Quantitative Evaluation of Near Regular Texture Synthesis Algorithms

Wen-Chieh Lin* James Hays** Chenyu Wu** Vivek Kwatra*** Yanxi Liu**
* College of Computer Science, National Chiao-Tung University
** School of Computer Science, Carnegie Mellon University
*** Department of Computer Science, University of North Carolina at Chapel Hill

Abstract

Near regular textures are pervasive in man-made and natural world. Their global regularity and local randomness pose new difficulties to the state of the art texture analysis and synthesis algorithms. We carry out a systematic comparison study on the performance of four texture synthesis algorithms on near-regular textures. Our results confirm that faithful near-regular texture synthesis remains a challenging problem for the state of the art general purpose texture synthesis algorithms. In addition, we provide comparison of human perception with computer evaluations on the quality of the texture synthesis results.

1. Introduction

Near regular textures are pervasive in both man-made and natural world. Even though textures are usually classified as either (structurally) regular or stochastic, most real-world textures fall somewhere in-between these two extremes. We view textures as a continuous spectrum where texture regularity varying gradually (Figure 1).

Regular textures are periodic patterns where the color/intensity and shape of all texture elements are repeating in equal intervals along two linearly independent directions. A *tile* is the smallest fundamental region [6] that can be used to synthesize a regular texture by tiling the 2D plane (e.g. wallpaper patterns). **Near-regular textures** can be viewed as statistical departures of regular textures along different dimensions [13]. Figure 1 shows a texture spectrum where the textures are arranged according to their regularity variation in geometry space. An ideal texture synthesis algorithm would be able to handle all types of textures on the texture spectrum. It is observed [14, 13] that the combination of regularity and randomness of near-regular textures challenges some state of the art texture synthesis algorithms. The goal of this paper is to evaluate the observations made in [14, 13] by comparing the performance of several texture synthesis algorithms in an objective and subjective (human evaluation), systematic and quantitative manner.

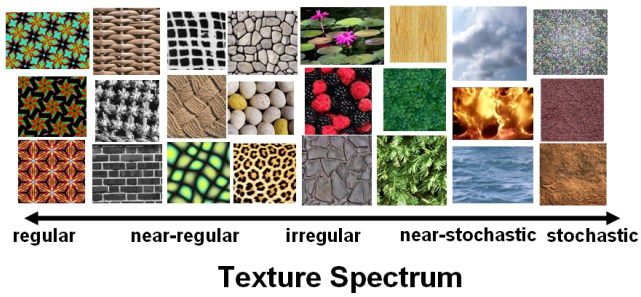


Figure 1. A texture spectrum arranged by texture regularity.

2. Selected texture synthesis algorithms

Work in texture synthesis has achieved impressive results in a variety of textures. These algorithms can be divided into two groups: statistical-model-based approaches [2, 3, 8, 16] and image-based approaches[1, 4, 5, 9, 10, 17, 18, 19]. Image based approaches refer to methods synthesize textures by directly copying image pixels or patches from the input texture and stitching them together in the synthesized image. They have the advantage of preserving image details by keeping the pixel neighborhood intact in the synthesized textures. On the other hand, these are local approaches in nature, with no special consideration given to the texture's global structures [14].

Fair and objective algorithm comparison is hard. Reimplementation of third-party algorithms has the risk of leaving out crucial details. To make the comparison results fair, systematic, and objective we contacted all the authors whose algorithms are compared in this paper and asked them to run their own algorithms or allow us to run their source code on the same set of input textures. Those algorithms whose authors could not comply with this request are not included. Some algorithms, e.g. [4], happen to have some published synthesis results from the same input textures (a subset of our total test set), this and only this subset is included for comparison. The four algorithms chosen below represent a spectrum of recent development in patch-based texture synthesis that made an impact on the field.

We compare four texture synthesis algorithms: (1) the

graph cut approach[9], (2) the near-regular texture synthesis approach[14], (3) the patch-based approach[10], and (4) the regularized patch-based approach described in [11], which is used as a controlled experiment (regularized patch-based vs. patch-based) to verify the difference of using the lattice structure of a near-regular texture or not in the same algorithm. We briefly describe each algorithm below.

2.1. Graph cut texture synthesis

Kwatra et al.[9] demonstrate a very effective general texture synthesis algorithm. Texture is synthesized by overlaying the entire input texture onto the synthetic texture at various offsets and using a graph cut algorithm to find the optimal region to add to the synthetic texture. The graph cut algorithm avoids the need for a fixed, a-priori patch size, and scales well to any dimension (such as video). However, for near regular textures the choice of offsets is as important as finding low-error seams. If the input texture is copied onto the synthesized texture at an offset that is inconsistent with the periodicity of the texture, any selection of seams will still violate the global regularity of the texture. Kwatra et al. describe patch placement algorithms which do a fair job of finding low error offsets. They treat the input texture as a template and compute the correlation between the template and the texture being synthesized to find the low error offsets. The maximum correlation offsets often, but not always, correspond to the offsets preserving the periodicity of the input texture.

