ISSN:0975-9646

Image Steganography By Using Reversible Texture Synthesis

Aarthi.K
PG Scholar,M.E - II year
Department of Computer Science and Engineering
Jansons Institute of Technology
Coimbatore, Tamilnadu

Gokul.S
PG Scholar,M.E - II year
Department of Computer Science and Engineering
Jansons Institute of Technology
Coimbatore, Tamilnadu

Abstract - **Texture synthesis has a variety of applications in graphics, computer vision and image and video processing. This process resamples a texture image, which synthesizes a new texture image with a similar appearance and size. Existing steganography methods adopt an existing image as a cover medium. This cover medium leads to two drawbacks, first one is size of the cover image is fixed. Hence the stego image contains distortion, this leads to second drawback that a hidden message is being conveyed in a stego image. To overcome these shortcomings we have come up with a reversible texture synthesis which involved hiding the message within the image by using the process of texture synthesis. Hence we call it as an image steganography by using reversible texture synthesis.**

Keywords **- Texture synthesis, Steganography, Reversible, Texture, Cover medium**

I. Introduction

The communication technologies around us have grown at great pace in recent times. For exchanging the data/information we need the data security to protect the data from third party organizations. For that we can use either steganography or cryptography to protect the data. Steganography is better than cryptography because the intended secret message does not attract attend to itself.

Image steganography can be broadly classified into spatial domain, transform domain, spread spectrum and model based steganography. In spatial domain, secret message is embedded in pixel value directly whereas transform domain methods achieve embedding by first transforming the image from spatial to frequency domain using any one of the transforms such as discrete cosine transform (DCT), discrete wavelet transform (DWT), Hadamard transform, Dual tree DWT, double density dual tree DWT (DD DT DWT), ridgelet transform, curvelet transform etc. and then embedding is done in suitable transform coefficients.

Texture synthesis [11] is the process of algorithmically constructing a large digital image from a small digital sample image by taking advantage of its structural content. There are several methods like tilling, stochastic, pixel-based, patch-based texture synthesis are available.

The remainder of this paper is organized as follows: in section II we review the texture synthesis techniques. In section III, we detail the proposed algorithm including embedding and extracting procedures. In section IV experimental results and performance analysis followed by conclusions in the final section.

II. Related Work

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 appearance and size.

Alexei A. Efros and Thomas K. Leung [1] says that the texture synthesis process grows a new image outward from an initial seed, one pixel at a time. Li-Yi Wei and Marc Levoy [7] introduced the Fast Texture Synthesis using Tree-structured Vector Quantization. It is a very simple algorithm that can efficiently synthesize a wide variety of textures.

Otori and Kuriyama [5] pioneered the work of combining data coding with pixel-based texture synthesis. Liang et al. [6] introduced the patch-based sampling strategy and used the feathering approach for the overlapped areas of adjacent patches.

Li-Yi Wei Marc Levoy [8] introduced Search-based texture synthesis algorithms are sensitive to the order in which texture samples are generated; different synthesis orders yield different textures. A multiscale texture synthesis [2] approach in order to both colourize and upscale greyscale textures. A parallel controllable synthesis algorithm for creating infinite texture without the limited variety of tiles.

Anteneh Addis Anteneh [3] tells that, second round of synthesis is conducted on the first synthesized image. The motivation for doing this is to optimize the synthesized image so that any irregularities caused during the first run of optimization will be minimized during multiple runs as the synthesized texture becomes more refined.

Greg Turk [4] draws upon texture synthesis methods that use image pyramids, and use a mesh hierarchy to serve in place of such pyramids. A texture created this way fits the surface naturally and seamlessly. Sylvain Lefebvre and Hugues Hoppe [9] present a texture synthesis scheme based on neighborhood matching.

Vivek Kwatra [10] pioneered patch regions from a sample image or video are transformed and copied to the output and then stitched together along optimal seams to generate a new output.

In this paper, we are taking the advantage of patch-based methods to embed a secret message during the synthesizing procedure.

III. PROPOSED METHOD

Steganography refers to embedding information or secret message into media. This approach works which takes advantage of patch-based methods to embed a secret message during the synthesizing procedure. This allows the source texture to be recovered in a message extracting procedure, providing the functionality of reversibility.

