The Promise and Perils of Near-regular Texture*

Yanxi Liu, Yanghai Tsin and Wen-Chieh Lin
The Robotics Institute, Carnegie Mellon University
5000 Forbes Ave. Pittsburgh, PA 15213. USA
{yanxi,ytsin,wclin}@cs.cmu.edu

Abstract

Motivated by the low structural fidelity for near-regular textures in current texture synthesis algorithms, we propose and implement an alternative texture synthesis method for near-regular texture. We view such textures as statistical departures from regular patterns and argue that a thorough understanding of their structures in terms of their translation symmetries can enhance existing methods of texture synthesis. We demonstrate the perils of texture synthesis for near-regular texture and the promise of faithfully preserving the regularity as well as the randomness in a near-regular texture sample.

*This work is partially funded by an NSF research grant # IIS-0099597.

1 Motivation

Near-regular textures are common in our daily life. They can be observed in man-made products, by hand or by machine, ranging from buildings to fabrics, as well as in nature and biological process of life science [15, 3, 33, 38, 17]. Humans have an innate ability to perceive and take advantage of symmetry [22]. Rao and Lohse showed in [31] that regularity plays an important role in human texture perception. However, it is not obvious how to automate this powerful insight.

Mathematically speaking, *regular texture* refers to *periodic patterns* that present non-trivial translation symmetry, with the possible addition of rotation, reflection and glide-reflection symmetries [29, 7, 16]. When studying periodic patterns, a useful fact from mathematics is the answer to Hilbert's 18th problem: there is only **a finite number of symmetry groups** for all possible periodic patterns in dimension n [2]. When $n = 1$ there are seven **frieze groups**, and when $n = 2$ there are 17 **wallpaper groups**. Here *group* is referring to the *symmetry group* of a periodic pattern. A symmetry group is composed of transformations that keep the pattern setwise invariant.

In computer vision and computer graphics, the application of this classic mathematics for regular or near-regular pattern analysis has yet to be fully explored. Only recently, have computer algorithms of symmetry group classification been developed for periodic patterns in real images under Euclidean [25, 27] and affine transformations [26], based on a careful analysis of the basic *tile* shapes of regular patterns. In computer graphics, one interesting recent work [21] is to find Escher-like tilings by deforming a single closed planar figure to tile a plane.

Near-regular texture is referring to textures that are not strictly symmetrical. The irregularity can be caused by various statistical departures from regular textures. These departures can happen along different dimensions of symmetry [24], for example, color (single, multi) [34], intensity (irregular statistical alterations, random noise), global or local geometric deformations (affine, projective, random) [25, 26], and resolution. See Figure 1 from [24] for some examples of symmetry dimensions. The focus of this paper is on faithful texture synthesis of near-regular textures where departure from regularity is primarily caused by statistical color and intensity variations, while the underlying structural regularity remains. There are many examples of this type of near-regular textures, e.g. brick walls, tiled floors, carpets, woven sheets, where the texture patterns (each brick, tile, straw or bamboo strip) vary only locally. The idea of viewing a random texture as a distorted version of a regular texture was expressed in an early paper by Zucker [42].

Existing work on texture synthesis has achieved impressive results for a variety of different types of textures e.g. [10, 12, 1, 11, 20, 36, 19, 37, 23, 40]. These texture synthesis algorithms share a common theme of local neighborhood-based statistical approaches. Distinctions can be drawn between approaches that constructively establish statistical models for the input texture [9, 41] versus others that seek to find matching joint statistics directly in the input samples [10, 30, 40]. More recently, non-parametric estimation of texture PDFs has become popular [12, 36, 11, 23]. These texture synthesis algorithms are relatively simple to implement, fast to run [36, 23] and able to reproduce a large varieties of textures, *from regular to random*, as claimed by the authors. However, after reviewing the results of existing

