A NEW SECURE VIDEO TRANSMISSION TECHNIQUE VIA SECRET – FRAGMENT-VISIBLE MOSAIC VIDEOS BY NEARLY COLOR TRANSFORMATION TECHNIQUES

[1]K.DAYANA, [2]P.HAREESH

[1]M.tech Student (DSCE), [2]Assistant Professor
Department of ECE, Annamacharya Institute of technology and Sciencess, Tirupathi.
E-mail: [1]supraja.naveen14@gmail.com, [2]harishpasupulati@gmail.com

Abstract- A new secure Video transmission technique is proposed, In which Frames are collected from the Host and Target video's. Post which, frames are shuffled and color transformed to generate a secret video of the same size. The main goal of video encryption is keeping video secure from unauthorized attackers. The reverse of video encryption is video Decryption, which regenerates the original video. We have many Encryption algorithms with their own advantages and limitations. It is nearly impossible to overcome all the disadvantages but we had made an effort to minimize the limitations with our proposed algorithm. A scheme of handling the overflows/underflows in the converted pixels' color values by recording the color differences in the untransformed color space is also proposed. The information required for recovering the secret video is embedded into the created mosaic video by a lossless data hiding scheme using a key.
Good experimental results showcases feasibility of the proposed method.

Index Terms- Color Transformation, Data Hiding, Image Encryption, Mosaic Frames, Secure Video Transmission.

I. INTRODUCTION

Currently, images / videos from various sources are frequently utilized and transmitted through the internet for various applications, such as online personal photograph albums, confidential enterprise archives, document storage systems, medical imaging systems, and military image databases. These images usually contain private or confidential information so that they should be protected from leakages during transmissions. Recently, many methods have been proposed for securing multimedia data transmission, for which two common approaches are image encryption and data hiding. Video encryption is a technique that makes use of the natural property of an frame, such as high redundancy and strong spatial correlation, to get an encrypted frame based on Shannon's confusion and diffusion properties [1]–[7]. The encrypted frames is a noise frame so that no one can obtain the secret frame from it unless he/she has the correct key. However, the encrypted video is a meaningless file, which cannot provide additional information before decryption and may arouse an attacker's attention during transmission due to its randomness in form. An alternative to avoid this problem is data hiding [8]–[18] that hides a secret message into a cover frames so that no one can realize the existence of the secret data, in which the data type of the secret message investigated in this paper is an video. Existing data hiding methods mainly utilize the techniques of LSB substitution [8], histogram shifting [9], difference expansion [10]–[11], prediction-error expansion [12]–[13], recursive histogram modification [14], and discrete cosine/wavelet transformations [15]–[18]. However, in order to reduce the distortion of the resulting frames, an upper bound for the distortion value is usually set on the payload of the cover frame. A discussion on this rate distortion issue can be found in [19]. Thus, a main issue of the methods for hiding data in video frames is the difficulty to embed a large amount of message data into a single video. Specifically, if one wants to hide a secret frame into a cover frame with the same size, the secret video frames must be highly compressed in advance. For example, for a data hiding method with an embedding rate of 0.5 bits per pixel, a secret video frames with 8 bits per pixel must be compressed at a rate of at least 93.75% beforehand in order to be hidden into a cover video frame. But, for many applications, such as keeping or transmitting medical videos, military videos, legal documents, etc., that are valuable with no allowance of serious distortions, such data compression operations are usually impractical. Moreover, most frame compression methods, such as JPEG compression, are not suitable for line drawings and textual graphics, in which sharp contrasts between adjacent pixels are often destructed to become noticeable artifacts [20]. In this paper, a new technique for secure video transmission is proposed, which transforms a secured target video into a meaningful color transformed secret video with the same size and looking like a preselected target video. The transformation process is controlled by a secret key, and only with the key can a person recover the target video nearly losslessly from the secret video. The proposed method is inspired by Lai and Tsai [21], in which a new type of computer art image, called secret-fragment-visible mosaic frame, was proposed. The secret video is the result of rearrangement of the frames of a host video in disguise of another frames called the target video preselected from a database. But as a extension and to overcome the weakness of Lai and Tsai [21] is the requirement of a large image database so that the generated secret video can be sufficiently similar to the selected target video frames. Using their method, the user is not allowed to select freely his/her favorite frame for use as the target frame. It is therefore desired in this study to remove this weakness of the method while keeping its merit, that is, it is aimed to design a new method that can transform a host frames into a secret fragment- visible mosaic video of the same size that has the visual appearance of any freely selected target stitched video frames without the need of a database.

