MIPRO 2014, 26-30 May 2014, Opatija, Croatia

Reversible De-Identification for Lossless Image Compression using Reversible Watermarking

Reuben A. Farrugia
*Department of Communications and Computer Engineering,
University of Malta,
Msida, Malta
reuben.farrugia@um.edu.mt

Abstract - De-Identification is a process which can be used to ensure privacy by concealing the identity of individuals captured by video surveillance systems. One important challenge is to make the obfuscation process reversible so that the original image/video can be recovered by persons in possession of the right security credentials. This work presents a novel Reversible De-Identification method that can be used in conjunction with any obfuscation process. The residual information needed to reverse the obfuscation process is compressed, authenticated, encrypted and embedded within the obfuscated image using a two-level Reversible Watermarking scheme. The proposed method ensures an overall single-pass embedding capacity of 1.25 bpp, where 99.8% of the images considered required less than 0.8 bpp while none of them required more than 1.1 bpp. Experimental results further demonstrate that the proposed method managed to recover and authenticate all images considered.

I. Introduction

Video surveillance cameras are becoming ubiquitous in many developed countries. This has raised several privacy concerns which have pushed policy makers to regulate their use. One approach to provide privacy is to obfuscate sensitive regions within an image/video which prevents the identification of the persons being captured. The authors in [1,2] have proposed an irreversible obfuscation method which however, prevents the use of the captured videos from aiding criminal investigation or to be used as evidence in court [3].

Reversible De-Identification is a process which, while still concealing the identity of individuals, enables persons in possession of high security credentials to recover the original multimedia content containing private information. The authors in [4] encode the region of interest (ROI) and background in separate data layers using JPEG2000. On the other hand, the authors in [5,6] employ encryption strategies directly on the pixel intensities of the ROI. However, these methods completely destroy the naturalness of the captured video.

A ROI transform-domain scrambling technique was presented in [7,8,9] for different image/video compression standards. The scrambling process better maintains the naturalness of the video. However this method is less secure since it reveals the intensity levels of the original content. Moreover, the obfuscation and reversibility processes are dependent on each other, and thus cannot be used in conjunction with other obfuscation methods. Non-reversible watermarking was adopted in [3,10] to solve the latter issue and embed the information needed to recover the De-Identified region within the video itself. However, both these schemes are irreversible since the noise introduced by the watermark embedding process is permanent. Moreover, these schemes have registered a substantial reduction in compression efficiency. The authors in [11] employ reversible watermarking to solve the former issue. However, this method induces significant distortions within the obfuscated image themselves.

This work presents a Reversible De-Identification method for lossless images. This approach adopts Reversible Watermarking to make the system reversible. The proposed solution is completely independent from the obfuscation process, and is thus generic. Nonetheless, this work employs the *k*-Same obfuscation process, which ensures *k*-anonymity, to obfuscate the face of frontal images. The difference between the original and obfuscated image is compressed, authenticated, encrypted and embedded within the obfuscated image itself. This method keeps the naturalness of the obfuscated images while the original image can only be recovered by individuals having the proper encryption key. The Reversible Watermarking schemes adopted in this work were found to outperform existing state-of-the-art schemes. Furthermore, experimental results demonstrate that the proposed scheme can recover and authenticate all obfuscated images considered.

This paper is organized as follows: Section II provides a high-level description of the proposed system and introduces the notation used in the latter sections. The Forward and Inverse Reversible De-Identification processes are explained in more detail in sections III and IV with the experimental results delivered in section V. The final comments and concluding remarks are drawn in section VI.

II. System Overview

Fig. 1 illustrates the schematic diagram of the Forward Reversible De-Identification process which receives the original image $\boldsymbol{I}$ and conceals the face of the person using the *Face Obfuscation* process to generate an obfuscated image $\boldsymbol{I}_{\Theta}$. This work considers color images using the YC_bC_r color space. The coordinates of the top left corner and bottom right corner of the De-identified region is enclosed within the bounding box $\boldsymbol{\beta}$, which is passed to both *ROI Extraction* processes to extract the

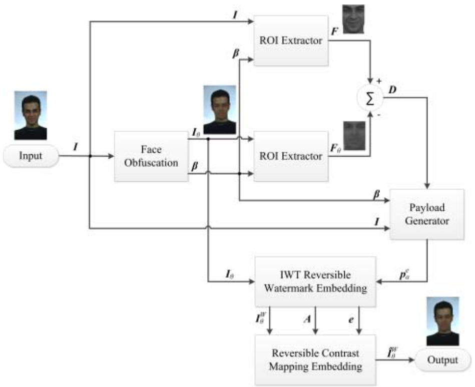

Figure 1. Schematic diagram of the Forward Reversible De-Identifcation Process

face image $\boldsymbol{F}$ and the obfuscated face image $\boldsymbol{F}_{\Theta}$. The face images are then subtracted to derive the difference face image $\boldsymbol{D}$.