2.2. Near-regular texture synthesis

Liu et al.[14] propose a texture synthesis algorithm for *geometrically-regular* near-regular textures or Type I near-regular texture [13]. The basic idea is to utilize the translational symmetry property[12] of a near-regular texture to find the underlying lattice structure of the texture patterns and locate the tiles. The lattice extraction process can be manually specified or fully automatic[7].

The extracted tiles represent the smallest parallelogram-shaped region that can reproduce the regular texture pattern under the texture's translation subgroup. For a regular texture, only one tile is needed to recover the full texture. For a near-regular texture, one needs a set of tiles collected by sampling the input texture in a principled manner to preserve both the geometric regularity and color/intensity variations in the input texture[14]. The tiles in the tile set have roughly the same size and shape but varied color/intensity. The output texture is synthesized by randomly picking a tile from the tile set and pasting the tile on to a lattice point. Dynamic programming and image blending techniques are applied on the overlapping regions to stitch the tiles.

2.3. Patch-based texture synthesis

Liang et al.[10] develop a patch-based synthesis algorithm. The basic idea of the algorithm is to synthesize textures by directly copying image patches from the input texture. The major difference from other image-based approaches is that they apply a modified approximate nearest neighbor technique to speed up the search for the best matched patch. With this improved search speed, the algorithm can run in real-time and reach similar image quality as other image-based synthesis algorithms. Image feathering technique is used to blend the overlapping regions of patches. This might blur the overlapping region slightly compared to the dynamic programming technique used in near-regular texture synthesis or the graph cut technique in the graph cut synthesis.

Patch placement in the patch-based approach is very different from that in near-regular texture synthesis. In the patch-based approach, the patch is rectangular and the patches are pasted in a scan-line order. Since the patch size and placement offset are arbitrarily defined by a user, they may not match the lattice structure of the input texture.

2.4. Regularized patch-based texture synthesis

We develop a regularized patch-based texture synthesis algorithm to deal with near-regular textures. We allow the parallelograms on a regular lattice to be deformed to quadrilaterals so that the texture elements can be separated by the deformed lattice. In other words, we deform a geometrically-irregular near-regular texture to a geometrically-regular near-regular texture. We then apply a modified patch-based approach to synthesize the geometrically-regular texture. Our modification to the patch-based approach allows patches to be pasted along the lattice axis direction and allow the patch shape to be a parallelogram rather than a rectangle. The patch-size and lattice construction vectors are provided by a user who identifies the underlying lattice structure of the input near-regular texture. A synthesized inverse deformation is used to warp the synthesized regular texture to a near-regular texture.

These four algorithms are summarized in Table 1 in terms of the patch shape/size determination, patch extraction, patch placement, and patch stitching methods used in these algorithms. Here, a *patch* is a 2D sample of the input texture. *Patch shape/size* and *extraction* refer to how the shape and size of the region are determined, and where each patch is located in the input texture. *Patch placement* and *stitching* refer to how the patches are placed and stitched in the synthesized texture.

3. Methods and Results

We compare the synthesis results of these four algorithms on regular textures and near-regular textures by comparing their underlying lattices. As a basis for comparison, the lattices of regular textures are automatically extracted, and those of near-regular textures are specified interactively. The regularity preservation test and the user evaluation test

Table 1. Summary and comparison of four synthesis algorithms.

Algorithms	**Graph cut**	**Near-regular synthesis**	**Regularized patch-based**	**Patch-based**
Patch shape/size	entire input texture	translational symmetry analysis, user intervention	user identified lattice, quadrilateral patch	user defined, rectangular patch
Patch extraction	N/A	lattice points	lattice points	random locations
Patch placement	random or maximal correlation locations	lattice points	lattice points	patch grids
Patch stitching	graph cut & blending	dynamic programming & blending	image-feathering	image-feathering

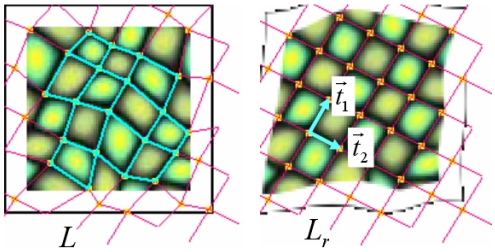


Figure 2. A near-regular texture overlaid with its lattice(left) and its geometrically regular counterpart(right), where L and L_r are the underlying lattices, and $\vec{t_1}$ and $\vec{t_2}$ are the generating vectors of the regular lattice L_r.