(a) (b) (c)

Fig. 1. Result yielded by the proposed method. (a) Secret Frame. (b) Target Frame. (c) Secret-fragment-visible mosaic frame created from (a) and (b) by the proposed method.

As an illustration, Fig. 1 shows a result yielded by the proposed method. Specifically, after a target frame is selected arbitrarily, the given secret frame is first divided into rectangular fragments called tile frames, which then are fit into similar blocks in the target frames, called target blocks, according to a similarity criterion based on color variations. Next, the color characteristic of each tile frame is transformed to be that of the corresponding target block in the target frames, resulting in a secret video which looks like the target frames. Relevant schemes are also proposed to conduct nearly lossless recovery of the original host video frames from the resulting secret video frames. The proposed method is new in that a meaningful secret video frames is created, in contrast with the image encryption method that only creates meaningless noise frames. Also, the proposed method can transform a host video frames into a disguising secret video frames without compression, while a data hiding method must hide a highly compressed version of the secret frames into a cover frames when the secret frames and the cover frames have the same data volume. In the remainder of this paper, the idea of the proposed method is described in Sections II and III. Detailed algorithms for secret video creation and host video recovery are given in Section IV. In Section V, results are presented to show the feasibility of the proposed method, and in Section VI, the security issue of the proposed method is discussed, followed by conclusions in Section VII.

II. IDEAS OF THE PROPOSED METHOD

The proposed method includes two main phases as shown by the flow diagram of Fig. 2: 1) Secret Video creation and 2) Host Video recovery.

In the first phase, secret video is yielded, which consists of the frames of an input host frame with color corrections according to a similarity criterion based on color variations.

The phase includes four stages: 1) fitting the host frames of the host video into the target blocks of a preselected target frames from target video; 2) transforming the color characteristic of each host frame in the secret frame to become that of the corresponding target block in the target frame; 3) rotating selected host frame into a direction with the minimum RMSE value with respect to its corresponding target frame; and 4) embedding relevant information into the secret video for future recovery of the host and target Videos. In the second phase, the embedded information is extracted to recover nearly losslessly the secret frame from the generated secret video. The phase includes two stages: 1) extracting the embedded information for secret frames recovery from the secret video, and 2) recovering the host and target videos using the extracted information.

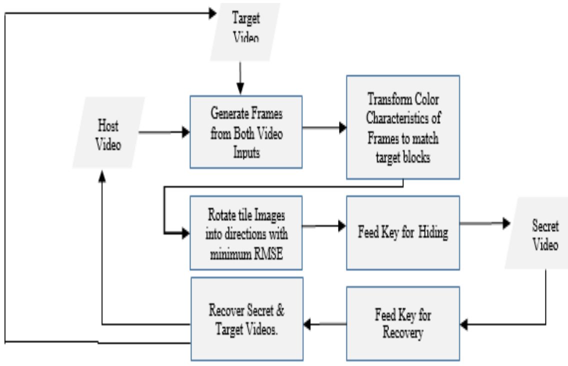

Fig. 2. Flow diagram of the proposed method.

III. IDEAS OF SECRET VIDEO GENERATION

Problems encountered in generating secret video frames are discussed in this section with solutions to them proposed.