The *Payload Generator* process is then used to convert the difference face image $\boldsymbol{D}$ and bounding box $\boldsymbol{\beta}$ into a packet $\boldsymbol{p}_a^e$ which is authenticated and encrypted. The packet $\boldsymbol{p}_a^e$ is then embedded within the obfuscated image $\boldsymbol{I}_{\Theta}$ using the *Integer Wavelet Transform* (*IWT) Reversible Watermark Embedding* process (1st level) which generates the embedded image $\boldsymbol{I}_{\theta}^{W}$, the auxiliary information $\boldsymbol{A}$ and the residual bitstream $\boldsymbol{e}$. This method provides a good compromise between capacity and distortion. However, additional information might be needed at the receiver to resolve overflow and underflow issues. The *Reversible Contrast Mapping Embedding* process (2nd level) is therefore used to embed this information ($\boldsymbol{A}$ and $\boldsymbol{e}$) within the embedded image $\boldsymbol{I}_{\theta}^{W}$, which usually corresponds to few bits, and generates the second level embedded obfuscated image $\hat{\boldsymbol{I}}_{\theta}^{W}$. This method is ideal since it does not need additional information to resolve overflow/underflow issues. Moreover, the distortions introduced at low bitrates is generally negligible. However, its performance significantly degrades at higher bitrates and is therefore not suitable to embed large payloads.

Fig. 2 depicts the schematic diagram of the Inverse Reversible De-Identification process. The second order embedded obfuscated image $\hat{\boldsymbol{I}}_{\theta}^{W}$ is inputted to the *Reversible Contrast Mapping Extraction* process which extracts the first level embedded obfuscated image $\boldsymbol{I}_{\theta}^{W}$ together with the auxiliary information $\boldsymbol{A}$ and the residual bitstream $\boldsymbol{e}$. The *IWT Reversible Watermark Extraction* process is then used to extract the original payload $\boldsymbol{p}_a^e$ and original obfuscated image $\boldsymbol{I}_{\Theta}$. The *Inverse Payload Generator* reverses the process of the *Payload Generator* and recovers the difference image $\boldsymbol{D}$ and the bounding box $\boldsymbol{\beta}$, which is used by the *ROI Extractor* process to extract the obfuscated face $\boldsymbol{F}_{\Theta}$. The difference image $\boldsymbol{D}$ and obfuscated face $\boldsymbol{F}_{\Theta}$ are then summed to derive the original face $\boldsymbol{F}$, which is used by the *ROI Replacement* process to recover the original image $\boldsymbol{I}_{rec}$.

It is important to notice at this stage that the packet $\boldsymbol{p}_a^e$ is authenticated and encrypted, and therefore the difference image $\boldsymbol{D}$ and bounding box $\boldsymbol{\beta}$ can only be recovered correctly by persons in possession of the correct security key. The embedding processes are chosen in order to provide minimal distortion so that it maintains the naturalness of the obfuscated image. Moreover, the authentication process ensures that the original image is recovered and ensures that the image is not modified.

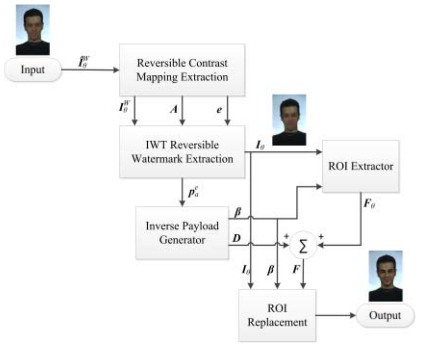

Figure 2. Schematic diagram of the Inverse Reversible De-Identifcation Process

III. Forward Reversible De-Identification

A. Face Obfuscation

The *Face Obfuscation* process receives the original image $\boldsymbol{I}$ and detects the face region and eye locations using the ground truth information available in the color FERET dataset. This can be automated using the face detector in [12] and the eye detector in [13] which ensure high accuracies. However, the main contribution of this work is to present a Reversible De-Identification method which is independent from the obfuscation process. Thus, the automation of the face and eye detectors is not in the scope of this work.