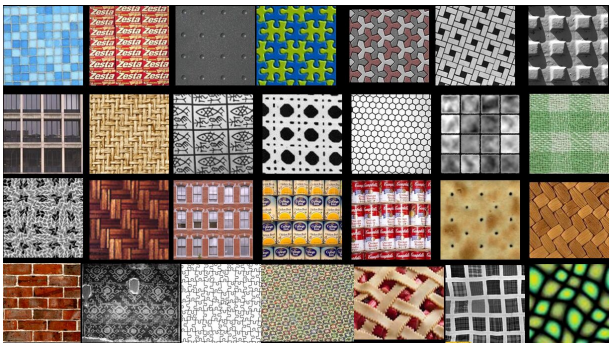


Figure 3. A sample set of near-regular textures used in this study

are used as a complementary pair to evaluate the synthesis results.

The first evaluation is objective since it checks if the global regularity is preserved based on a pair of well-defined, quantitative geometric and appearance regularity measures [13]. The basic idea is to view a near-regular texture as a statistical distortion of a regular, wallpaper-like congruent tiling and the degree of its regularity is determined by how much the geometry and appearance of individual tiles differ from their regular state. Figure 2 shows an example of a near-regular texture and its regular state with their underlying lattices overlaid. Since a near-regular texture can be transformed to its regular state by deforming its underlying lattice, the geometric regularity of the near-regular texture can be obtained by computing the amount of deformation between the underlying lattice of the near-regular texture and that of its regular counterpart.

The followings are the mathematical definitions of the geometric and appearance regularity.

Geometric regularity—G score

$$G = \sum_{i=1}^{N_i} \frac{(l_i - \|\vec{t_1}\|)^2}{\|\vec{t_1}\|^2} + \sum_{j=1}^{N_j} \frac{(l_j - \|\vec{t_2}\|)^2}{\|\vec{t_2}\|^2}$$

$$+ \sum_{k=1}^{N_k} \frac{(l_k - \|\vec{t_1} + \vec{t_2}\|)^2}{\|\vec{t_1} + \vec{t_2}\|^2} + \sum_{m=1}^{N_m} \frac{(l_m - \|\vec{t_1} - \vec{t_2}\|)^2}{\|\vec{t_1} - \vec{t_2}\|^2} \quad (1)$$

where $l_i, l_j, l_k,$ and $l_m,$ are the lengths of the links in lattice L corresponding to links in L_r along the directions of $\vec{t_1}, \vec{t_2}, \vec{t_1} + \vec{t_2}$, and $\vec{t_1} - \vec{t_2}$, respectively. $\vec{t_1}$ and $\vec{t_2}$ are computed from the input texture through an optimization process[13]. N_i, N_j, N_k and N_m are the total number of links in L corresponding to each direction.

Appearance regularity—A score

$$A = \frac{1}{m} \sum_{i=1}^{m} std([T_1(i), T_2(i), ..., T_n(i)]) \quad (2)$$

where T_j is a column vector consisting of all pixels of tile t of a regular texture; m is the number of pixels within a tile. The standard deviation is computed over all tiles.

We refer geometric regularity and appearance regularity to *G score* and *A score* respectively in this paper. Note that none of the synthesis algorithms in this comparison study optimizes the G score of a synthesized texture to match that of the input texture, i.e., the synthesis process is independent of the regularity measurements.

The second evaluation is subjective, human subjects are used to examine and score the overall quality of the synthesized textures in comparison to the input texture in terms of color/intensity, statistical variations, and structures. The results are summarized in Tables 2 and 3 respectively.

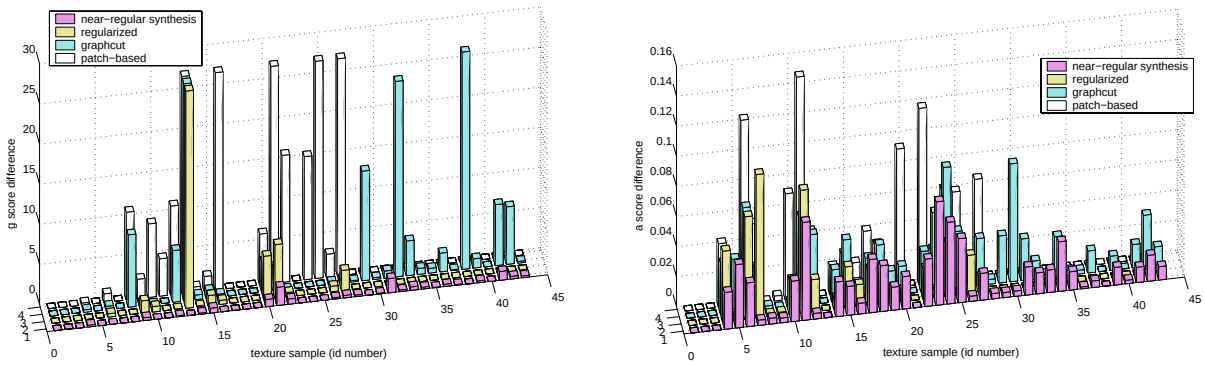
Table 2. Results of regularity preservation test on regular textures (top) and near-regular textures (bottom), where ✓denotes that regularity is preserved, and × denotes not. *g err.* and *a err.* are the difference of the geometric and appearance regularity between a synthesized texture and its input texture. The G score error with asterisk denotes that no lattice structure exists in the texture and the maximum score error is assigned.

