sequences. We investigate the possibility to dynamically adjust constraining parameters associated with the disparity and the epipolar geometry to improve the accuracy of tracked features.

This paper is organised as follows. In Section 2 some related work are surveyed. The proposed method is described in Section 3. In Section 4, we perform experiments using the proposed method and compare its performance with a conventional implementation. Finally, Section 6 concludes this work and gives ideas for future work.

II. RELATED WORK

Works have been done previously to improve feature tracking [2]. One effective approach is presented in [3], which is able to dramatically improve the overall accuracy using outlier rejection. A multi-frame integration strategy predicts and corrects each tracked feature according to the previous observations in [4], which achieves precise ego-motion estimation without global optimization. There are also keyframe-based feature selection strategies, such as proposed in [5]. Some other approaches also include integrate sensory data fusion technique, such as in [6] and [7], which fuses the global positioning system data with visually recovered motion to decrease accumulated motion error. An overall review can be found in [8].

The general pipeline of binocular visual odometry is as shown in Figure 1. The input of pipeline is a pair of binocular images and the output contains the estimated 3D structure of the tracked scene points and the motion of the vision system relative to the location where the last input images were taken.

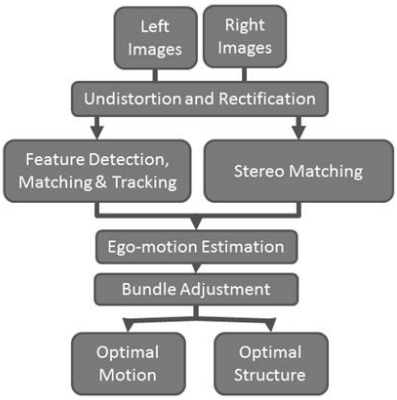


Figure 1. Pipeline of a binocular feature-based visual odometry framework.

The imaging geometry of a pinhole camera can be described by a nonlinear perspective projection model given in [9]. To model the nonlinear lens distortion geometry, we first project a point (x,y,z) in the 3-D space onto an ideal pixel ($\dot{u}$,$\dot{v}$) in normalised image plane, as

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (1)$$

where ~ denotes projective equivalence (i.e. equality up to a scale), the rotation matrix R∈SO(3) and the translation t define the extrinsic parameters transforming world coordinates to the

camera-centred frame. The nonlinear lens distortion is then approximated by the polynomial function

$$\begin{pmatrix}\breve{u}\\ \breve{v}\\ 1\end{pmatrix} \sim \begin{pmatrix}\dot{u} & 2\dot{u}\dot{v} & r^2+2\dot{u}^2 & 0\\ \dot{v} & r^2+2\dot{v}^2 & 2\dot{u}\dot{v} & 0\\ 0 & 0 & 0 & 1\end{pmatrix} D \qquad (2)$$

where the row-vector D models distortion factors. The normalised coordinates ($\breve{u}$,$\breve{v}$) are transformed onto the image plane to form a pixel (u,v) observed in a captured image by

$$\begin{pmatrix}u\\ v\\ 1\end{pmatrix} \sim \begin{pmatrix}f_u & 0 & u_c\\ 0 & f_v & v_c\\ 0 & 0 & 1\end{pmatrix} \begin{pmatrix}\breve{u}\\ \breve{v}\\ 1\end{pmatrix} \qquad (3)$$

Figure 2. An example of rectified stereo image pair.

After the above procedure, the input binocular images can be rectified, and Figure 2 shows an example of rectified image pair with pipolar lines marked in red.

III. Proposed Method

A good selection of reliable feature points can improve the tracking process. However, approaches using fixed thresholds are often unable to cope with the varying environmental parameters in visual odometry. In this work we propose to utilize an adaptive strategy to dynamically adjust the quality-control threshold for feature matching according to the survival rate of tracked features.

The adaptive strategy sets forth two constraints derived from the epipolar geometry: a scale constraint and an angular constraint, to facilitate the accurate matching of features. The main steps in the proposed method are illustrated in Figure 3.