The upper left and bottom right coordinates of the face region are included in the bounding box $\boldsymbol{\beta}$ and used to extract the face $\boldsymbol{F}$ which is aligned using affine transformations [14]. The aligned face image $\boldsymbol{F}$ is then concealed using the *k*-same algorithm, which computes the average face derived over the *k* closest aligned faces in Eigen-space, to generate the obfuscated aligned face image $\boldsymbol{F}_{\Theta}$. More information on the *k*-same algorithm can be found in [2]. The obfuscated face image $\boldsymbol{F}_{\Theta}$ is then re-aligned to match the orientation of the original face image $\boldsymbol{F}$ using affine transformations and then overwrites the face region in the original image $\boldsymbol{I}$ to derive the obfuscated image $\boldsymbol{I}_{\Theta}$.

B. ROI Extraction

The *ROI Extraction* process is a simple algorithm which employs the bounding box coordinates $\boldsymbol{\beta}$ to identify the region to be cropped from the input image $\boldsymbol{I}$ (or $\boldsymbol{I}_{\Theta}$). The cropped sub-image is then stored in the face image $\boldsymbol{F}$ (or obfuscated face image $\boldsymbol{F}_{\Theta}$).

C. Payload Generator

The *Payload Generator* Process receives the difference image $\boldsymbol{D}$ which is compressed using the predictive coding method presented in [15] followed by the Deflate algorithm [16]. The original image $\boldsymbol{I}$ is authenticated using SHA-1 which generates a 20-Byte *Hash*. The *Hash* will be used by the Inverse Reversible De-Identification process to ensure that it recovers the original image $\boldsymbol{I}$, and is thus appended to the *Payload*. The bounding box coordinates $\boldsymbol{\beta}$ are also required at the receiver to identify the face region and are therefore included as information within the header. The resulting packet $\boldsymbol{p}_a$, illustrated in Fig. 3, was then encrypted using AES-128 to generate the encrypted packet $\boldsymbol{p}_a^e$.

$\boldsymbol{\beta}$	Payload	Hash

Figure 3. The authenticated packet $\boldsymbol{p}_a$.

D. IWT Reversible Watermarking Embedding

The *IWT Reversible Watermarking Embedding* process first derives the number of decompositions N_{dec} needed to embed the packet $\boldsymbol{p}_a^e$ within $\boldsymbol{I}_\Theta$ using

$$N_{dec} = \begin{cases} \left\lceil -\frac{1}{2}\log_2(1-C) \right\rceil & \text{if } C < 1 \\ M & \text{Otherwise} \end{cases} \tag{1}$$

where M indicates the maximum number of decomposition allowed and C represents the capacity needed to embed $\boldsymbol{p}_a^e$ bits and is computed using

$$C = \frac{\left|\mathbf{p}_a^e\right|}{Ch \times W \times H} \tag{2}$$

where $|\ |$ represents the cardinality of the set, W and H represent the number of columns and rows in the image and Ch represents the number of color channels (in our case 3). This process then adopts the CDF(2,2) integer wavelet transform specified in [17] to decompose the image. This method employs *Forward Integer Wavelet Expansion* to embed the actual information while a novel *Threshold Selection* strategy is used to identify the set of thresholds which provide enough capacity while minimize the overall distortions. More information is provided in the following subsections.

1) Threshold Selection

The proposed *Threshold Selection* method is based on the observation that different sub-bands provide different levels of distortions [18]. However, in order to reduce the complexity of the optimization function, the following assumptions were made

- The chrominance sub-bands have similar properties and thus share the same threshold.
- The *HL* and *LH* sub-bands within the same color channel (luminance or chrominance) are assumed to have similar characteristics and therefore have the same threshold.

The number of thresholds to be considered by the *Threshold Selection* process is given by

$$N_T = 2(1 + N_{dec}) \tag{3}$$

In order to clarify this, consider a single level of decomposition. In this case $N_T = 4$ where T_1 is the threshold for coefficients in sub-band HH_1 of the luminance component, T_2 is the threshold for coefficients in sub-band HL_1 and LH_1 of the luminance component, T_3 is the threshold for coefficients in sub-band HH_1 in the chrominance components (C_b and C_r) and T_4 is the threshold for coefficients in sub-bands HL_1 and LH_1 in the chrominance components. Note that the subscript for the sub-bands represents the level of decomposition. The thresholds corresponding to sub-bands at higher level of decomposition are considered to be the same for the same color component. Therefore, if we consider a second level of decomposition ($N_T = 6$), the additional threshold T_5 controls the coefficients in sub-bands HH_2, HL_2 and LH_2 for the luminance component while threshold T_6 is responsible for the coefficients in sub-bands HH_2, HL_2 and LH_2 for the chrominance components. The same happens for higher levels of decompositions.