Regular Textures		**Graph cut**			**Near-regular synthesis**			**Regularized patch-based**			**Patch-based**		
Textures	Description		g err.	a err.		g err.	a err.		g err.	a err.		g err.	a err.
1	wallpaper	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00
2	wallpaper	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00
3	wallpaper	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00	✓	0.00	0.00
4	jigsaw puzzle	✓	0.29	0.03	✓	0.21	0.02	✓	0.00	0.05	✓	0.24	0.04
5	pavement tiles	✓	0.00	0.04	✓	0.00	0.04	✓	0.00	0.00	✓	0.00	0.01
success rate(%), average		**100**	**0.06**	**0.014**	**100**	**0.04**	**0.013**	**100**	**0**	**0.010**	**100**	**0.05**	**0.010**

Near-regular Textures		**Graph cut**			**Near-regular synthesis**			**Regularized patch-based**			**Patch-based**		
Textures	Description		g err.	a err.		g err.	a err.		g err.	a err.		g err.	a err.
6	punched card	✓	0.4	0.07	✓	0.0	0.03	✓	0.4	0.07	×	1.1	0.12
7	hexagonal net	✓	0.0	0.03	✓	0.0	0.00	✓	0.0	0.10	✓	0.0	0.05
8	metal	×	9.0	0.00	✓	0.0	0.00	✓	0.0	0.00	×	10.9	0.00
9	ceramic tiles	✓	0.3	0.00	✓	0.4	0.00	✓	1.5	0.00	×	2.4	0.00
10	fish tiles	✓	0.1	0.00	✓	0.0	0.03	✓	0.7	0.02	×	9.1	0.07
11	wall	✓	0.0	0.06	✓	0.0	0.07	✓	0.0	0.08	×	4.7	0.15
12	squares	×	6.4	0.05	✓	0.1	0.00	✓	0.1	0.02	×	11.1	0.04
13	pavement tiles	×	27.1*	0.00	✓	0.0	0.00	×	27.1*	0.00	×	27.1*	0.00
14	cans	✓	0.6	0.02	✓	0.1	0.02	✓	0.1	0.01	×	0.7	0.02
15	swirl	✓	1.0	0.04	✓	0.6	0.02	✓	1.3	0.03	×	1.8	0.03
16	basket	✓	0.4	0.01	✓	0.0	0.01	✓	0.5	0.01	×	27.1*	0.02
17	fabric	✓	0.0	0.03	✓	0.0	0.04	✓	0.0	0.02	✓	0.0	0.04
18	fabric	✓	0.0	0.03	✓	0.0	0.03	✓	0.0	0.02	✓	0.0	0.03
19	fabric	✓	0.0	0.02	✓	0.0	0.02	✓	0.0	0.01	✓	0.0	0.01
20	knotted mat	✓	3.8	0.02	✓	0.9	0.02	✓	5.4	0.01	×	6.3	0.09
21	pie	×	2.9	0.00	✓	2.3	0.00	×	6.7	0.00	×	27.1*	0.00
22	fabric	✓	0.1	0.03	✓	0.6	0.03	✓	0.5	0.02	×	15.8	0.12
23	toothpastes	✓	0.0	0.05	✓	0.0	0.07	✓	0.0	0.01	✓	0.0	0.02
24	windows	✓	0.0	0.08	✓	0.0	0.05	✓	0.0	0.04	×	15.3	0.06
25	windows	✓	0.0	0.04	✓	0.0	0.04	✓	0.0	0.00	×	27.1	0.06
26	fabric	×	0.3	0.00	✓	0.2	0.00	✓	0.7	0.03	×	2.9	0.01
27	basket	✓	0.5	0.03	✓	0.3	0.02	×	2.5	0.01	×	27.1*	0.07
28	fabric	✓	0.2	0.00	✓	0.2	0.00	N/A			N/A		
29	squares&hexagons	×	13.7	0.03	×	0.0	0.00	N/A			N/A		
30	mosaic	×	0.2	0.08	✓	0.1	0.00	N/A			N/A		
31	jigsaw puzzle	✓	0.9	0.03	×	1.8	0.02	N/A			N/A		
32	fabric	×	24.4	0.01	✓	0.2	0.01	N/A			N/A		
33	cracker	×	4.3	0.00	✓	0.4	0.01	N/A			N/A		
34	brick wall	×	0.7	0.03	✓	0.2	0.03	N/A			N/A		
35	brick wall	×	0.6	0.00	✓	0.0	0.01	N/A			N/A		
36	brick wall	×	2.3	0.00	✓	0.3	0.00	N/A			N/A		
37	brick wall	✓	0.1	0.01	✓	0.1	0.00	N/A			N/A		
38	brick wall	×	27.1*	0.00	✓	0.0	0.00	N/A			N/A		
39	brick wall	×	1.2	0.01	✓	0.2	0.01	N/A			N/A		
40	carpet	✓	0.0	0.00	✓	0.0	0.00	N/A			N/A		
41	rug	×	7.6	0.02	✓	1.1	0.01	N/A			N/A		
42	rug	×	7.2	0.03	✓	0.3	0.02	N/A			N/A		
43	cans	×	0.2	0.01	✓	0.2	0.01	N/A			N/A		
success rate(%), average		**55**	**3.77**	**0.023**	**95**	**0.28**	**0.017**	**86**	**2.16**	**0.023**	**23**	**9.88**	**0.046**