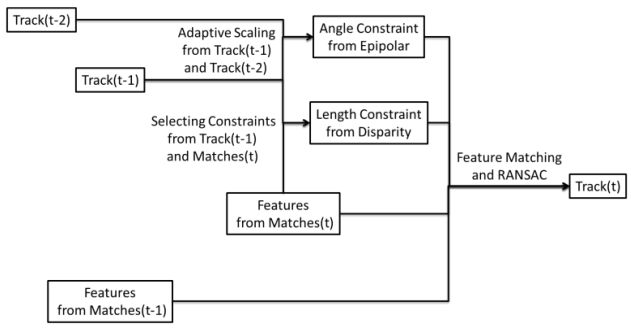


Figure 3. Steps involved in the proposed adaptive feature matching strategy.

A. Scale constraint

The difference in the distances between two matched features in one image and the next image usually varies in a gradual manner in a video sequence that has been recorded continuously in real-time. Therefore, the distance can be used to determine the reliability of tracked features. The pixel distances of two matched features as well as the measured depth values are combined to provide the scale cost function

$$\epsilon_{\text{scale}}(\chi,\chi') = z(\chi)\|\rho(\chi)-\rho(\chi')\| \qquad (4)$$

where χ and χ' are corresponding matching features, ρ(·) and z(·) denote the image coordinates and depth value of a feature respectively. This model expects small movements of tracked image features which are further away, yet allows a closer point to have a larger change. The compensation of depth reflects to the nature of perspective projection.

B. Angular constraint

The epipolar constraint imposes scale and angular limitations on matched features. The angular cost function can be defined by

$$\epsilon_{angle}(\chi,\chi') = \tau(\rho(\chi'),\rho(\chi),e_{t-1}) \qquad (5)$$

where χ and χ' are corresponding matching features, ρ(·) denotes the image coordinates of the features, τ represents the angle defined by three image points, and e_(t-1) is the last known position of epipole. The cost function in Eq. (5) constrains a matched feature χ' to a reflective fan-shaped area centred at χ.

C. Adaptive adjustment

The constraints introduced in the previous subsections are combined into a summarised term to indicate if a match of features should be rejected or accepted for the final decision. Given a set of cost functions Θ (in this work Θ={ϵ_scale,ϵ_angle }) and a candidate feature match $\chi \rightarrow \chi$ ^'∈M_t, where M_t denotes the set of accepted feature mappings from frame t-1 to frame t, the summarised cost is defined as

$$C_t(\chi,\chi',\Theta) = \begin{cases}\infty, \exists\epsilon\in\Theta \text{ s.t. } \Delta_t(\chi,\chi',\epsilon) > N_t\\ \|\nu(\chi)-\nu(\chi')\|, \quad \text{otherwise}\end{cases} \qquad (6)$$

where ν(χ) denotes the normalised vector representation of an image feature and Nt is the quality controlling parameter which is updated each time based on the ratio of previously accepted feature mappings from frame t-2 and frame t-1.

The quality parameter is relaxed to tolerate a larger room of error in evaluating matches when the number of tracked features is decreasing, and is set stricter as the number of matched features raises. In this way, the balance between quality and survival rate of tracked features can be achieved in an adaptive manner.

IV. Experiment Results

The proposed method is compared with a conventional implementation that also uses the epipolar constraint to filter out incorrect feature matches. The proposed method is tested using KITTI benchmark suite [10]. The suite includes real-world sequences collected in urban areas by a vehicle equipping with

The authors would like to thank the Ministry of Science and Technology in Taiwan for sponsoring this work under grant numbers MOST 104-2221-E-390-020 and the National Center for High-performance Computing, National Applied Research Laboratories for sponsoring this work under grant number 03104A7100

four calibrated cameras, an inertial measurement unit (IMU) and a global positioning system (GPS).

The sequence used for testing has been taken by stereo cameras and consists of 107 continuous image frames. The acquired sequence covers a driving distance of approximately 107 m. The result is compared with the motion as measured by the GPS and IMU mounted on the moving vehicle. Figure 4 shows the egomotion calculated by the proposed approach.