This is a simple and secure high capacity steganographic algorithm for information hiding. Synthesize a cover-image texture which is increased in size according to user desire. The input texture is smaller in size which is gradually grows, if the size is original into double and the texture is increased in size and resulting high resolution image is obtained. The secret information is placed in the high resolution image patches based on the lookup table created. The created lookup table denotes the location of the information. The secret message is embedded in the images based on LSB replacement of the DCT coefficients of the cover-image.

The same process is reversed in the received side in order to obtain the secret message and the original source patch. Here before embedding the secret message, we securing by embedding it, using the DCT of the input image. So we can identify whether the secret information is hidden in which location since secret information hidden in the DCT co-efficient of the original patches. The image, secret information is well secure since it is located in high resolution image which is of the sizes and differ.

The input texture is obtained and in that texture the embedding of the process is employed. In source patch this procedure the secret information is placed in source patch using DCT. The input texture is decomposed using DCT and the secret information is placed in the coefficients of the images in the LSB of the original table. Finally, to the obtained images, the inverse cosine transform is applied and it is used for the next procedure and after hiding the secret message in the image and composition table is created i.e., index table is created. The obtained source code is placed in the image matrix with help of the index table; it acts as the key for image composition.

The performance of process is measured based on the MSE and other parameters. On the other hand, each byte of the secret information is first encrypted based on an exponential modular arithmetic which is then partitioned into two 4-bit words.

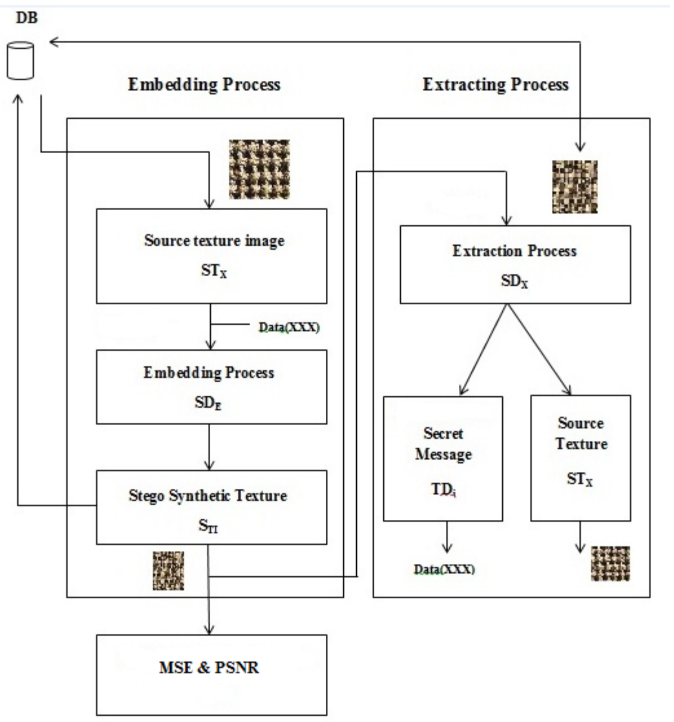


Fig.1 System Architecture

A texture synthesis re-samples a small texture image drawn by captured in a photograph in order to synthesize a new texture image with a similar local appearance and arbitrary size. Then weave the texture synthesis process into steganography concealing secret messages as well as the source texture. In contrast to using an existing cover image to hide messages, this algorithm conceals the source texture image and embeds secret messages through the process of texture synthesis. This allows us to extract the secret messages and the source texture from stego synthetic texture.

A. *Message Embedding Process*

The message embedding procedure follows the three process message embedding procedure. This process contains three sub steps. They are Index table generation process, Patch composition process and Message-oriented texture synthesis process.

Embedding Steps

Step 1 - Read the texture image
Step 2 - Generate Index table
Step 3 - Split the image into patches
Step 4 - Identify the location of Patches to hide the Secret data
Step 5 - Read the secret data
Step 6 - Convert image into binary format
Step 7 - Embed the secret data in the identified Location
Step 8 - Produce Stego image

Index table generation process is used to record the location of the source patch in synthetic texture. Patch composition process paste the source patches into a workbench to produce a composition image and then finally we embed a secret message via the message oriented texture synthesis to produce the final stego synthetic texture.

To distribute the patch we use SURF algorithm. For that first plot the surf location point and embedding process is done. Then we separate the blocks by using pixel strong points. After that secret data is pasted in to the blocks to get embedding map. If the length of the data is pasted in to the blocks to get embedding map then the text is embedded into image. Then the patch composition process and message oriented process is done.