One naive approach is to use exhaustive search to find the optimal set of thresholds. However, this has a time complexity of the order of $O(n^{N_T})$ where n represents the search range. An alternative approach is to use a meta-heuristic approach to solve this optimization problem. This work employs Differential Evolution (DE) [19], which is a population based optimization algorithm, to derive the set of threshold which minimize a distortion criterion while ensuring that the capacity of the proposed system is sufficient to embed the message $\boldsymbol{s}$. The time complexity provided by this scheme is of the order of $O(G \times NP \times N_T)$, where NP is the population size and G corresponds to the number of generations.

Differential Evolution starts with a random set of possible solutions and tries to find better solutions through mutation and cross-over. The set of potential thresholds $\boldsymbol{\Delta}$ is a list of threshold NP vectors which are initialized using a uniformly distributed random number generator. The threshold vectors $\mathbf{T} = \boldsymbol{\Delta}_i$ which do not provide enough capacity are pruned and replaced by another random vector which satisfies it. The population of thresholds evolves over a number of generations using mutation and cross-over. The mutation process generates a mutant vector for every threshold vector contained within the population of NP vectors using

$$\mathbf{v}_i = \boldsymbol{\Delta}_{r1} + \alpha \times (\boldsymbol{\Delta}_{r2} - \boldsymbol{\Delta}_{r3}) \tag{4}$$

where $\{r1,r2,r3\} \in [0,NP\text{-}1]$ are mutually different integers which are different from index i, while α is a mutation factor that controls the magnitude of the differential variation. The cross-over process is then used to increase the diversity of the mutated vectors, and are included in another list of potential thresholds $\overline{\boldsymbol{\Delta}}_i$ according to

$$\overline{\boldsymbol{\Delta}}_i = \begin{cases} \mathbf{v}_{i,j} & \text{if } \upsilon \le \Gamma \\ \boldsymbol{\Delta}_{i,j} & \text{otherwise} \end{cases} \tag{5}$$

where υ is a uniformly distributed random number [0,1] and Γ represents the cross-over probability which has a constant value. The potential thresholds vectors $\overline{\boldsymbol{\Delta}}_i$ which do not provide enough capacity are pruned from the list.

The initial threshold vectors $\boldsymbol{\Delta}_i$ and mutated threshold vectors $\overline{\boldsymbol{\Delta}}_i$ are then used to embed the bitstream $\boldsymbol{s}$, where each time the threshold set $\mathbf{T}$ is included in the header of $\boldsymbol{s}$ (see Fig. 4). The distortion between the obfuscated image $\boldsymbol{I}_\Theta$ and the resulting expanded obfuscated image $\boldsymbol{I}_\Theta^W$ is computed using the weighted Peak Signal-to-Noise

Ratio (wPSNR) and is used to represent the fitness of the threshold **T** [20]. The *NP* threshold vectors which have the largest fitness values from the two sets are then grouped to replace the initial set **Δ** . The iterative process is terminated when either the maximum number of generations G_{max} is reached or else when the difference between the largest and smallest wPSNR within the list of thresholds **Δ** is smaller than some constant Φ.

2) *Forward Integer Wavelet Expansion*

The *Forward Integer Wavelet Expansion* process receives the set of thresholds **T** which are derived by the *Threshold Selection* process and encapsulates the packet $\boldsymbol{p}_a^e$ shown in Fig. 3 to generate the packet ***s*** to be embedded (Fig. 4). The field N_{dec} is a $\lceil \log_2 M \rceil$ bit field which specifies the number of decompositions used by the embedding process, **T** is a variable length field which specifies the threshold adopted (8-bit per threshold) and *L* is a 10-bit field which specifies the cardinality of $\boldsymbol{p}_a^e$ in bytes. This information is inserted as header information since the decoder needs it to reverse the process.

N_{dec}	**T**	L	$\boldsymbol{p}_a^e$

Figure 4. The actual bit stream to be embedded ***s***.