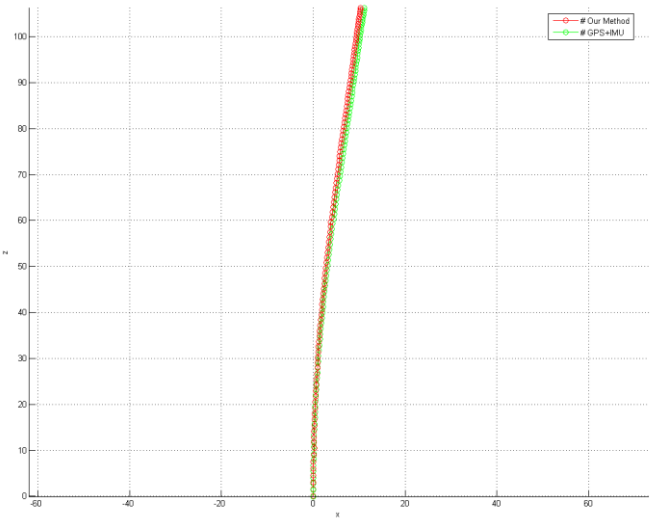


Figure 4. Egomotion estimated by the proposed method (red) and motion measured by GPS and IMU (green).

As can be seen from Figure 4, the estimated path is quite close to that of the ground truth, which demonstrates that the proposed method is effective in estimating the egomotion from stereo images alone, without other measurement devices.

By integrating the egomotion of the camera system estimated using the proposed method and the 3D coordinates of feature points calculated from the stereo image pairs, we can construct a 3D model along the cameras' path. Figure 5 shows the resultant 3D scene reconstructed from the egomotion estimated. The egomotion estimated by the proposed method is shown in red and motion measured by GPS and IMU is shown in green for comparison.

The proposed method is applied to four datasets from the KITTI database. The translation and rotation errors between the estimated egomotion and the given ground truths are given in Table 1. From the table, it can be seen that the average translation error is about 2.9%; in metric terms, it means that a movement of 1 m has an average error of 0.029 m. Also, the average rotation error is about 0.0156 radians per meter. By referring to the list of algorithm performances at the KITTI website, our results are comparable to the top 10 to 20 algorithms. In addition, some of the top algorithms utilize input measurements from alternative sensory devices, whereas the proposed algorithm computes the camera system's egomotion from the stereo image sequences alone. Thus, it shows that the proposed method is able to provide significantly good egomotion estimation by improving the accuracy of feature tracking.

To further demonstrate the performance of the proposed algorithm, we compare it with conventional egomotion estimation approaches with and without bundle adjustment (BA) [11]. Bundle adjustment is often used to optimize the final results and can often improve the outcome of egomotion estimation. Table 2 provides the reprojection root-mean square error of the proposed method with conventional approaches. From Table 2, it can be seen that the proposed method has better performance over conventional approaches. The results obtained by the proposed method can also be further improved by optimizing the calculated egomotion using bundle adjustment.

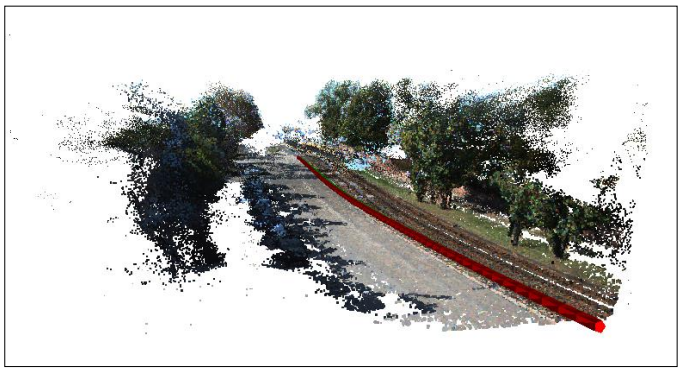


Figure 5. 3D scene reconstruction based on the egomotion estimated by the proposed method (red) and motion measured by GPS and IMU (green).

TABLE 1: TRANSLATION AND ROTATION ERRORS IN ESTIMATED EGOMOTION FOR THE PROPOSED METHOD.