A. Color Transformations between Blocks

In the first phase of the proposed method, each host frame T in the given host video is fit into a target block B in a preselected target frame. Since the color characteristics of T and B are different from each other, how to change their color distributions to make them look alike is the main issue here. Reinhard *et al.* [22] proposed a color transfer scheme in this aspect, which converts the color characteristic of an frame to be that of another in the l$\alpha\beta$ color space. This idea is an answer to the issue and is adopted in this paper, except that the RGB color space instead of the l$\alpha\beta$ one is used to reduce the volume of the required information for recovery of the original host video. More specifically, let *T* and *B* be described as two pixel sets *{p*1, *p*2. . . *pn}* and *{p'*1, *p'*2, . . . , *p'n }*, respectively. Let the color of each *pi* be denoted by (*ri*, *gi*, *bi*) and that of each *p'i* by (*r' i*, *g'i*, *b'i*). At first, we compute the means and standard

$$\mu c = \frac{1}{n}\sum_{i=1}^{n} ci \qquad \mu c' = \frac{1}{n}\sum_{i=1}^{n} ci' \qquad \ldots\ldots\ldots\ldots\ldots 1$$

$$\sigma c = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ci - \mu c)2},$$

$$\sigma c' = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ci' - \mu c')2} \qquad \ldots\ldots\ldots\ldots\ldots 2$$

in which *ci* and *c' i* denote the C-channel values of pixels *pi* and *p' i*, respectively, with *c* = *r*, *g*, or *b* and C=R, G, or B. Next, we compute new color values (*r" i* , *g" i* , *b" i*) for each *pi* in *T* by

$$ci" = qc(ci - \mu c) + \mu c' \qquad \ldots\ldots\ldots\ldots\ldots 3$$

in which *qc* = *σ' c*/*σc* is the standard deviation quotient and *c* = *r*, *g*, or *b*. It can be verified easily that the ew color mean and variance of the resulting tile frame *T'* are equal to those of *B*, respectively. To compute the original color values (*ri*, *gi*, *bi*) of *pi* from the new ones (*r" i* , *g" i* , *b" i*), we use the following formula which is the inverse of (3):

$$ci = \frac{1}{qc}(ci" - \mu c') + \mu c \qquad \ldots\ldots\ldots\ldots\ldots 4$$

Furthermore, we have to embed into the created mosaic frame sufficient information about the new tile frame *T_* for use in the later stage of recovering the original secret frame. For this, theoretically we can use (4) to compute the original pixel value of *pi*. However, the involved mean and standard deviation values in the formula are all real numbers, and it is impractical to embed real numbers, each with many digits, in the generated mosaic frame. Therefore, we limit the numbers of bits used to represent relevant parameter values in (3) and (4). Specifically, for each color channel we allow each of the means of *T* and *B* to have 8 bits with its value in the range of 0 to 255, and the standard deviation quotient *qc* in (3) to have 7 bits with its value in the range of 0.1 to 12.8. That is, each mean is changed to be the closest value in the range of 0 to 255, and each *qc* is changed to be the closest value in the range of 0.1 to 12.8. We do not allow *qc* to be 0 because otherwise the original pixel value cannot be recovered back by (4) for the reason that 1/*qc* in (4) is not defined when *qc* =0.

B. Choosing Appropriate Target Blocks and Rotating Blocks to Fit Better with Smaller RMSE Value

In transforming the color characteristic of a tile frame *T* to be that of a corresponding target block *B* as described above, how to choose an appropriate *B* for each *T* is an issue. For this, we use the standard deviation of the colors in the block as a measure to select the most similar *B* for each *T*. Specially, we sort all the host frames to form a sequence, *Shost*, and all the target blocks to form another, *Starget*, according to the average values of the standard deviations of the three color channels. Then, we fit the first in *Shost* into the first in *Starget*, fit the second in *Shost* into the second in *Starget* , and so on. Additionally, after a target block *B* is chosen to fit a host frame *T* and after the color characteristic of *T* is transformed, we conduct a further improvement on the color similarity between the resulting host frame *T_* and the target block *B* by rotating *T_* into one of the four directions, 0o, 90o, 180o, and 270o, which yields a rotated version of *T_* with the minimum root mean square error (RMSE) value with respect to *B* among the four directions for final use to fit *T* into *B.*