The packet ***s*** is inserted within the high frequency sub-bands of the integer wavelet coefficients. During the embedding process, a wavelet coefficient $w_{\mu,j}$ in sub-band μ is selected at random following a random permutation generated using the encryption key. A wavelet coefficient $w_{\mu,j}$ is considered for embedding if its magnitude is smaller than the threshold responsible of sub-band μ. A bit *b* is embedded using

$$\widehat{w}_{j,\mu} = 2w_{j,\mu} + b \tag{6}$$

The other wavelet coefficients are not considered for embedding. However, in order to prevent ambiguities at the receiver, they are expanded using

$$\widehat{w}_{j,\mu} = \begin{cases} w_{j,\mu} + \mathrm{T}_\mu & \text{if } w_{j,\mu} \geq 0 \\ w_{j,\mu} - \mathrm{T}_\mu + 1 & \text{Otherwise} \end{cases} \tag{7}$$

where T_μ is the threshold responsible for sub-band μ.This process is terminated once all the bits in ***s*** are embedded. In the rare event where this process fails to embed all bits, these bits are stored in a bit-sequence ***e***. The expanded obfuscated image $\boldsymbol{I}_\theta^W$ is then obtained using the inverse integer wavelet transform. The coordinates of pixels which encounter underflow/overflow issues and their corresponding values are included within the list of auxiliary information ***A***.

E. *Reversible Contrast Mapping*

The only problem with the proposed *Forward Integer Wavelet Expansion* process is that sometimes ***A*** and ***e*** are not empty. This work adopts the syntax shown in Fig. 5 to represent this information ***r*** to be embedded. The *Flag* is a 2-bit field which indicates whether ***A*** and ***e*** are empty or not. In case that one of them (or both) are not empty, the number of bits needed to embed the information in ***A*** (or ***e***) is signaled in N_A (or N_e). The fields N_A and N_e are encoded using 8-bits each while the size of ***A*** and ***e*** are variable length.

					Flag Values	
					A	***e***
				Flag	0	0
		Flag	$\boldsymbol{N_e}$	***e***	0	1
		Flag	$\boldsymbol{N_A}$	***A***	1	0
Flag	$\boldsymbol{N_A}$	***A***	$\boldsymbol{N_e}$	***e***	1	1

Figure 5. The packet ***r*** to be embedded within $\boldsymbol{I}_\theta^W$.

This work adopts the Reversible Contrast Mapping (RCM) [21] to embed the packet ***r*** within the watermarked obfuscated image $\boldsymbol{I}_\theta^W$. The main advantage of using RCM is that it embeds all information within the image without any ambiguities and provides an additional capacity of 0.5 bpp. However, the main limitation of the RCM is its limited capacity and that the distortion can become significant when embedding large payloads. However, the packet size ***r*** is expected to be very low (generally 2-bits).

The forward RCM transforms two neighboring pixel pairs (x, y) into (x', y') using

$$x' = 2x - y, y' = 2y - x \tag{8}$$

To prevent overflow and underflow, the transform is restricted to a sub-domain defined by the pixels which satisfy the conditions

$$0 \leq 2x - y \leq 255, 0 \leq 2y - x \leq 255 \tag{9}$$

The RCM scheme replaces the least significant bits (LSBs) of the transformed pairs (x', y'). The LSB of x' is used to indicate whether information is embedded within y' or not. A value of '1' indicates that information is embedded while a value of '0' can indicate two things. It can be that both pixel pair values (x, y) were odd or else that the pair are ambiguous and cannot be used for embedding. In the former case, the information bit is still embedded within the LSB of y'. On the other hand, the latter case cannot be used to embed information and thus the LSB of y' is not changed, while the true value of the LSB of x is inserted within the bit-sequence being embedded. More information about the RCM scheme can be found in [21].

IV. Inverse Reversible De-Identification

A. *Reversible Contrast Mapping Extraction*

The *Reversible Contrast Mapping Extraction* process receives the image $\hat{\boldsymbol{I}}_\theta^W$ and recovers $\boldsymbol{I}_\theta^W$ and ***r***. The information bit can be extracted from the LSB of y' when the LSB of x' is '1'. However, in the event when the LSB of x' is '0', both LSBs of x' and y' are forced to be odd and condition (9) is checked. If the condition is satisfied it then represents an odd pixel pair while if it does not it indicates that $y = y'$ and the original LSB value of x is extracted from the bitstream. More information about this is available in [21]. The auxiliary information ***A*** and residual bitstream ***e*** are then extracted from the packet ***r***. The original pixel values are recovered using

$$x = \left\lceil \frac{2}{3}x' + \frac{1}{3}y' \right\rceil, y = \left\lceil \frac{1}{3}x' + \frac{2}{3}y' \right\rceil \tag{10}$$