B. *Message Extracting Process*

The message extracting for the receiver side involves generating the index table, retrieving the source texture, performing the texture synthesis, and extracting and authenticating the secret message concealed in the stego synthetic texture.

Extracting Steps

Step 1 - Stego image input

Step 2 - Identify the location of patch to hide the secret data

Step 3 - Extract the secret data

In the receiver side, the same index table as the embedding procedure can be generated. The next step is the source texture recovery. The recovered source texture which will be exactly the same as the source texture. In the next step, the composition image generation to paste the source patches into a workbench to produce a composition image by referring to the index table. This generates a composition image that is identical to the one produced in the embedding procedure. The final step is the message extraction. We can extract all of the secret messages that are concealed in the stego synthetic texture patch by patch.

IV. EXPERIMENT RESULTS

In the first step is to read the input image and then in that image we have to find out the surf location to embed the secret messages. After that we need to create embedding map to hide the data. The information is hidden based on the strong points.

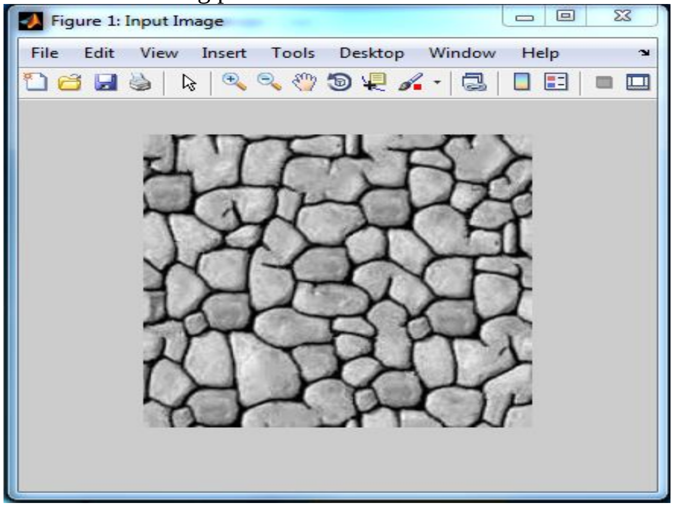


Fig.2 input image

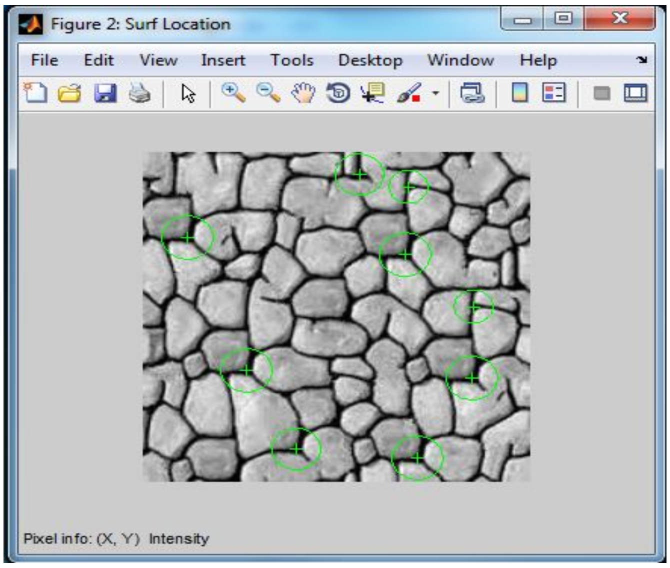
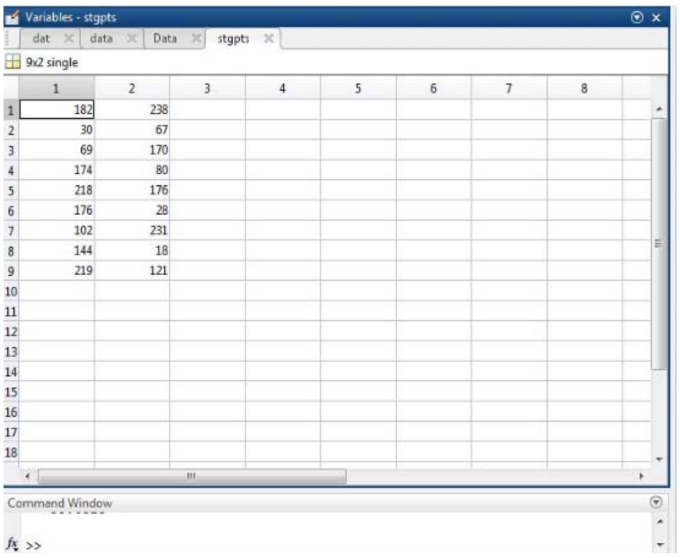