Dataset #	TRANSLATION ERROR (%)	ROTATION ERROR (DEG/M)
1	1.08	0.0043
2	1.72	0.0062
3	5.92	0.0381
4	2.70	0.0138
Average	2.855	0.0156

TABLE 2: COMPARISON OF REPROJECTION RMS ERROR

Method	Error (pixels)
Conventional	29.55
Conventional with BA	3.21
Proposed Method	0.69
Proposed Method with BA	0.15

V. CONCLUSIONS

In this paper we propose an improved feature tracking algorithm for visual odometry. The tracking adaptively adjusts the strictness in matching features between two consecutive frames, according to the change of feature survival rate over time. The proposed method differs from typical visual odometry implementation by excluding wrong correspondences while maintaining good balance in feature

numbers. Comparitive experiments have been conducted to evaluate the performance of the proposed method. From the experimental results, it can be seen that the proposed method outperforms conventional methods and is able to achieve accurate egomotion estimation by improving the quality of feature tracking. In addition, according to the comparitive results provided on the KITTI benchmark website, the performance of the proposed algorithm is comparable to the top 10 to 20 algorithms from around the world. Based on the results obtain in this paper, we will extend the research further to develop a LiDAR-enabled visual odometry framework in the future. In additon we also aim to improve the error measures used in optimization by taking into account more factors, such as age of tracked feature and error covariance.

REFERENCES

[1] H. Moravec, "Obstacle Avoidance and Navigation in the Real World by A Seeing Robot Rover," in Tech. Report CMU-RITR-80-03, Robotics Institute, Carnegie Mellon University, Sep, 1980.

[2] J. Shi and C. Tomasi, "Good Features to Track," in Proc. Inter-national Conference on Pattern Recognition 1994, pp 539-600, June 1994.

[3] B. Kitt, A. Geiger, and H. Lategahn, "Visual Odometry Based on Stereo Image Sequences with RANSAC-based Outlier Rejection Scheme," in Proc. Intelligent Vehicles Symposium 2010, pp. 486-492, June 2010.

[4] H. Badino, A. Yamamoto, and T. Kanade, "Visual Odometry by Multi-frame Feature Integration", in Proc. First International Workshop on Computer Vision for Autonomous Driving at ICCV. 2013.

[5] F. Bellavia, M. Fanfani, F. Pazzaglia, and C. Colombo, "Ro-bust Selective Stereo SLAM without Loop Closure and Bundle Adjustment," in Proc. 17th International Conference on Image Analysis and Processing (ICIAP'13), pp 462-471, 2013.

[6] C. F. Olson, L. H. Matthies, M. Schoppers, and M. W. Maimone. "Rover Navigation using Stereo Ego-motion. Robotics and Autonomous Systems," Vol. 43, No. 4, pp 215-229, June 2003.

[7] [7] M. Pollefeys, D. Nister, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, and H. Towles, "Detailed Real-time Urban 3D Reconstruction from Vide,." International Journal of Computer Vision (IJCV), Vol. 78, No. 2-3, pp. 143-167, July 2008.

[8] D. Scaramuzza and F. Fraundorfer, "Visual Odometry Part I: The First 30 Years and Fundamentals," IEEE Robotics and Automation Society, Vol. 18, No. 4, pp. 80-92, 2011.

[9] Z. Zhang, "A Flexible New Technique for Camera Calibration," In Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 11, pp.1330-1334, 2000.

[10] J. Fritsch, T. Kuehnl and A. Geiger, "A New Performance Measure and Evaluation Benchmark for Road Detection Algorithms," in Proc. International Conference on Intelligent Transportation Systems 2013. 2013.

[11] M. I. A. Lourakis, A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," Journal of ACM Trans-actions on Mathematical Software, Vol. 36, No. 1, pp. 1-30, 2009.