B. *IWT Reversible Watermarking Extraction*

The *IWT Reversible Watermarking Extraction* reverses the *IWT Reversible Watermarking Embedding* process and extracts the payload information $\boldsymbol{p}_a^e$ and the original obfuscated image $\boldsymbol{I}_\Theta$. It must be noted here that initially, the decoder has no knowledge about the number of decompositions employed N_{dec} and the threshold values $\mathbf{T}$. The decoder thus assumes that a single decomposition is employed and that the threshold values are set to zero. These values are then updated once they are extracted from the header information of $\boldsymbol{s}$. It is important that the encoder does the same thing during embedding in order to ensure synchronization between the encoder and decoder.

The watermarked obfuscated image $\boldsymbol{I}_\theta^W$ is then decomposed into N_{dec} levels using the CDF(2,2) integer wavelet transform and the wavelet coefficients of the high frequency sub-bands are scanned using the random permutation index generated using the encryption key. The bits of the packet are extracted from coefficients which satisfy the condition $-2T_{\mu,j}+1<\hat{w}_{\mu,j}<2T_{\mu,j}$ and are extracted from the LSB of $\hat{w}_{\mu,j}$ while the original wavelet coefficient is recovered using

$$w_{\mu,j}=\left\lfloor \frac{\hat{w}_{\mu,j}}{2} \right\rfloor \qquad (11)$$

The remaining coefficients are not used for embedding and are recovered using

$$w_{\mu,j}=\begin{cases}\hat{w}_{\mu,j}-\mathrm{T}_\mu & \text{if } w_{\mu,j}\geq 0\\ \hat{w}_{\mu,j}+\mathrm{T}_\mu\text{-}1 & \text{Otherwise}\end{cases} \qquad (12)$$

This method terminates once all the L bytes are extracted. The resulting image is then inverse CDF(2,2) transformed to recover the original obfuscated image $\boldsymbol{I}_\Theta$. The auxiliary information (if any) is then used to recover ambiguous pixel values while the residual bitstream $\boldsymbol{e}$ (if any) is appended to the recovered payload.

C. *ROI Replacement*

The *ROI Replacement* process replaces the region marked by the bounding box $\boldsymbol{\beta}$ with the recovered face image $\boldsymbol{F}$. The image $\boldsymbol{I}_{rec}$ can be authenticated by comparing the hash derived by computing the *SHA-1* on $\boldsymbol{I}_{rec}$ to the *Hash* value present in the tail of the packet $\boldsymbol{p_a}$.

V. Simulation Results

The results presented in this section consider two different sets of images. The first set was composed of the standard test images *Lena*, *Barbara*, *Aircraft* and *Mandrill* while the second set consisted of 2000 frontal images from the color FERET dataset. All images considered in this work were converted in the YC_bC_r color space using 4:4:4 sampling. The standard test images were used to evaluate the effectiveness of the proposed *Threshold Selection* process while the frontal images were used to evaluate the whole system.

The proposed algorithm has set the maximum number of decompositions M to 3 which ensures a single pass embedding capacity offered by the first level of watermarking of 0.9844 bpp. The Difference Expansion method was configured using values suggested in [19] and thus adopted α = 0.5, NP = 100 and Γ = 0.3. This paper does not claim that this corresponds to an optimal configuration, but claims that it provides performance superior to state of the art IWT threshold selection schemes such as [17] (see Fig. 6). These results clearly demonstrate that the proposed scheme manages to provide better quality of the stego image $\boldsymbol{I}_\theta^W$at different capacities. Simulation results further demonstrate that the proposed scheme needs on average 20 generations to converge. This correspond to 2000 invocations of the fitness function which is significantly less than the 255^{N_T} invocations needed by exhaustive search.

Fig. 7 demonstrates the cumulative density function (CDF) of the capacity needed to embed packet $\boldsymbol{s}$ within the obfuscated image using the set of 2000 frontal images from the color FERET dataset. It can be seen that a capacity smaller than 0.8 bpp is needed 99.8% of the time while they never require more than 1.1 bpp. It must be mentioned that the proposed scheme has a single-pass embedding capacity close to 1.25 bpp and is thus able to embed the information necessary to recover all images considered in this test. It is important to mention here that frontal images represent a very difficult scenario for our system since the area covered by the ROI is large in relation to the background. Lower capacities are expected when considering common surveillance scenarios. Simulation results further demonstrate that the residual bitstream $\boldsymbol{e}$ was empty for 99.8% of the time and the Auxiliary information $\boldsymbol{A}$ was empty for 99.85% of the time. This result confirms that most of the time the RCM reversible watermarking scheme embeds just 2-bits within the obfuscated image. Moreover, the additional capacity needed in these circumstances was at most 0.015bpp, which is very small and provides negligible distortions.