C. Handling Overflows/Underflows in Color Transformation

After the color transformation process is conducted a described previously, some pixel values in the new host frame *T_* might have overflows or underflows. To deal with this problem, we convert such values to be non-overflow or non underflow ones and record the value differences as residuals for use in later recovery. Specifically, we convert all the transformed pixel values in *T_* not smaller than 255 to be 255, and all those not larger than 0 to be 0. Next, we compute the differences between the original pixel values and the converted ones as the residuals and record them as part of the information associated with *T_*. Accordingly, the pixel values, which are just on the bound of 255 or 0, however, cannot be distinguished from those with overflow/underflow values during later recovery since all the pixel values with overflows/underflows are converted to be 255 or 0 now. To remedy this, we define the residuals of those pixel values which are on the bound to be 0 and record them as well. However, as can be seen from (3), the ranges of possible residual values are unknown, and this causes a problem of deciding how many bits should be used to record a residual. To solve this problem, we record the residual values in the untransformed color space rather than in the transformed one. That is, by using the following two formulas, we compute first the smallest possible color value *cS* (with *c* = *r*, *g*, or *b*) in *T* that becomes larger than 255, as well as the largest possible value *cL* in *T* that becomes smaller than 0, respectively, after the color transformation process has been conducted:

$$C_S = \left[\left(\frac{1}{q_c}\right)(255 - \mu' c) + \mu c\right];$$

$$C_L = \left[\left(\frac{1}{q_c}\right)(0 - \mu' c) + \mu c\right]; \quad \ldots\ldots\ldots\ldots\ldots 5$$

Next, for an untransformed value *ci* which yields an overflow after the color transformation, we compute its residual as |*ci* - *cS*|; and for *ci* which yields an underflow, we compute its residual as |*cL* - *ci*|. Then, the possible values of the residuals of *ci* will all lie in the range of 0 to 255 as can be verified. Consequently, we can simply record each of them with 8 bits. And finally, because the residual values are centralized around zero, we use further in this study the Huffman encoding scheme to encode the residuals in order to reduce the number of required bits to represent them.

D. Embedding Information for Secret Video Recovery

In order to recover the host frames from the secret video, we have to embed relevant recovery information into the secret video. For this, we adopt a technique proposed by Coltuc and Chassery [24] and apply it to the least significant bits of the pixels in the created secret video to conduct data embedding. Unlike the classical LSB replacement methods [8], [25], [26], which substitute LSBs with message bits directly, the reversible contrast mapping method [24] applies simple integer transformations to pairs of pixel values. Specifically, the method conducts forward and backward integer transformations as follows, respectively, in which (*x*, *y*) are a pair of pixel values and (*x'*, *y'*) are the transformed ones

$$x' = 2x - y,\ y' = 2y - x \quad \ldots\ldots 6$$

$$x = \left[\frac{2}{3}x' + \frac{1}{3}y'\right],\ y = \left[\frac{1}{3}x' + \frac{2}{3}y'\right] \quad \ldots\ldots 7$$

The method yields high data embedding capacities close to the highest bit rates and has the lowest complexity reported so far. The information required to recover a host frame *T* which is mapped to a target block *B* includes: 1) the index of *B*; 2) the optimal rotation angle of *T*; 3) the truncated means of *T* and *B* and the standard deviation quotients, of all color channels; and 4) the overflow/underflow residuals. These data items for recovering a host frame *T* are integrated as a five-component bit stream of the form *M* = *t*1*t*2. . .*tmr*1*r*2*m*1*m*2. . .*m*48*q*1*q*2. . .*q*21*d*1*d*2. . .*dk* in which the bit segments *t*1*t*2 . . . *tm*, *r*1*r*2, *m*1*m*2 . . . *m*48, *q*1*q*2 . . . *q*21, and *d*1*d*2 . . . *dk* represent the values of the index of *B*, the rotation angle of *T*, the means of *T* and *B*, the standard deviation quotients, and the residuals, respectively. In more detail, the numbers of required bits for the five data items in *M* are discussed below: 1) the index of *B* needs *m* bits to represent, with *m* computed by