Blocking artifact reduction in frame rate up-conversion

Nam-Uk Kim
Department of Computer Engineering, Sejong University
Digital Media System Laboratory
Seoul, Republic of Korea
nukim@sju.ac.kr

Yung-Lyul Lee
Department of Computer Engineering, Sejong University
Digital Media System Laboratory
Seoul, Republic of Korea
Corresponding author
yllee@sejong.ac.kr

***Abstract*— a new technology for video frame rate up-conversion (FRUC) is presented by combining an occlusion detection (OD) method with blocking artifact detection. First motion estimation (ME) is performed to obtain a motion vector. Then, the OD method is used to refine the MV in the occlusion region. If the current block is occluded region, blocking artifact detection is performed. If blocking artifacts are strong, then the block is partitioned into smaller blocks. The experimental results show that the proposed algorithm provides better PSNR (Peak Signal to Noise Ratio) values than the conventional approaches.**

Keywords—component; frame rate up-conversion; occlusion detection; median filter; bi-directional motion compensation; blocking artifacts

I. INTRODUCTION

Frame rate up-conversion is one of the main issues that have been discussed in recent years. It is used to convert video/film materials with a low frame rate to ones with a high frame rate so that they can be displayed with smooth motion and high perceived quality. FRUC is a technique which inserts an interpolated frame between two adjacent original frames [1], [2]. The existing FRUC algorithms can be divided into two categories. First, simple FRUC algorithms such as frame repetition and temporal linear interpolation [3], but these methods usually cause ghost artifacts and blurring artifacts. Therefore, in the second category, motion compensated techniques are used to reduce these artifacts. Such methods are called motion-compensated frame rate up-conversion (MC-FRUC) algorithms.

Many MC-FRUC algorithms have been proposed [4]-[11], in which full search ME is performed on the previous and current frames to obtain motion vectors and then the motion vectors are used to reconstruct the interpolated frame. The performance of MC-FRUC depends on the accuracy of ME and motion-compensated interpolation (MCI). The conventional approach to MC-FRUC utilizes unidirectional motion fields [4], [5]. A typical algorithm divides one of the two adjacent frames into blocks, estimates the motion vector of each block with respect to the other frame, and interpolates new frames along the motion vectors. However, this causes holes and overlaps of blocks in the interpolated frame. To solve this problem, BDMC (Bi-Directional Motion Compensation) is proposed using bidirectional motion fields [6], [7]. This approach divides the frame to be interpolated into blocks before it is actually created. Each block has two motion vectors, one pointing to the previous frame and the other to the current frame. The pixels in the block are interpolated by motion compensation using these two motion vectors. The motion vectors are either estimated using an algorithm of bidirectional motion estimation [8], [9], or derived from a unidirectional motion field. However, there are two essential problems that affect the performance of BDMC. These are blocking artifacts and the lack of information on occlusion. An occluded area exists in only one of the two existing frames and, therefore, should be interpolated using only one of the frames. To restrict the limitations imposed by this problem, a new MC-FRUC method that involves occlusion detection and blocking artifact detection is proposed.

II. CONVENTIONAL FRUC ALGORITHMS

In MC-FRUC, block-based ME is performed to find the motion vector which minimizes the sum of absolute differences (SAD) between the current block and prediction blocks. A few well known MC-FRUCs are introduced in this Section.

A. Median Filter

Motion vectors are not always valid for every pixel or object in a frame. Thus, visible artifacts occur wherever such wrong motion vectors are used in the frame. One effective method of solving this problem is to use a median filter [9]. In this method, a wrongly interpolated pixel is either substituted or averaged with its neighboring pixels.

Motion compensated averaging is especially vulnerable when wrong motion vectors are used to interpolate a stationary area. Fig. 1 shows the 16x16 block-based BDMC process. The block position (m, n), which is a multiple of 16 horizontally and vertically, in the interpolated frame, f_{IP}, utilizes the motion vector from the collocated block position (m, n) in the current frame, f_t, to the previous frame, f_{t-1}, as drawn as motion vector in Fig. 1. The dark block in the block position (m, n) in f_{IP} is interpolated (averaged) by the two blocks drawn with dashed lines, as shown in Fig. 1, in the previous frame, f_{t-1}, and the current frame, f_t, by using +MV/2 and –MV/2, respectively.

