REVERSIBLE TEXTURE SYNTHESIS FOR DATA SECUITY

A.Shunmuga Priya, Ms.V.Jyothirbindhu
Department of computer science National Engineering College Thoothukudi,India
Assistant Professor National Engineering college Thoothukudi,India

Abstract:

A texture synthesis process resamples a smaller texture image, which synthesizes a new texture image with a similar local appearance and an arbitrary size. We weave the texture synthesis process into steganography to conceal secret messages. Instead of using an existing cover image to hide messages, our techniques hide the source texture image and embeds secret messages through the process of textures synthesis. By using this approach we can extract the secret messages and source texture from a stego synthetic texture. Our techniques has three distinct advantages. First, our scheme offers the embedding capacity that is proportional to the size of the stego texture image. Second, a steganalytic algorithm is not likely to defeat our steganographic approach. Third, the reversible capability inherited from our scheme provides functionality, which allows recovery of the source texture.

Keyword:-**Steganography, texture synthesis, Encryption, Decryption,**

1.INTRODUCTION

Steganography is the method of hiding a message, file, image, or video within another file, message, image, or video. The word steganography combines from the two Greek words steganos means protected, and grapheins means writing. The advantage of steganography than cryptography is that the secret message does not attract the attention of the attackers by simple observation. The cryptography protects only the content of the message, while steganography protects the both messages and communication environment. Most stenographic methods take over an existing image as a cover medium. When embedding secret messages into this cover image, distortion of the image may occur. Because of this reason two drawbacks occur. First, since the size of the cover image is fixed, the more secret messages which are embedded leads to more image distortion. Therefore to maintain image quality it will provide limited embedding capacity to any specific cover image. Second, that image steganalysis approach is used to detect hidden messages in the stego image.

The method uses a secret key for source texture recovery. Secret Key Steganography is defined as a steganographic system that requires the exchange of a secret key (stego-key) prior to communication. The source texture is embedded using the secret key. Only the parties who know the secret key can reverse the process and recover the source texture. Here a perceived invisible communication channel is present. This steganography method exchanges a stego-key, which makes it more susceptible to interception. Our approach offers three advantages. First, since the texture synthesis can synthesize an arbitrary size of texture images, the embedding capacity which the scheme offers is proportional to the size of the stego texture image. Secondly, a steganalytic algorithm is not likely to defeat this steganographic approach since the stego texture image is composed of a source texture rather than by modifying the existing image contents. Third, the reversible capability inherited from the scheme provides functionality to recover the source texture.

2.RELATED WORKS

Texture synthesis has received a lot of attention recently in computer vision and computer graphics [8]. The most recent work has focused on texture synthesis by example, in which a source texture image is re-sampled using either pixel-based or patch-based algorithms to produce a new synthesized texture image with similar local appearance and arbitrary size.

Pixel-based algorithms [9]–[11] generate the synthesized image pixel by pixel and use spatial neighborhood comparisons to choose the most similar pixel in a sample texture as the output pixel. Since each output pixel is determined by the already synthesized pixels, any wrongly synthesized pixels during the process influence the rest of the result causing propagation of errors.

Otori and Kuriyama [12], [13] pioneered the work of combining data coding with pixel-based texture synthesis.

Secret messages to be concealed are encoded into colored dotted patterns and they are directly painted on a blank image. A pixel-based algorithm coats the rest of the pixels using the pixel-based texture synthesis method, thus camouflaging the existence of dotted patterns. To extract messages the printout of the stego synthesized texture image is photographed before applying the data-detecting mechanism. The capacity provided by the method of Otori and Kuriyama depends on the number of the dotted patterns. However, their method had a small error rate of the message extraction.

Patch-based algorithms [14], [15] paste patches from a source texture instead of a pixel to synthesize textures. This approach

of Cohen *et al.* and Xu *et al.* improves the image quality of pixel-based synthetic textures because texture structures inside the patches are maintained. However, since patches are pasted with a small overlapped region during the synthetic process, one needs to make an effort to ensure that the patches agree with their neighbors.

Liang *et al.* [16] introduced the patch-based sampling strategy and used the feathering approach for the overlapped areas of adjacent patches.

Efros and Freeman [17] present a patch stitching approach called "image quilting." For every new patch to be synthesized and stitched, the algorithm first searches the source texture and chooses one candidate patch that satisfies the pre-defined error tolerance with respect to neighbors along the overlapped region. Next, a dynamic programming technique is adopted to disclose the minimum error path through the overlapped region. This declares an optimal boundary between the chosen candidate patch and the synthesized patch, producing visually plausible patch stitching.