$$m = [\log[(WS \times HS)/NT]]$$

in which *WS* and *HS* are respectively the width and height of the secret frame *S*, and *NT* is the size of the target frame *T*; 2) it needs two bits to represent the rotation angle of *T* because there are four possible rotation directions; 3) 48 bits are required to represent the means of *T* and *B* because we use eight bits to represent a mean value in each color channel; 4) it needs 21 bits to represent the quotients of *T* over *B* in the three color channels with each channel requiring 7 bits; and 5) the total number *k* of required bits for representing all the residuals depends on the number of overflows or underflows in *T*_. Then, the above-defined bit streams of all the host frames are concatenated in order further into a total bit stream *Mt* for the entire secret frame. Moreover, in order to protect *Mt* from being attacked, we encrypt it with a secret key to obtain an encrypted bit stream *M*'*t* , which is finally embedded into the pixel pairs in the mosaic frame using the method of Coltuc and Chassery [24] described above. It may require more than one iteration in the encoding process since the length of *M*' *t* may be larger than the number of pixel pairs available in an iteration. A plot of the statistics of the numbers of required bits for secret frame recovery is shown in Fig. 8(b). Moreover, we have to embed as well some related information about the mosaic frame generation process into the mosaic frame for use in the secret frame recovery process. Such information, described as a bit stream *I* like *M* mentioned previously, includes the following data items: 1) the number of iterations conducted in the process for embedding the bit stream *M*' *t* ; 2) the total number of used pixel pairs in the last iteration for embedding *M*' *t* ; and 3) the Huffman table for encoding the residuals. With the bit stream *M*'*t* embedded into the secret video, we can recover the host video back as will be described later. Itis noted that some loss will be incurred in the recovered secret frame, or more specifically, in the color transformation process using (3), where each pixel's color value *ci* is multiplied by the standard deviation quotient *qc*, and the resulting real value *c*" *i* is truncated to be an integer in the range of 0 through 255. However, because each truncated part is smaller than the value of 1, the recovered value of *ci* using (4) is still precise enough to yield a color nearly identical to its original one. Even when overflows/underflows occur at some pixels in the color transformation process, we record their residual values as described previously and after using (4) to recover the pixel value *ci*, we add the residual values back to the computed pixel values *ci* to get the original pixel data, yielding a nearly losslessly recovered host video. According to the results of the experiments conducted in this paper, each recovered secret frame has a very small RMSE value with respect to the original host video, as will be shown later in Section V.

IV. ALGORITHMS OF THE PROPOSED METHOD

Based on the above discussions, the detailed algorithms for mosaic frame creation and secret frame recovery may now be described respectively as Algorithms 1 and 2.

Algorithm 1 Secret Video Creation

Input: a Host frame S, a target frame T, and a secret key K.
Output: a secret-fragmented Video with Mosaic Frames F.
Steps:
Stage 1. *Fitting the Host frames into the target blocks.*
Step 1. If the size of the target frame T is different from that of the host image S, change the size of T to be identical to that of S; and divide the host image S into n tile images $\{T1, T2, \ldots, Tn\}$ as well as the target image T into n target blocks $\{B1, B2, \ldots, Bn\}$ with each Ti or Bi being of size NT .
Step 2. Compute the means and the standard deviations of each host image Ti and each target block Bj for the three color channels according to (1) and (2); and compute accordingly the average standard deviations for Ti and Bj , respectively, for i = 1 through n and j = 1 through n.
Step 3. Sort the host images in the set $Shost = \{T1, T2, \ldots, Tn\}$ and the target blocks in the set $Starget = \{B1, B2, \ldots, Bn\}$ according to the computed average standard deviation values of the blocks; map in order the blocks in the sorted $Shost$ to those in the sorted $Starget$ in a 1-to-1 manner; and reorder the mappings according to the indices of the host images, resulting in a *mapping sequence* L of the form: $T1 \rightarrow Bj1$, $T2 \rightarrow Bj2$, . . . , $Tn \rightarrow Bjn$.
Step 4. Create a Secret Video F by fitting the host images into the corresponding target blocks according to L.