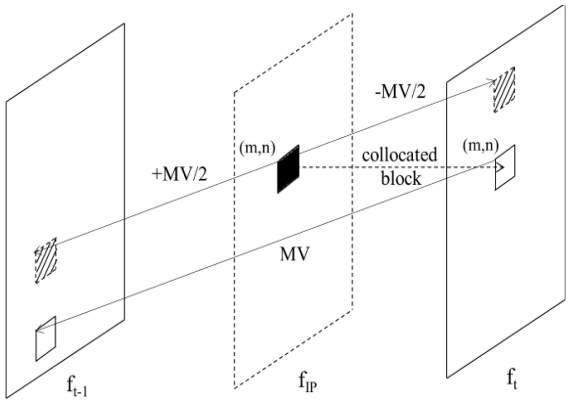


Fig. 1. BDMC (Bi-Directional motion compensation) process

Static median filtering (SMF) will take the median pixel value of the two non-motion-compensated pixels and one BDMC pixel within the block position (m, n) in f_{IP} as follows:

$$\text{value1} = f_{t-1}(x, y) \quad (1)$$

$$\text{value2} = f_t(x, y)$$

$$\text{value3} = \{f_{t-1}(x + MV_x/2, y + MV_y/2) + f_t(x - MV_x/2, y - MV_y/2)\}/2$$

$$f_{IP}(x, y) = \text{median (value1, value2, value3)}$$

where (x, y) is the pixel value within the block position (m, n) in f_{IP}, $f_{t-1}(x, y)$ and $f_t(x, y)$ are the pixel values in the collocated position (x, y) of f_{t-1} and f_t. Equation (1) signifies that SMF considers two collocated pixels in f_{t-1} and f_t and one pixel by BDMC. This method will be effective in a stationary area, but it is not suitable for moving object areas.

Dynamic median filter (DMF) is similar to SMF, but takes the median value of the two motion-compensated pixels shown as the dashed blocks in Fig. 1 and the average value of two non-motion-compensated (collocated) pixels as follows:

$$\text{value 1} = f_{t-1}(x + MV_x/2, y + MV_y/2) \quad (2)$$

$$\text{value 2} = f_t(x - MV_x/2, y - MV_y/2)$$

$$\text{value 3} = \{f_{t-1}(x, y) + f_t(x, y)\}/2$$

$$f_{IP}(x, y) = \text{median (value 1, value 2, value 3)}$$

In the case where the forward and backward motion vectors are accurate, the compensated pixels will have quite similar luminance value, thus DMF will select one of them. However, if the motion vectors are unreliable, the compensated inputs may be discarded by DMF. One drawback of DMF is visible artifacts due to wrong motion compensation.

III. PROPOSED METHOD

In MC-FRUC, block-based ME is preformed to find the motion vector, among the candidate motion vectors, which minimizes the SAD between the current block and prediction blocks. However, motion vectors in the occlusion area are not reliable and lead to annoying blocking artifacts. In addition, the occlusion problem usually happens along the object boundary, where the motion vectors are not continuous and have some value differences compared with those of the spatial neighboring blocks. To overcome the existing MC-FRUC problem, the proposed method refines the pixel values in the occlusion area. If the proposed occlusion detection method indicates that occlusion occurs, find blocking artifacts and remove artifacts with de-blocking filter with median filtered interpolated pixels. Otherwise, BDMC is used after obtaining the bidirectional motion vectors. The occlusion detection method and blocking artifacts detection method are introduced in the next section.

Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar:

A. Occlusion Detection Algorithm

Occlusion problems are a key issue in MC-FRUC. If the motion vector can be properly estimated, the interpolation is carried out simply along the motion trajectory. However, if the pixel to be interpolated is only available in the previous or current frame in the occlusion area, it is not easy to find a suitable pixel. A spatial motion vector analysis of the neighboring block in the interpolated frame can detect the occlusion area. Therefore, the median value among Value 1, Value 2 and Value 3 in Equation (2) is selected when occlusion occurs.

A full search ME based on 16x16 block with 32 search range is performed to find the motion vector having the minimum SAD value for each block. After computing the motion vectors in the ME process, the OD method is performed in a block-based manner in the interpolated frame to refine and enhance their accuracy. DMF is applied if occlusion occurs. Otherwise, BDMC is applied. The occlusion detection function is shown below

$$T_x = abs\,(MVx\,(m, n - 1) + MVx\,(m - 1, n)) \quad (3)$$

$$T_y = abs\,(MVy\,(m, n - 1) + MVy\,(m - 1, n))$$

$$T = abs\,(T_x - T_y)$$

If ($abs\,(MVx\,(m, n - 1) - MVx\,(m, n)) > T \;||$

$abs\,(MVy\,(m - 1, n) - MVy\,(m, n)) > T$)

Return true

Else

Return false

where MVx (m, n) and MVy (m, n) are the motion vector components in the x and y directions and the abs function returns the absolute value.