Figure 2: Surf Location
File Edit View Insert Tools Desktop Window Help
Pixel info: (X, Y) Intensity

Fig.3 Detection of strong points using SURF



	1	2
1	182	238
2	30	67
3	69	170
4	174	80
5	218	176
6	176	28
7	102	231
8	144	18
9	219	121

Fig.4 Strong points to hide secret data

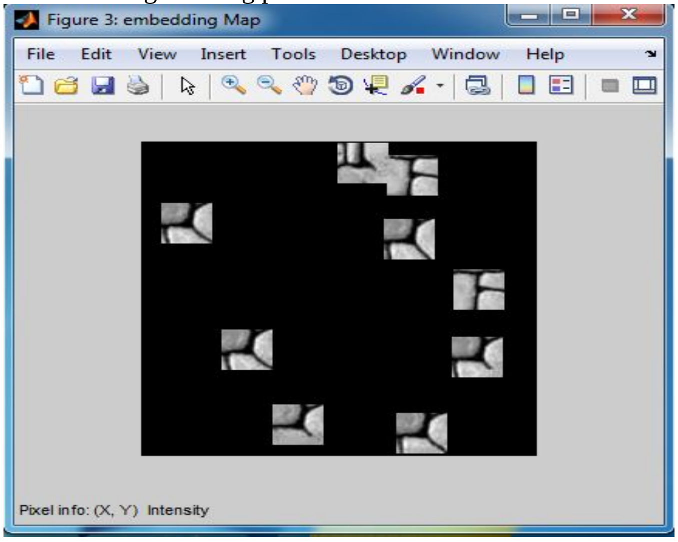


Figure 3: embedding Map
File Edit View Insert Tools Desktop Window Help
Pixel info: (X, Y) Intensity