Stage 2. P*erforming color conversions between the Host Frames and the target blocks.*
Step 5. Create a *counting table TB* with 256 entries, each with an index corresponding to a residual value, and assign an initial value of zero to each entry (note that each residual value will be in the range of 0 to 255).
Step 6. For each mapping $Ti \rightarrow Bji$ in sequence L, represent the means μc and $\mu' c$ of Ti and Bji, respectively, by eight bits; and represent the standard deviation quotient qc appearing in (3) by seven bits, according to the scheme described in Section III(A) where $c = r, g,$ or b.
Step 7. For each pixel pi in each host image Ti of secret video F with color value ci where $c = r, g,$ or b, transform ci into a new value $c''i$ by (3); if $c''i$ is not smaller than 255 or if it is not larger than 0, then change $c''i$ to be 255 or 0, respectively; compute a residual value Ri for pixel pi by the way described in Section III(C); and increment by 1 the count in the entry in the counting table TB whose index is identical to Ri.
Stage 3. R*otating the host frames.*
Step 8. Compute the RMSE values of each color transformed host image Ti in F with respect to its corresponding target block Bji after rotating Ti into each of the directions θ =0o, 90o, 180o and 270o; and rotate Ti into the *optimal* direction θo with the smallest RMSE value.
Stage 4. E*mbedding the secret Video recovery information.*
Step 9. Construct a Huffman table HT using the content of the counting table TB to encode all the residual values computed previously.
Step 10. For each host frame Ti in mosaic image F, construct a bit stream Mi for recovering Ti in the way as described in Section III(D), including the bit-segment which encode the data items of: 1) the index of the corresponding target block Bji; 2) the optimal rotation angle $\theta°$ of Ti; 3) the means of Ti and Bji and the related standard deviation quotients of all three color channels; and 4) the bit sequence for overflows/underflows with residuals in Ti encoded by the Huffman table HT constructed in Step 9.
Step 11. Concatenate the bit streams Mi of all Ti in F in a raster-scan order to form a total bit stream Mt ; use the secret key K to encrypt Mt into another bit stream $M't$; and embed $M't$ into F by the reversible contrast mapping scheme proposed in [24].
Step 12. Construct a bit stream I including: 1) the number of conducted iterations Ni for embedding Mt; 2) the number of pixel pairs $Npair$ used in the last iteration; and 3) the Huffman table HT constructed for the residuals; and embed the bit stream I into mosaic image F by the same scheme used in Step 11.

Algorithm 2 Secret Video recovery

Input: a Secret Video F with n Host Frames $\{T1, T2, \ldots, Tn\}$ and the secret key K.
Output: the secret image S.
Steps:
Stage 1. *Extracting the Host & Target Frames with recovery Mechanism – By Using Secret Key K as Input.*
Step 1. Extract from F the bit stream I by a reverse version of the scheme proposed in [24] and decode them to obtain the following data items: 1) the number of iterations Ni for embedding $M't$; 2) the total number of used pixel pairs $Npair$ in the last iteration; and 3) the Huffman table HT for encoding the values of the residuals of the overflows or underflows.

Step 2. Extract the bit stream *M' t* using the values of *Ni* and *Npair* by the same scheme used in the last step.

Step 3. Decrypt the bit stream *M' t* into *Mt* by *K*.
Step 4. Decompose *Mt* into *n* bit streams *M*1 through *Mn* for the *n to-be-constructed* tile images *T*1 through *Tn* in *S*, respectively.

Step 5. Decode *Mi* for each tile image *Ti* to obtain the following data items: 1) the index *ji* of the block *Bji* in *F* corresponding to *Ti*; 2) the optimal rotation angle *θ*° of *Ti*; 3) the means of *Ti* and *Bji* and the related standard deviation quotients of all color channels; and 4) the overflow/underflow residual values in *Ti* decoded by the Huffman table *HT*.