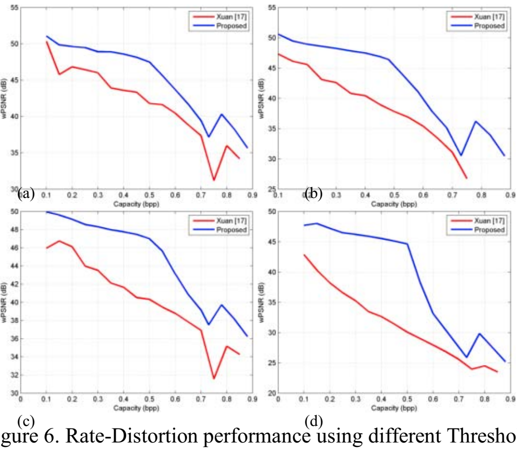

Figure 6. Rate-Distortion performance using different Threshold Selection methods for (a) *Aircraft*, (b) *Barbara*, (c) *Lena* and (d) *Mandrill* test images.

The images in Fig. 8 demonstrate the superiority of the proposed method in relation to other state of the art methods. It can be seen that the encryption [5] and scrambling [8] processes provide images which are not natural. Moreover, it can be seen that the reversible de-identification process presented in this work, which embeds the information using reversible watermarking, manages to keep the image natural. It is important to

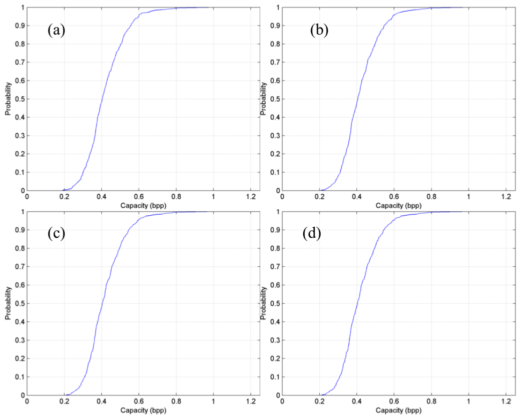

Figure 7. CDF of the capacity needed to embed ***s*** at different obfuscation levels (a) $k = 2$, (b) $k = 5$, (c) $k = 10$ and (d) $k = 20$.

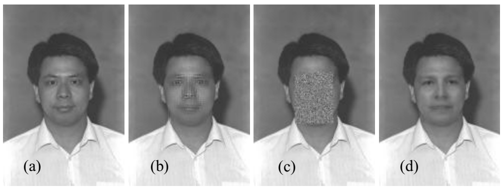

Figure 8. Comparing the resulting reversible de-identified images (a) Original Image, (b) Scrambling of DCT coefficients [8], (c) Encryption of pixel values [5] and (d) proposed method.

notice that the person de-identified using *k*-same is not recognizable from his facial features since eyes, nose and mouth features are modified and that the distortion introduced by the watermark is almost negligible. This work can be extended to other soft biometrics, such as hear. The system was also tested using erroneous encryption keys at the receiver. In all tests conducted the receiver failed to recover the original image when the wrong encryption key is used.

VI. Conlusion

This work presents a novel Reversible De-Identification method for lossless compressed images. The proposed scheme is generic and can be employed with other obfuscation strategies other than *k*-Same. A two-level Reversible-Watermarking scheme was adopted which uses Differential Evolution to find the optimal set of thresholds and provides a single-pass embedding capacity close to 1.25 bpp. Simulation results have shown that this method is able to recover the original image if the correct encryption key is employed. It further shows that 0.8 bpp were sufficient to cater for 99.8% of the frontal images considered and none of the image needed more than 1.1 bpp. Future work will focus on the extension of this algorithm for lossy image and video compression standards.

References

[1] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y. Li Tian and A. Ekin, "Blinkering Surveillance: Enabling video privacy through Computer Vision," *IBM Research Report*, vol. 22886, 2003.

[2] E.M. Newton, L. Sweeney and B. Malin, "Preserving privacy by de-identifying face images," *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, no. 2, pp. 232-243, Feb. 2005.