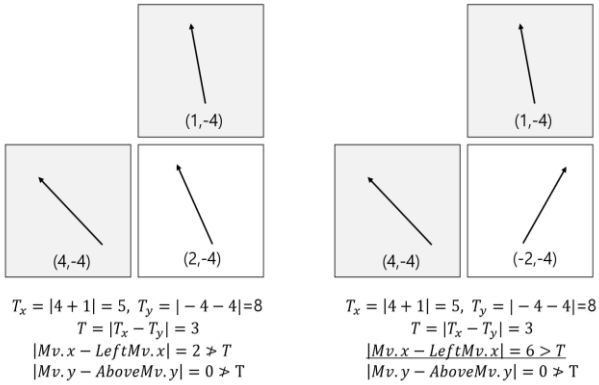


Fig. 2. Example of occlusion detection

B. Blocking artifact detection in frequency domain

Occlusion detection method alleviates lack of motion information in the occluded region, but degradations like blocking artifacts and ghost artifacts is induced in the occlusion area. To reduce these artifacts, blocking artifact detection method in frequency domain is proposed.

When the current block is decided as an occluded region, the block usually produces blocking artifacts because the block producing strong blocking artifacts have the wrong vector or motion discontinuity. To find the appropriate motion vector, the current NxN block is divided into small N/2xN/2 blocks and block-based motion estimation process is performed again for each N/2xN/2 block. This process is continued until 4x4 block. If the smallest block has blocking artifacts, then strong de-blocking filter is applied to the left and top side pixels. Otherwise, weak de-blocking filter is applied to those pixels.

Conceptual 1-Dimensional view of the pixel values producing strong blocking artifacts are shown in Fig. 3.

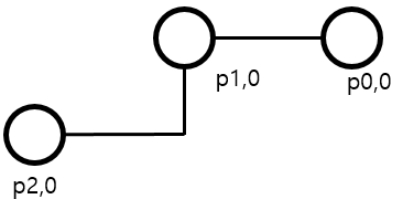


Fig. 3. Pixel values with strong blocking artifact

To detect strong blocking artifacts shown in Fig. 3, the block boundary is investigated by using 1-D DCT-2 kernel shown in Eq. (4) to detect blocking artifacts. The energy of transform coefficients being concentrated on the secondary element of 1-D DCT, as shown in Fig. 4, can detect the information of blocking artifacts. The strong and weak deblocking filtering in HEVC(High Efficiency Video Coding) are applied to the block boundary according to the strong and weak blocking artifacts.

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad k = 0, \ldots, N-1. \tag{4}$$

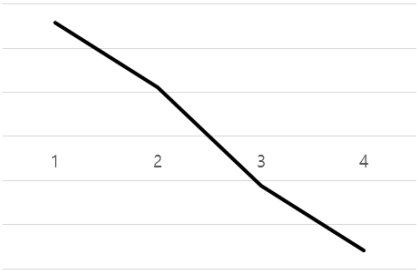


Fig. 4. DCT-2 kernel's secondary elements

IV. Experimental Results

Twenty four (24) test sequences, in which each sequence has 100 frames, are used in the experiments. The number of original video sequences is down-sampled by a factor of 2 to compare the proposed interpolation frames with the original dropped frames. The overall performance of the proposed method is slightly better than that of the others.

TABLE I. Experimental conditions.

Block Size	16x16 pixels
Search Range	±32 pixels
Video Resolution	2560x1600 ~ 352x288

TABLE II. Comparison of PSNR results.