Stage 2. *Recovering the Host & Target Video Files.*
Step 6. Recover one by one in a raster-scan order the tile images *Ti*, *i* = 1 through *n*, of the desired secret image *S* by the following steps: 1) rotate in the reverse direction the block indexed by *ji*, namely *Bji*, in *F* through the optimal angle *θ*° and fit the resulting block content into *Ti* to form an *initial* tile image *Ti*; 2) use the extracted means and related standard deviation quotients to recover the original pixel values in *Ti* according to (4); 3) use the extracted means, standard deviation quotients, and (5) to compute the two parameters *cS* and *cL*; 4) scan *Ti* to find out pixels with values 255 or 0 which indicate that overflows or underflows, respectively, have occurred there; 5) add respectively the values *cS* or *cL* to the corresponding residual values of the found pixels; and 6) take the results as the final pixel values, resulting in a *final* Host Si and Target Frames *Ti*. Step 7. Stitch all the final host frames in to host video and all the target frames in to Target Video.

V. RESULTS

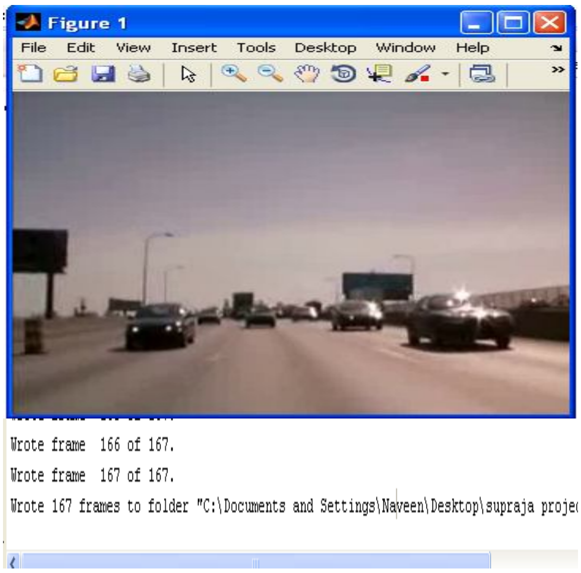

Fig 5.1 – Frames Collection from Host Video

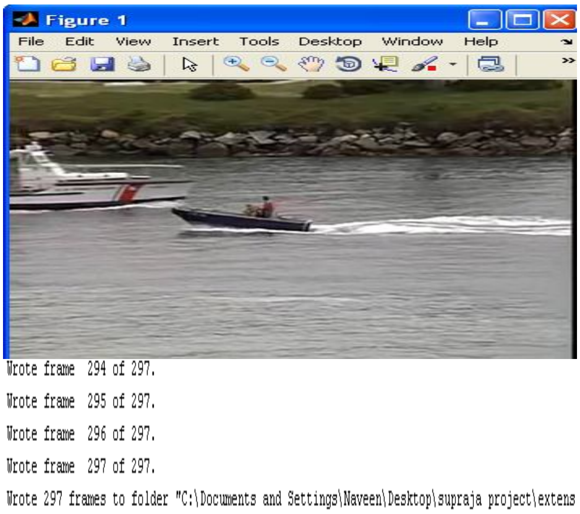

Fig 5.2 – Frames Collection from Target Video

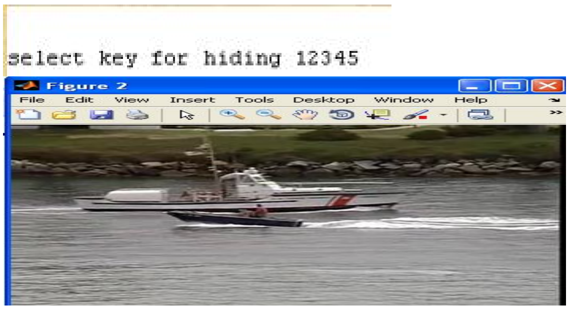

Fig 5.3 – Data Hiding by Encrypting with Secret Key – Secret Video Creation

Fig 5.4 – Decrypting using the Secret Key.