[3] W. Zhang, S.S. Cheung and M. Chen, "Hiding privacy information in video surveillance systems," in *IEEE Int. Conf. on Image Processing*, Genoa, Italy, Sep. 2005.

[4] I. Martinez-Ponte, X. Desumont, J. Meessen and J.F. Delaigle, "Robust Human Face Hiding ensuring Privacy," in *Proc. of Int. Workshop on Image Analysis for Multimedia Services*, Montreux, Switzerland, Apr. 2005.

[5] T.E. Boult, "Pico: Privacy throough invertible cryptographic obscuration," in *IEEE Proc. of the Computer Vision for Interactive Intelligent Environment*, Whashington DC, USA, Nov. 2005.

[6] K. Martin and K.N. Plataniotis, "Privacy protected surveillance using secure visual object coding," *IEEE Trans. Circuits and System for Video Technol.*, vol. 18, no. 8, pp. 1152-1162, Aug. 2008.

[7] F. Dufaux, M. Ouaret, Y. Abdeljaoued, A. Navarro, F. Bergnenegre and T. Ebrahimi, "Privacy Enabling Technology for Video Surveillance," in *SPIE Mobile Multimedia/Image Processing for Military and Security Applications*, Orlando, Florida, May 2006.

[8] F. Dufaux and T. Ebrahimi Scrambling for privacy protection in video surveillance systems, "Scrambling for privacy protection in video surveillance systems," in *IEEE Trans on Circuits and Systems for Video Technol.*, vol. 18, no. 8, pp. 1168-1178, Aug. 2008.

[9] H. Sohn, W. De Neve and Y-M. Ro, "Privacy protection in video surveillance systems: Analysis of subband-adaptive scrambling in JPEG XR," in *IEEE Trans. Circuits and Systems for Video Technol.*, vol. 21, no. 2, pp. 170-177, Feb. 2011.

[10] J. Meuel, M. Chaumont and W. Puech, "Data hiding in H.264 video for lossless reconstruction of region of interest," in *European Signal Processing Conf.*, Poznan, Poland, Sep. 2007.

[11] S.S. Cheung, J.K. Panichuri and T.P. Nguyen, "Managing privacy data in pervasive camera networks," in *IEEE Int. Conf. on Image Processing*, San Diego, California, USA, Oct. 2008.

[12] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Proc. Computer Vision and Pattern Recognition*,Kauai, USA, Dec. 2001.

[13] F. Hahmann, G. Boer and H. Schramm, "Combination of Facial Landmarks for Robust Eye Localization using the Discriminative Generalized Hough Transform," in *Int. Conf. of the Biometrics Special Interest Group*, Darmstadt, Germany, Sep. 2013.

[14] R. C. Gonzalez and R.E. Woods, *Digital Image Processing*, Second Edition, Prentice Hall, 2001.

[15] M. Weinberger, G. Seroussi and G. Sapiro, "LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm," in *Proc. IEEE Data Compression Conf.*, Washington DC., USA, Apr. 1996.

[16] P. Deutsch, *DEFLAGE Compressed Data Format Specification version 1.3*, RFC1951 (International), May 1996.

[17] G. Xuan, Y.Q. Shi, P. Chai, X. Cui, Z. Ni and X. Tong, "Optimum Histogram Pair based image Lossless Data Embedding," in *Proc. Int. Workshop on Digital Watermarking*, Berlin, Germany, 2008.

[18] Z. Wang, E.P. Simoncelli and A.C. Bovik, "Multi-Scale Structural Similarity for Image Quality Assessment, in *IEEE Proc.Asilomar Conf. on Signals, Systems and Computers*, Pacific Grove, CA, USA, Nov. 2003.

[19] R. Storn, K. Price, "Differential Evolution: A simple and efficient heuristic for global optimization over continuous spaces," *J. on Global Optimization*, vol.11, no. 4, pp. 341-359, Dec. 1997.

[20] F. De Simone, M. Ouaret, F. Dufaux, A.G. Tescher and T. Ebrahimi, "A Comparative study of JPEG2000, AVC/H.264 and HD Photo," in *Proc. SPIE Optics and Photonics, Applications of Digital Image*, San Diego, USA, Aug. 2007.

[21] D. Coltuc and J-M. Chassery, "Very Fast Watermarking by reversible Contrast Mapping," *IEEE Sig. Proc. Letters*, vol. 14, no. 4, Apr. 2007.

[22] T. Ahonen, A. Hadid and M. Pietikainen, "Face Description with Local Binary Patterns: Application to Face Recognition," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037-2041, Dec. 2006.