Ni *et al.* [18] proposed an image reversible data hiding algorithm which can recover the cover image without any distortion from the stego image after the hidden data have been extracted. Histogram shifting is a preferred technique among existing approaches of reversible image data hiding because it can control the modification to pixels, thus limiting the embedding distortion, and it only requires a small size location map, thereby reducing the overhead encountered. The current state-of-the-art for reversible image data hiding is the general framework presented by Li *et al.* [19].

EXISTING SYSTEM

Using an existing cover image to hide messages, the algorithm conceals the source texture image and embeds secret messages through the process of texture synthesis. A typical steganographic application includes communications between two parties whose existence is unknown to a possible attacker and whose success depends on the existence of this communication. Many of the image steganographic algorithms uses an existing image as a cover medium. The expense of embedding secret messages into the cover image is the image distortion encountered in the stego image. There is no significant visual difference exists between the two stego synthetic textures and the pure synthetic texture.

3. PROPOSED SYSTEM

Proposed system uses an algorithm that can provide various numbers of embedding capacities, produces reasonable texture images and recover the source texture. The proposed system uses an image reversible data hiding algorithm which can recover the cover image without any distortion from stego image after the hidden data have been extracted. The proposed system uses the concept of patch is used. Patch is image block of source texture where its size is user specified. Patch size is denoted by its width and height. The proposed system uses the concept of kernel block, which formed by subdividing the source texture into a number of non-overlapping kernel block each of which has size of kernel width and kernel height. The shape of overlapped area in proposed algorithm varies because we have pasted source patches into workbench. The proposed algorithm selects "appropriate" patches by taking into consideration of secret messages

$$SPn = Sw/Kw \times Sh/Kh \quad (1)$$

❖ **Message Embedding Procedure**

In message embedding procedure, first divides the source texture image into image block, called patches. To produce an index table for recording the location of the corresponding source patch. Establish a blank image as workbench where its size is equal to the synthetic texture. Then paste the source patches into workbench by referring the source patch IDs stored in the index table to produce a composition image. Then find Mean square error of overlapped region between the synthesized area and the patch which want to place. Ranking these patches based on increasing order of Mean Square Error. Then select patches from list where its rank equals the decimal value of an n-bit secret message.

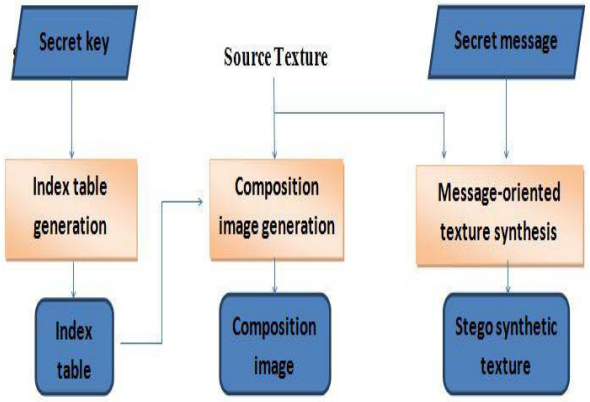

Fig. 1. The flowchart of the three-process message embedding procedure.

Message Extracting Procedure

The message extracting for the receiver side involves extracting the secret message concealed in the stego synthetic

texture, generating the index table, retrieving the source texture.

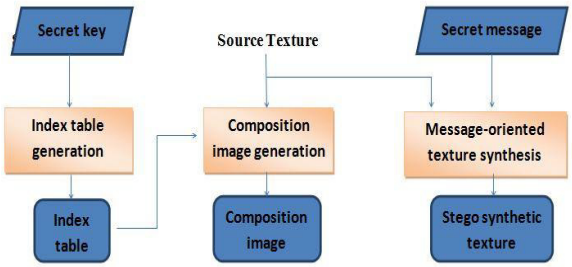

Fig. 2. The flowchart of the four-step message extracting procedure.

4.MODULES DESCRIPTION

Index Table Generation

The first process of this work is the index table generation where an index table is created to preserve the location of the source patch set SP inside the synthetic texture. The index table will allow us to access the synthetic texture and extract the source texture completely. The texture of any size according to user's wish can be generated using this index table. The dimension of the index table (Tpw × Tph) is first determined.]Given the parameters Tw and Th, which are the width and the height of the synthetic texture we intend to synthesize, the number of entries in this index table can be determined using equation (2) where TPn denotes the number of patches in the stego synthetic texture.

$$TPn=TPw\times TPh=[Tw-Pw/Pw-Pd+1]\times[Th-Ph/\ Ph-Pd+\ 1] \quad (2)$$

Inorder to achieve the manner of reversibility during the distribution of the source texture, we avoid positioning a source texture patch on the borders of the synthetic texture. The first-priority position L1 and the second-priority position L2, for two types of priority locations where ‖ L1 ‖ and ‖ L2 ‖, derived in (4), represent the number in the first -priority and second-priority positions, respectively.