3.1. Regularity preservation test

On the tested regular textures, all four synthesis algorithms preserve the global regularity almost equally well. On the tested near-regular textures, their performance differs greatly. The success rates of the graph cut approach, the near-regular synthesis approach, the regularized patch-based approach, and the patch-based approach are 55%, 95%, 86%, and 23%, respectively. We also included six



(a) G score errors (b) A score errors

Figure 4. The G and A score errors of synthesized textures, where the x-axis is the texture sample ID, and the z-axis is the G/A score difference between a synthesized texture and its original input texture. The numeric value of G/A score difference of each texture is listed in Table 2.

synthesis results of the *image quilting* approach[4] from its website[1]. The global regularities of the four out of the six tested textures are not preserved. The regularity difference between the input and the synthesized texture pair from each algorithm is shown in Table 2 and displayed as bar diagrams in Figure 4 for each input texture respectively[2].

3.2. Analysis of algorithms based on regularity preservation test

The above results indicate the importance of respecting global regularity of a near-regular texture during synthesis. The graph cut approach[9], near-regular synthesis[14], the patch-based approach[10] and the regularized patch-based approach handle global regularity in different ways and levels. The patch-based approach does not have any explicit mechanism to preserve the regularity (see Table 1). Although one can set an appropriate patch size and placement offset for regular textures, the patch-based approach cannot handle near-regular textures well. In particular, because its patch extraction algorithm does not utilize global regularity to constrain the sampling position be at the lattice points, it increases the chance that the best matched patch may not align with the boundary of a texture element. In the worst situations, this may cause the patch-based approach to totally break the texture elements and violate global and local regularity in synthesized textures.

Unlike the patch-based approach, the regularized patch-based approach utilizes a user-specified lattice in the synthesis process (Table 1). This lattice information is used to set the patch shape/size, patch extraction and placement locations. This helps to locate the tiles correctly in the input texture and to paste the tiles at right positions in the synthesized texture. Therefore, the regularized patch-based approach can preserve the global regularity much better than the patch-based approach.

The graph cut approach attempts to handle the global regularity by incorporating a local correlation technique to determine the best pasting location so that the underlying periodicity, if it exists, can be preserved (Table 1). This is the reason why the graph cut approach performs much better than the patch-based approach (Table 2). The correlation-based patch placement works well on regular textures and some near-regular textures; however, for near-regular textures in which the color/intensity of texture elements is not regular, the correlation technique cannot guarantee the preservation of global regularity. This is especially true when an input texture contains an interlocked structure, such as woven fabric or brick walls.

The near-regular texture synthesis approach utilizes the translational symmetry property to analyze the global regularity of near-regular textures. This analysis helps to identify the underlying lattice of the input texture and the lattice determines the parameters of patch size/shape, patch extraction and patch placement locations. A difference between the the regularized patch-based approach and near-regular texture synthesis approach is that the former samples tiles without overlapping while the latter samples tiles with overlapping size at half or third quarters of a tile. There are two advantages of sampling tiles with overlapping. First, searching the best matched tile is more robust as there is a larger overlapping between tiles. Second, the overlapping can be used to adjust the pasting location so that the best matched tile is registered to the existing synthesized texture. Through use of overlapped tile sets, the near-regular texture synthesis approach can preserve the global regularity better than the regularized patch-based approach. Moreover, the interlocked structure of a near-regular texture is also better preserved because tiles are extracted and pasted/registered

[1] http://www.cs.berkeley.edu/∼efros/research/quilting.html.

[2] For a detailed examination of all input and output texture pairs see http://graphics.cs.cmu.edu/data/texturedb/gallery/ or http://www.cs.nctu.edu.tw/∼wclin/nrtcmp/nrtcmp.html.

accurately at lattice points. This is the reason that the near-regular texture outperforms the graph cut approach in brick wall textures and woven fabric textures.