Fig.5 Text Embedding Map

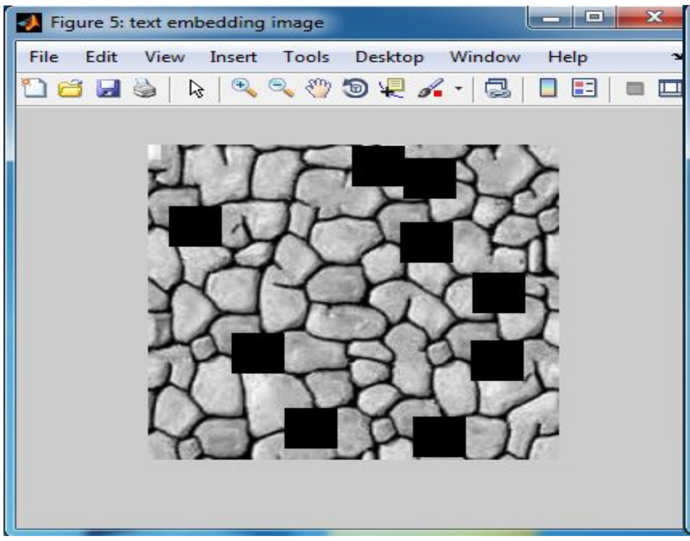


Figure 5: text embedding image
File Edit View Insert Tools Desktop Window Help

Fig.6 Text Embedding Image

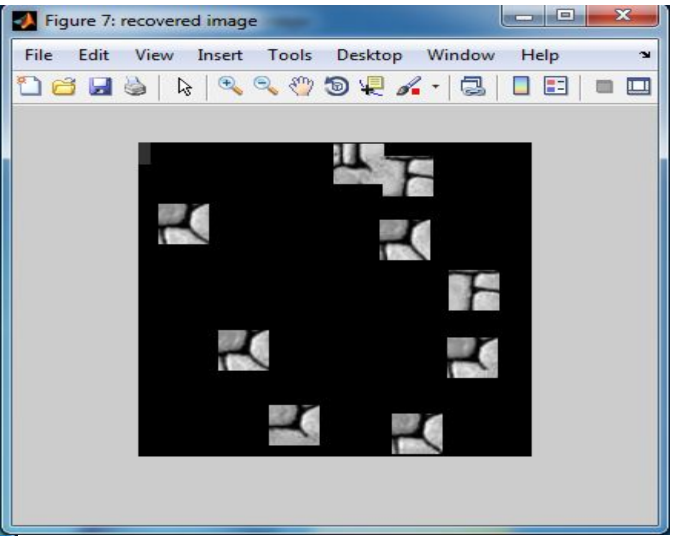


Figure 7: recovered image
File Edit View Insert Tools Desktop Window Help

Fig.9 Recovered image after extraction

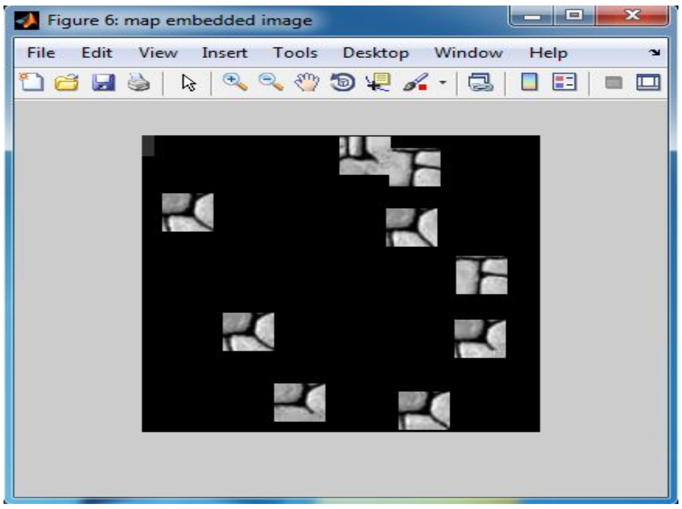
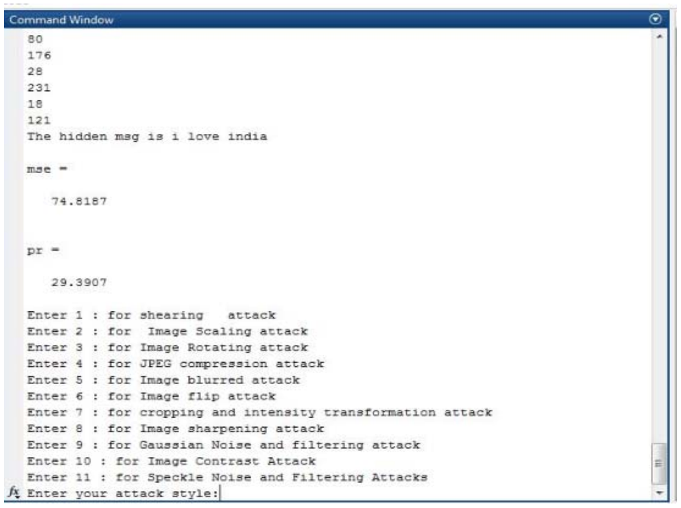


Figure 6: map embedded image
File Edit View Insert Tools Desktop Window Help

Fig.7 Text embedded image



```
80
176
28
231
18
121
The hidden msg is i love india

mse =

   74.8187


pr =

   29.3907

Enter 1 : for shearing   attack
Enter 2 : for  Image Scaling attack
Enter 3 : for Image Rotating attack
Enter 4 : for JPEG compression attack
Enter 5 : for Image blurred attack
Enter 6 : for Image flip attack
Enter 7 : for cropping and intensity transformation attack
Enter 8 : for Image sharpening attack
Enter 9 : for Gaussian Noise and filtering attack
Enter 10 : for Image Contrast Attack
Enter 11 : for Speckle Noise and Filtering Attacks
fx Enter your attack style:
```