$$\left.\begin{aligned} L1 &= Tpw-2/2\times Tph-2/2 \\ L2 &= Tpw-2/2\times Tph-2/2 \end{aligned}\right\} \quad (3)$$

Patch Based Composition

The second step is to attach the source patches into a workbench to create a composition image. First set up an empty image as the workbench where the size of the workbench is proportional to the synthetic texture. By referring to the source patch IDs stored in the index table, we then attach the source patches into the workbench. During the attaching process, if no imbrications of the source patches are found, we can attach the source patches directly into the workbench. However, if pasting locations cause the source patches to overlap each other, we employ the image quilting technique to reduce the visual artifact on the overlapped area.

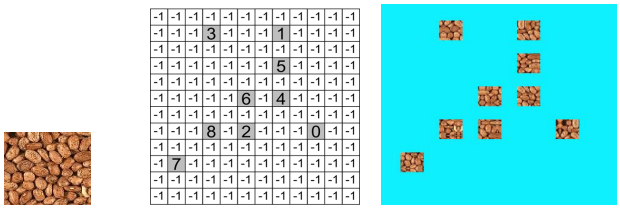

Fig. 4. An illustration of composition image; (a) the source texture (96 × 96),(b) the index table after patch distribution, (c) the composition image(488 × 488) by referring (a) and (b).

Message Embedding Module

The index table generation is carried out first where we produce an index table to record the location of source patch. The index table allows us to access the synthetic texture and retrieve the source texture completely. This reversible embedding style reveals one of the major benefits. The index table has the initial values of -1 for each entry, that shows the table is blank. We then assign values after distributing the source patch ID in the synthetic texture. In patch composition process we paste the source patches into the workbench to produce composite image. After generation of index table and composition image and after pasting source patches into workbench, we will embed the secret message via message oriented texture synthesis to produce stego synthetic texture

Capacity Determination

The embedding capacity our algorithm can offer is related to the capacity in bits that can be concealed at each patch (BPP, bit per patch), and to the number of embeddable patches in the stego synthetic texture (EPn). Each patch can conceal at least one bit of the secret message; thus, the lower bound of BPP will be 1, and the maximal capacity in bits that can be concealed at each patch is the upper bound of BPP, as denoted by BPPmax. In contrast, if we can select any rank from the candidate list, the upper bound of BPP will be log2(CPn) . The total capacity (TC) Of our algorithm is the multiplication of BPP and EPn. The number of the embeddable patches is the difference

between the number of patches in the synthetic texture (TPn) and the number of source patches subdivided in the source texture (SPn).

$$TC = \text{BPP} \times EP_n = \text{BPP} \times (TP_n - SP_n) \quad (4)$$

Source Texture Recovery, Message extraction, and Message Authentication

Generate the index table, given the secret key held in receiver side. The same index table as the embedding procedure can be generated. Recover the source texture. Source texture can be recovered or retrieved by referring the index table, then we arrange the blocks based on the order. Hence, the recovered texture will be same as the source texture. Generate the composite image, by pasting source patches into the workbench by referring the index table. Extract message by constructing candidate list and then perform match authentication step.

TABLE I

THE EMBEDDING CAPACITY PROVIDED BY OUR ALGORITHM

$BPP_{max} = \lfloor log_2[(S_w - P_w + 1) \times (S_h - P_h + 1)] \rfloor$
$SP_n = \frac{S_w}{P_w - (2 \times P_d)} \times \frac{S_h}{P_h - (2 \times P_d)}$
$TP_n = \left\lfloor \frac{(T_w - P_w)}{(P_w - P_d)} + 1 \right\rfloor \times \left\lfloor \frac{(T_h - P_h)}{(P_h - P_d)} + 1 \right\rfloor$
$EP_n = TP_n - SP_n$
$TC = \text{BPP} \times EP_n$

BPP_{max}: the maximal capacity in bits that can be concealed at each patch.
S_w: width of source texture, S_h: height of source texture.
P_w: width of a patch, P_h: height of a patch.
SP_n: the number of source patches subdivided in the source texture.
P_d: boundary depth of a patch.
TP_n: number of patches in the synthetic texture.
T_w: width of synthetic texture, T_h: height of synthetic texture.
EP_n: number of embeddable patches in the stego synthetic texture.
TC: total embedding capacity.