3.3. User evaluation test

10 subjects (students) participated in the user evaluation test. Each subject is given both the input and synthesized texture pair on a computer screen and asked to give a score between 1 to 4 (worst to best) in responding to the following question: how well the image characteristics of the input texture are faithfully preserved in the synthesized texture? We compute the mean and standard deviation of the scores from 10 subjects for each synthesized texture. The results are listed in Table 3. The standard deviation of the user scores of *each texture* in Table 3 shows the degree of agreement among different subjects. We use red and blue to indicate the standard deviations of those synthesized textures that are greater than 1.15 (disagree tendency) and less than 0.33 (agree tendency) respectively. Synthesized textures whose standard deviations of the user scores are small (less than 0.33) are mostly generated by the graph cut approach and the patch-based approach. With a closer inspection, we can find that these synthesized textures either receive high scores or low scores. This is not surprising since these synthesized results are either very good or very poor, thus different subjects tend to give similar scores to these textures.

3.4. Analysis on user evaluation results

ANOVA [15] is a common statistical analysis tool that tests if the variations among data are caused by some potential factors or by chance. to analyze the user evaluation scores. There are two major findings: (1)near-regular texture synthesis algorithm performs statistically significantly better than the other three texture synthesis algorithms (all three pair-wise comparisons have $p < 0.001$ in Table 4); (2)the user scores are highly correlated to the degrees of regularity, i.e. the scores for regularity-preserved textures and regularity-violated textures differ statistically significantly (F(1,1398)=879.81, $p < 0.001$ in Table 5), with averages 3.4 and 1.9 respectively.

Table 4 summarizes the performance of four texture synthesis algorithms in the user evaluation test. From Table 4, it demonstrates that the performance of the near-regular synthesis approach is statistically significantly better than the other three approaches in the user evaluation test. The near-regular texture synthesis approach not only gets higher scores but also has smaller standard deviation (including variations across different users and tested textures), which implies that the evaluation scores of the near-regular synthesis approach, from different users on different tested textures, are more consistent than the other three approaches. Furthermore, it shows that the performance of the graph cut approach and the patch-based approach are less consistent

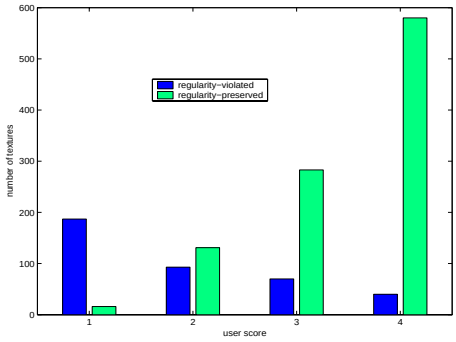


Figure 5. Histogram of the user scores of all synthesized textures, where green bars represent those textures whose global regularity is preserved and blue bars represent those violated. The range of user score, from best to worst, is 4 to 1. This plot shows that the preservation of global regularity has strong correlation with user evaluation scores, but is not the only criterion that humans use to evaluate a synthesized texture.

across different near-regular textures than that of the near-regular synthesis approach and the regularized patch-based approach.

The second finding in ANONA shows that global regularity is actually an important factor when humans evaluate the quality of the synthesized regular and near-regular textures. We plot the histogram of the user scores of all textures in Figure 5. Among the 1400 synthetic textures tested in the user evaluation test (10 subjects $\times$ 140 synthesized textures), 1010 textures are regularity-preserved. The histogram of the user scores of regularity-preserved textures is plotted in green bars, while that of the regularity-violated textures is plotted in blue bars. One can observe that most of the regularity-preserved synthesized textures are located in higher score region while most of the regularity-violated synthesized textures are located in lower score region. From Table 5 and Figure 5, we observe strong correlations between the preservation of global regularity and the (high) user evaluation scores.

Global regularity, however, is not the only criterion human subjects use to evaluate the quality of textures. Several regularity-preserved synthesized textures have received low scores and some regularity-violated synthesized texture received high scores. Colors/intensity, statistical variations and other factors also affect one's judgment.

4. Conclusion and future work

In this paper, we compare the performance of four texture synthesis algorithms on regular and near-regular textures. The global regularity property of the regular and near-regular textures provides us a more consistent and objective criterion to evaluate the synthesis results.

Our comparison study shows that near-regular texture synthesis challenges several state-of-the-art algorithms, in-

Table 3. Results of user evaluation on regular textures (top) and near-regular textures (bottom), where ✓ denotes that regularity is preserved, and × denotes not. The numbers in the table are the mean and standard deviation of scores given by 10 subjects. The range of score, from best to worst, is 4 to 1. The standard deviations of the synthesized textures that are greater than 1.15 and less than 0.33 are shown in red texts and blue texts respectively.