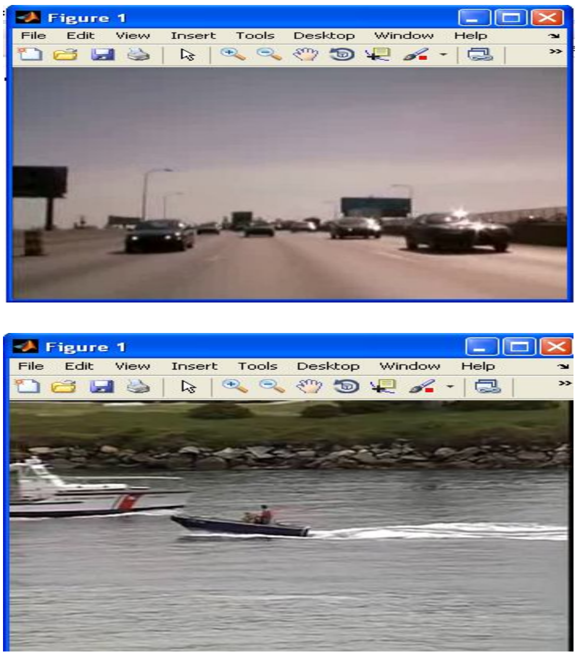

Fig 5.5 – Retreiving Host & Target Video

CONCLUSION

A new secure video transmission method has been proposed, which not only can create meaningful secret videos but also can transform a host frames into a visible mosaic frames with the same data size for use as a camouflage of the host frame. By the use of proper pixel color transformations as well as a skillful scheme for handling overflows and underflows in the converted values of the pixels' colors, secret-fragment visible mosaic frames with very high visual similarities to arbitrarily-selected target frames can be created with no need of a target image database. Also, the original host and target videos can be recovered nearly losslessly from the created secret videos. Good experimental results have shown the feasibility of the proposed method. Future studies may be directed to applying the proposed method to images of color models other than the RGB.

REFERENCESS

[1] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *Int. J. Bifurcat. Chaos*, vol. 8, no. 6, pp. 1259–1284, 1998.

[2] G. Chen,Y. Mao, and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps," *Chaos Solit. Fract.*, vol. 21, no. 3, pp. 749–761, 2004.

[3] L. H. Zhang, X. F. Liao, and X. B. Wang, "An image encryption approach based on chaotic maps," *Chaos Solit. Fract.*, vol. 24, no. 3, pp. 759–765, 2005.

[4] H. S. Kwok and W. K. S. Tang, "A fast image encryption system based on chaotic maps with finite precision representation," *Chaos Solit. Fract.*, vol. 32, no. 4, pp. 1518–1529, 2007.

[5] S. Behnia, A. Akhshani, H. Mahmodi, and A. Akhavan, "A novel algorithm for image encryption based on mixture of chaotic maps," *Chaos Solit. Fract.*, vol. 35, no. 2, pp. 408–419, 2008.

[6] D. Xiao, X. Liao, and P. Wei, "Analysis and improvement of a chaosbased image encryption algorithm," *Chaos Solit. Fract.*, vol. 40, no. 5, pp. 2191–2199, 2009.

[7] V. Patidar, N. K. Pareek, G. Purohit, and K. K. Sud, "A robust and secure chaotic standard map based pseudorandom permutationsubstitution scheme for image encryption," *Opt. Commun*., vol. 284, no. 19, pp. 4331–4339, 2011.

[8] C. K. Chan and L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*., vol. 37, pp. 469–474, Mar. 2004.

[9] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol*., vol. 16, no. 3, pp. 354–362, Mar 2006.

[10] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol*., vol. 13, no. 8, pp. 890–896, Aug. 2003.

★★★