5.RESULTS

Results of Embedding Capacity On personal computer we collect our experimental results. For results we can take the four source texture. Following table shows the embedding capacity of our algorithm when various resolutions of synthetic texture are produced. For fixed number of BPP, the larger the resolution of source texture S_w x S_h (96x96 vs.192x192). Overall embedding capacity is TC. For 10 BPP our procedure will offer 6160 bits vs. 5890 bits. Because the large source texture contains more source patches S (9 vs. 36) and we need to paste these source patches which S_{pn} cannot hide any secret bits. It will decrease the number of embeddable patches E . Following graph presents the total E_{en} embedding capacity our algorithm. However, we can hire BPP (11 vs. 14) for conveying the further secret messages (6776bits vs. 8246 bits). The maximal capacity provided by our algorithm is 36456 bits provide.

TABLE II
TOTAL EMBEDDING CAPACITY IN BITS OUR ALGORITHM CAN PROVIDE

Synthetic texture size: $T_w \times T_h = 1008 \times 1008$;
Patch size: $P_w \times P_h = 48 \times 48$; Boundary depth: $P_d = 8$

$S_w \times S_h$	SP_n	EP_n	TC (5BPP)	TC (10 BPP)	TC (BPP_{max})
96×96	9	616	3080	6160	6776
128×128	16	609	3045	6090	7308
192×192	36	589	2945	5890	8246

CONCLUSION

This paper proposes the Reversible texture synthesis for data security. The source texture is given and the scheme produces the stego synthetic texture hiding the secret message. The patch based concept is used instead of pixel based approach. Provides the reversible approach to recover source texture and secret message from stego synthetic texture. In this given the source texture and then kernel blocks and source texture is generated. Index table is generated which stores the starting location of each patch. After it index table is generated, composite image is generated and then secret message is embedded into it. Reverse procedure is carried out to recover source texture and secret

REFERENCES

[1]. Kuo-Chen Wu and Chung-Ming Wang, "Steganography Using Reversible Texture Synthesis", IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 24, NO. 1,(2015).
[2]. N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer*, vol. 31, no. 2, pp. 26–34, 1998.
[3]. F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding survey," Proc. IEEE, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.
[4] Y.-M. Cheng and C.-M.Wang, "A high-capacity steganographic approach for 3D polygonal meshes," The Visual Computer, vol. 22, no. 9, pp. 845-855, 2006.
[5] S.-C. Liu and W.-H. Tsai, "Line-based cubism-like image |A new type of art image and its application to lossless data hiding," IEEE Trans. Inf. Forensics Security, vol. 7, no. 5, pp. 1448-1458, 2012.
[6] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," IEEE Trans. Image Process., vol. 23, no. 4, pp. 1779-1790, 2014.

[7] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color, and gray-scale images," MultiMedia, IEEE, vol. 8, no. 4, pp. 22-28, 2001.
[8] Y. Guo, G. Zhao, Z. Zhou, and M. Pietik•ainen, "Video texture synthesis with multi-frame LBP-TOP and di_eomorphic growth model," IEEE Trans. Image Process., vol. 22, no. 10, pp. 3879-3891, 2013.
[9] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000, pp. 479-488.
[10] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in Proc. of the Seventh IEEE International Conference on Computer Vision, 1999, pp. 1033-1038.
[11] C. Han, E. Risser, R. Ramamoorthi, and E. Grinspun, "Multiscale texture synthesis," *ACM Trans. Graph.*, vol. 27, no. 3, 2008, Art. ID 51.
[12] H. Otori and S. Kuriyama, "Data-embeddable texture synthesis," in *Proc.8th Int. Symp. Smart Graph.*, Kyoto, Japan, 2007, pp. 146–157.
[13] H. Otori and S. Kuriyama, "Texture synthesis for mobile data communications,"*IEEE Comput. Graph. Appl.*, vol. 29, no. 6, pp. 74–81,Nov./Dec. 2009.
[14] M. F. Cohen, J. Shade, S. Hiller, and O. Deussen, "Wang tiles for image and texture generation," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 287–294,2003
.[15] K. Xu *et al.*, "Feature-aligned shape texturing," *ACM Trans. Graph.*,vol. 28, no. 5, 2009, Art. ID 108
.[16] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum, "Real-time texture synthesis by patch-based sampling," *ACM Trans. Graph.*, vol. 20, no. 3,pp. 127–150, 2001
[17] A. A.Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. 28th Annu. Conf. Comput. Graph. Interact. Techn.*,2001, pp. 341–346.
[18] 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.
19] X. Li, B. Li, B. Yang, and T. Zeng, "General framework to histogram shifting-based reversible data hiding," *IEEE Trans. Image Process.*,vol. 22, no. 6, pp. 2181–2191, Jun. 2013.
[20] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *Amer. Statist.*, vol. 42, no. 1, pp. 59–66, 1988.
[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.