Video sequences	DMF[2]	BDMC[6]	Proposed method
Traffic	35.71	35.41	**36.07**
PeopleOnStreet	24.90	25.98	**26.07**
Nebuta	25.58	25.29	**26.02**
SteamLoco	34.50	34.29	**35.03**
Kimono	31.36	32.04	**32.24**
ParkScene	32.86	32.52	**33.17**
Cactus	30.58	31.15	**31.36**
BasketballDrive	27.23	27.43	**27.60**
BQTerrace	29.43	29.28	**29.91**
BasketballDrill	28.81	29.14	**29.28**
BQMall	28.42	**29.12**	**29.12**
PartyScene	29.01	**30.02**	29.84
RaceHorsesC	25.42	25.26	**25.72**
BasketballPass	32.41	32.16	**32.66**
BQSquare	34.67	**35.91**	35.05
BlowingBubble	31.90	**32.12**	32.08
RaceHorses	26.43	26.30	**26.95**
FourPeople	39.96	40.26	**40.33**
Johnny	40.10	**40.90**	40.72
KristenAndSara	41.60	**41.94**	41.93
BasketballDrill Text	28.69	29.11	**29.15**

ChinaSpeed	25.16	**25.63**	25.56
SlideEditing	41.86	**43.45**	42.58
SlidShow	52.03	**52.67**	52.43

V. CONCLUSION

In this paper, the proposed FRUC using Occlusion detection and blocking artifacts detection method shows improved image quality in the interpolated frames compared with the other methods. However, when observing the interpolated frames carefully, blurring can still be seen in the interpolated frames. Therefore, further research is necessary in order to reduce blurring in moving object areas.

ACKNOWLEDGMENT

This research was in part supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(Ministry of Science, ICT and Future Planning) (NRF-2015R1A2A2A01006085)

REFERENCES

.

[1] C. Cafforio, F. Rocca, and S. Tubaro, "Motion compensasted image interpolation," IEEE Trans. Commun., vol.38, no.2, pp.215-222, Feb. 1990.

[2] O.A. Ojo and G. de Haan, "Robust motion-compensated video up conversion," IEEE Trans.Consum. Electron., vol.43,no.3,pp.1045-1056,Nov. 1997.

[3] K. Hilman, H.-W. Park, and Y.-M Kim, "Using motion-compensated frame rate up conversion for the correction of 3:2 pull down artifacts in video sequences," IEEE Trans. Circuits Syst. Video Technol., vol.10, no.6, pp.839-877, Sept. 2000.

[4] A.-M Huang and T.Q. Nguyen, "A multistage motion vector processing method for motion-compensated frame interpolation," IEEE Trans. Image Process., vol.17, no.5, pp.649-708, May 2008.

[5] Y.Ling, J. Wang, Y.Liu, and W. Zhang, "A novel spatial and temporal correlation integrated based motion-compensated interpolation for frame rate up-conversion." IEEE Trans. Consum. Electron., vol.54, no.2, pp.863-869, May 2008.

[6] W. Demin, L. Zhang, and A. Vincent, "Motion-compensated frame rate up-conversion part II: New algorithms for frame interpolation," IEEE Trans. Broadcast., vol.56, no.2, pp.142-149, June 2010

[7] G. Dan and T. Nguyen, "Motion vector processing for frame rate up conversion," Proc, IEEE Int. Conf. Acoustics, Speech, and Signal Processing, 2004, pp.309-312, 2004.

[8] T. Ha, S. Lee, and J. Kim, "Motion compensated frame interpolation by neew block-based motion estimation algorithm," IEEE Trans. Consum. Electron., vol.50, no.2, pp.752-759, May 2004.

[9] B.-T. Choi, S.-H. Lee, and S.-J. Ko, "New frame rate up-conversion using bidirectional motion estimation," IEEE Trans. Consum. Electron., vol.46, no.3, pp.603-609, Aug. 2000.

[10] H. Lui, R. Xiong, D. Zhao, S. Ma, and W. Gao, "Multiple hypotheses Bayesian frame rate up-conversion by adaptive fusion of motion-compensated interpolations," IEEE Trans. Circuits Syst. Video Technol., vol.22, no.8, pp.1188-1198, Aug. 2012.

[11] S.-G. Jeong, C.Lee, and C.-S. Kim, "Motion-compensated frame interpolation based on multihypothesis motion estimation and texture optimization," IEEE Trans. Image Process., vol.22, no.11, pp.4497-4509, Nov. 2013.