Fig.10 Recovered secret message and MSE & PSNR Value

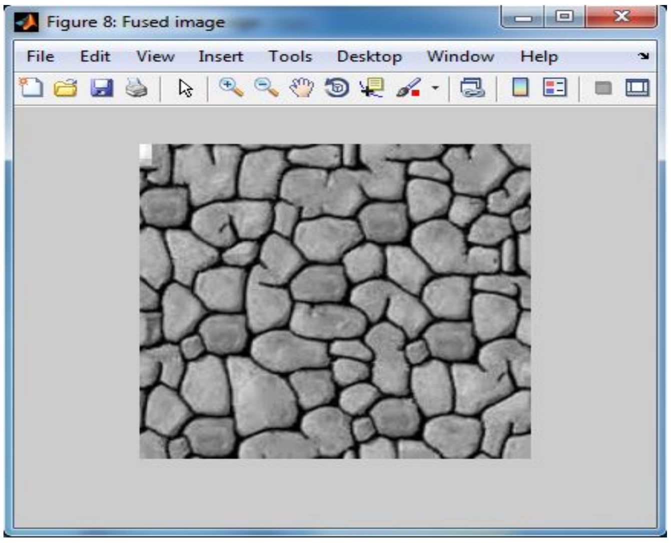


Fig.8 Fused Image

V. PERFORMANCE ANALYSIS

To check whether the method proposed in this paper is good, we need to calculate PSNR Value which is normally calculated to find the noise of the image. For calculating the PSNR value we have to find out the MSE value.

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \quad (1)$$

PSNR is,

$$PSNR = 10.\log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

$$= 20.\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \quad (2)$$

$$= 20.\log_{10}(MAX_I) - 10.\log_{10}(MSE)$$

Where, MAX_I is maximum possible pixel value of the image. For the experiment, the different color images have been used and in each image same amount of data is inserted and checked. It is shown in Table I.

Hence the data is successfully hidden inside the image without any noise. If anybody checks the fused image with an original image there is no visual discontinuities.

Table I. MSE and PSNR Values

Cover image	Size of the cover image	Data inserted	MSE	PSNR
	20.2 KB	I love India	38.9674	32.2238
	8.36 KB	I love India	82.8322	28.9488
	9.87 KB	I love India	74.8187	29.3907

VI. Conclusion And Future Work

A Novel reversible steganography using SURF algorithm in texture synthesis can produce a large stego synthetic texture concealing secret messages. The first that can exquisitely weave the steganography into a conventional patch-based texture synthesis. This algorithm may be secure and robust against an RS steganalysis attack. The proposed scheme offers substantial benefits and provides an opportunity to extend steganographic applications. This method uses to distribute the patches evenly in both the sender and receiver side and this novel method to determine the capacity in embedding process may be fast when compare to other traditional method.

One of possible future study is to expand this scheme to support other kinds of texture synthesis approaches to improve the image quality of the synthetic textures. Another possible study would be to combine other steganography approaches to increase the embedding capacities.

Acknowledgment

My sincere thanks to everyone who have contributed towards the preparation of the paper. All work done in this paper will surely help to the researchers for future work.

References

[1] Alexei A.Efros and Thomas K.Leung "*Texture Synthesis by Non-Parametric Sampling*", IEEE International Conference on Computer Vision, Corfu, Greece, September 1999.

[2] Andres Hast, Martin Ericsson and Stefan Seipel "*Multiscale Texture synthesis and colourization of greyscale textures*" WSCG 2010 Communication Papers.

[3] Anteneh Addis "*Texture synthesis using TSVQ and Target Re-synthesis*", CMSC 635 – Project.

[4] GregTurk "*Texture Synthesis on Surfaces*" GVU Center, College of Computing.

[5] Hirofunni otori and Shigeru Kuriyama "*Data-Embeddable Texture Synthesis*" Toyohashi University of technology.

[6] Lin Liang, Ce Liu,Ying-Qing Xu, Baining Guo and Heung-Yeung Shum" *Real-Time Texture synthesis by Patch-based Sampling*" ACM Transactions on Graphics, Vol. 20, No. 3, July 2001.

[7] Li-Yi Wei and Marc Levoy "*Fast Texture Synthesis using Tree-Structured Vector Quantization*" Stanford University.

[8] Li-Yi Wei Marc Levoy "*Order-Independent Texture Synthesis*" Stanford University.

[9] Sylvain Lefebvre and Hugues Hoppe "*Parallel Controllable Texture Synthesis*" Microsoft Research.

[10] Vivek Kwatra Arno Schodl Irfan Essa Greg Turk Aaron Bobick "*Graphcut Textures: Image and Video Synthesis Using Graph Cuts*" GVU Center / College of Computing.

[11] https://en.wikipedia.org/wiki/Texturesynthesis