Regular Textures		**Graph cut**			**Near-regular synthesis**			**Regularized patch-based**			**Patch-based**		
Textures	Description		mean	std		mean	std		mean	std		mean	std
1	wallpaper	✓	3.5	0.71	✓	3.6	0.70	✓	3.3	0.95	✓	3.4	0.84
2	wallpaper	✓	3.6	0.70	✓	3.6	0.84	✓	3.5	0.71	✓	3.5	0.71
3	wallpaper	✓	3.8	0.42	✓	3.8	0.42	✓	3.8	0.42	✓	3.8	0.42
4	jigsaw puzzle	✓	3.9	0.32	✓	3.8	0.42	✓	3.6	0.70	✓	3.6	0.70
5	pavement tiles	✓	3.9	0.32	✓	3.6	0.52	✓	3.7	0.48	✓	3.7	0.48
success rate(%), mean, std		**100**	**3.7**	**0.53**	**100**	**3.7**	**0.59**	**100**	**3.6**	**0.67**	**100**	**3.6**	**0.64**

Near-Regular Textures		**Graph cut**			**Near-regular synthesis**			**Regularized patch-based**			**Patch-based**		
Textures	Description		mean	std		mean	std		mean	std		mean	std
6	punched card	✓	3.8	0.63	✓	3.1	0.88	✓	2.6	1.07	×	1.5	0.85
7	hexagonal net	✓	3.7	0.48	✓	3.6	0.70	✓	3.4	0.84	✓	3.5	0.71
8	metal	×	1.5	0.97	✓	3.3	0.82	✓	3.4	0.70	×	1.3	0.67
9	ceramic tiles	✓	3.5	0.71	✓	3.1	0.88	✓	2.8	0.79	×	2.4	0.97
10	fish tiles	✓	3.7	0.48	✓	3.7	0.48	✓	2.6	0.84	×	1.1	0.32
11	wall	✓	3.9	0.32	✓	3.3	0.82	✓	3.7	0.48	×	1.0	0.00
12	squares	×	1.1	0.32	✓	3.4	0.52	✓	3.5	0.71	×	1.1	0.32
13	pavement tiles	×	1.4	0.52	✓	3.6	0.52	×	3.2	0.63	×	1.1	0.32
14	cans	✓	3.0	0.82	✓	3.0	1.05	✓	3.5	0.53	×	2.5	0.97
15	swirl	✓	3.4	0.84	✓	3.4	0.84	✓	2.6	1.07	×	2.6	1.17
16	basket	✓	2.4	0.97	✓	3.3	0.82	✓	2.9	0.57	×	1.6	1.07
17	fabric	✓	3.9	0.32	✓	3.7	0.48	✓	3.5	0.71	✓	3.6	0.70
18	fabric	✓	3.7	0.48	✓	3.4	0.84	✓	2.8	0.92	✓	2.9	0.88
19	fabric	✓	3.8	0.42	✓	3.4	0.97	✓	3.0	0.94	✓	2.9	1.20
20	knotted mat	✓	3.8	0.42	✓	3.5	0.71	✓	3.4	0.84	×	1.3	0.48
21	pie	×	2.9	0.99	✓	3.6	0.70	×	2.3	0.48	×	1.1	0.32
22	fabric	✓	3.0	0.94	✓	3.7	0.48	✓	3.2	0.63	×	1.4	0.97
23	toothpastes	✓	3.2	0.92	✓	3.8	0.42	✓	3.5	0.71	✓	2.8	1.03
24	windows	✓	3.4	0.97	✓	2.6	0.97	✓	2.4	1.07	×	1.3	0.48
25	windows	✓	3.3	0.82	✓	3.7	0.48	✓	3.5	0.71	×	1.2	0.63
26	fabric	×	3.4	0.70	✓	3.2	0.79	✓	3.0	0.94	×	1.4	0.52
27	basket	✓	3.2	0.63	✓	3.5	0.71	×	2.2	0.63	×	1.1	0.32
28	fabric	✓	3.9	0.32	✓	3.6	0.52	N/A			N/A		
29	squares&hexagons	×	1.5	0.97	×	3.1	0.74	N/A			N/A		
30	mosaic	×	2.2	1.03	✓	3.6	0.70	N/A			N/A		
31	jigsaw puzzle	✓	3.5	0.53	×	2.4	0.97	N/A			N/A		
32	fabric	×	1.3	0.48	✓	3.8	0.42	N/A			N/A		
33	cracker	×	2.2	0.92	✓	3.7	0.48	N/A			N/A		
34	brick wall	×	2.6	1.17	✓	3.8	0.42	N/A			N/A		
35	brick wall	×	3.4	0.70	✓	3.4	0.70	N/A			N/A		
36	brick wall	×	2.9	0.57	✓	3.3	0.48	N/A			N/A		
37	brick wall	✓	3.2	0.79	✓	3.6	0.70	N/A			N/A		
38	brick wall	×	1.1	0.32	✓	3.7	0.48	N/A			N/A		
39	brick wall	×	1.2	0.42	✓	3.1	0.88	N/A			N/A		
40	carpet	✓	3.8	0.42	✓	3.7	0.67	N/A			N/A		
41	rug	×	2.3	0.67	✓	2.7	1.25	N/A			N/A		
42	rug	×	2.4	0.84	✓	3.3	0.48	N/A			N/A		
43	cans	×	2.7	1.16	✓	3.4	0.70	N/A			N/A		
success rate(%), mean, std		**55**	**2.9**	**1.13**	**95**	**3.4**	**0.76**	**86**	**3.0**	**0.87**	**23**	**1.9**	**1.10**

cluding the graph cut[9], patch-based[10], and image quilting approaches[4]. The results from the regularized patch-based approach and the near-regular texture synthesis approach show that the global regularity of the near-regular textures can be better preserved if the synthesis algorithm analyzes the underlying lattice structure and uses this information in the synthesis process. The synthesis results by the near-regular texture synthesis algorithm demonstrate that

Table 4. Summary of the user evaluation test on the four texture synthesis algorithms. Across different users and textures, ANOVA method[15] is used to verify the statistical significant difference between the near-regular texture synthesis (NRTS) algorithm and the other three. p-values show that the three pairwise comparisons are all statistically significant.

Algorithms	**Graph cut**	**Near-regular texture synthesis**	**Regularized patch-based**	**Patch-based**
mean score	3.0	3.4	3.1	2.2
standard deviation	1.11	0.75	0.86	1.24
standard error of the mean	0.054	0.036	0.052	0.075
ANOVA compare against the NRTS	F(1,858)=54.18 $p < 0.001$	N/A	F(1,698)=27.37 $p < 0.001$	F(1,698)=315.95 $p < 0.001$

Table 5. Comparison of user scores on regularity-preserved and regularity-violated textures. ANOVA method[15] (section 3.4) is used to verify if there is a statistically significant difference between the mean of the two types of the synthesized textures. This table shows that the global regularity-preserved textures get higher scores, and the score difference is statistically significant.

	Regularity preserved	**Regularity violated**	**Total**
texture counts	1010	390	1400
mean score	3.41	1.91	3.00
std	0.77	1.03	1.09
standard error	0.024	0.052	0.029
ANOVA	F(1,1398)=879.81, $p < 0.001$		

both the global regularity and local randomness of a near-regular texture can be faithfully preserved. This is reflected in the quantitative evaluations of the regularity preservation test and the user evaluation test, where the near-regular texture synthesis approach has higher success rate (95%), smaller G score error, and statistically significantly higher human user scores than the other methods ($p < 0.001$, Table 2).

The user evaluation results (Tables 5 and 4) indicate that global regularity is indeed an important factor when human subjects evaluate the faithfulness of the synthesized regular or near-regular textures. The mean user score of regularity-preserved textures and that of the regularity-violated are statistically significantly different ($p < 0.001$).

References

[1] M. Ashikhmin. Synthesizing natural textures. In *Symposium on Interactive 3D Graphics*, pages 217–226, 2001.

[2] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture mixing and texture movie synthesis using statistical learning. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):120–135, 2001.

[3] J. S. D. Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *ACM SIGGRAPH*, pages 361–368, 1997.

[4] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *ACM SIGGRAPH*, pages 341–346, 2001.

[5] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *ICCV (2)*, pages 1033–1038, 1999.

[6] B. Grünbaum and G. Shephard. *Tilings and Patterns*. W.H. Freeman and Company, New York, 1987.

[7] J. Hays, M. Leordeanu, A. Efros, and Y. Liu. Discovering texture regularity as a higher-order correspondence problem. In *ECCV*, 2006.

[8] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *ACM SIGGRAPH*, pages 229–238, 1995.

[9] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. In *ACM SIGGRAPH*, pages 277–286, 2003.

[10] L. Liang, C. Liu, Y. Xu, B. Guo, and H. Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics*, 20(3):127–150, 2001.

[11] W.-C. Lin, J. Hays, C. Wu, V. Kwatra, and Y. Liu. A comparison study of four texture synthesis algorithms on regular and near-regular textures. Technical Report CMU-RI-TR-04-01, Robotics Institute, Carnegie Mellon University, 2004.

[12] Y. Liu, R. T. Collins, and Y. Tsin. A computational model for periodic pattern perception based on frieze and wallpaper groups. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 26, pages 354 – 371, 2004.

[13] Y. Liu, W.-C. Lin, and J. Hays. Near-regular texture analysis and manipulation. In *ACM SIGGRAPH*, pages 368–376, 2004.

[14] Y. Liu, Y. Tsin, and W.-C. Lin. The promise and perils of near-regular texture. *International Journal of Computer Vision*, 62(1-2):145–159, 2005.

[15] J. Neter, M. H. Kutner, W. Wasserman, and C. J. Nachtsheim. *Applied Linear Statistical Models*. McGraw-Hill, 4th edition, 1996.

[16] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–71, 2000.

[17] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *ACM SIGGRAPH*, pages 479–488, 2000.

[18] L.-Y. Wei and M. Levoy. Texture synthesis over arbitrary manifold surfaces. In *ACM SIGGRAPH*, pages 355–360, 2001.

[19] J. Zhang, K. Zhou, L. Velho, B. Guo, and H.-Y. Shum. Synthesis of progressively-variant textures on arbitrary surfaces. In *ACM SIGGRAPH*, pages 295–302, 2003.