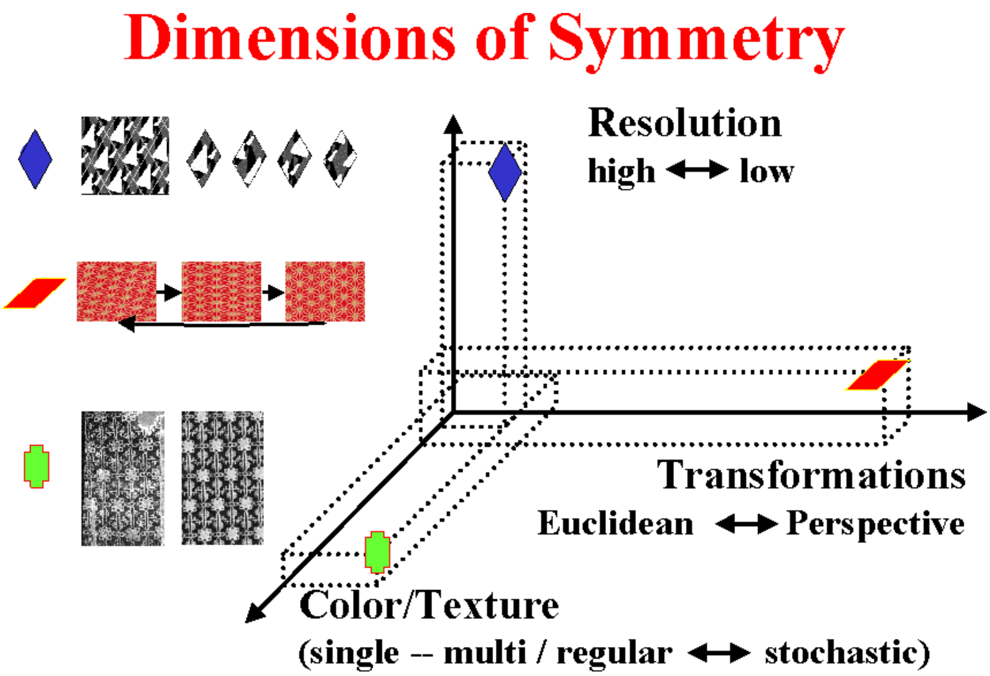


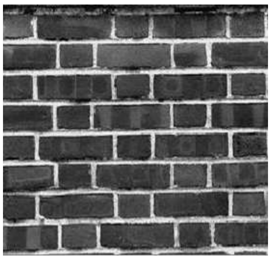
Figure 1: Symmetry or regularity of images spans a continuous, multi-dimensional space.

work applied to near-regular textures, we observe that the structural regularity is usually not well preserved in the synthesized texture. This is especially true when the input sample has interlocked near-regular patterns, or is oriented obliquely. For example, to the best of our knowledge, we have not yet seen an existing texture synthesis algorithm that preserves the regularity in a brick wall sample (Figure 2 (a)). In addition, the structural property of near-regular textures has not been used as an objective measure for texture synthesis algorithms.

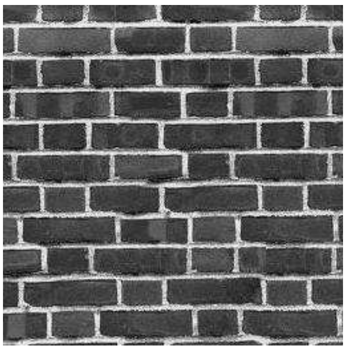
This situation motivates us to propose and implement an alternative texture synthesis method for near-regular texture that is particularly faithful to its structural property while preserving the randomness observed in the input data. Figure 2 and Figure 3 demonstrate two sample results from our texture synthesis algorithm in contrast to the texture synthesis results reported in [11].

Section 2 defines basic properties of regular texture in terms of its generating tile, symmetry groups and lattice types. In Section 3 we explain our texture analysis and synthesis algorithm and demonstrate some experimental results. Section 4 discusses several relevant issues in near-regular texture synthesis, from window size to the concept of textons. Section 5 concludes with a summary and future research directions.

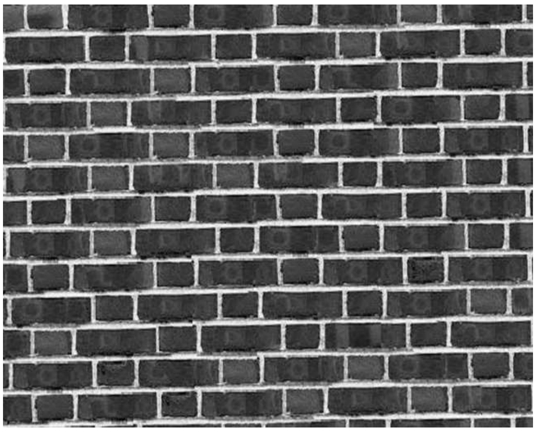
Figure 2: (a): input texture samples. (b): texture synthesis results from [11]. This is one of the best results on brick wall texture synthesis that we can find. However, the regularity in the input texture samples is not faithfully preserved in the synthesized texture: two short bricks are stacked together and there are more than two brick sizes in the synthesized image. (c): the texture synthesis results of our algorithm proposed in this paper.



(a) Input sample

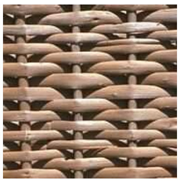


(b) Output of [11]

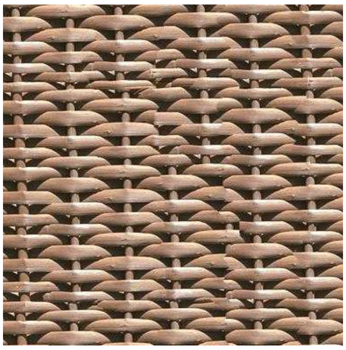


(c) Our result

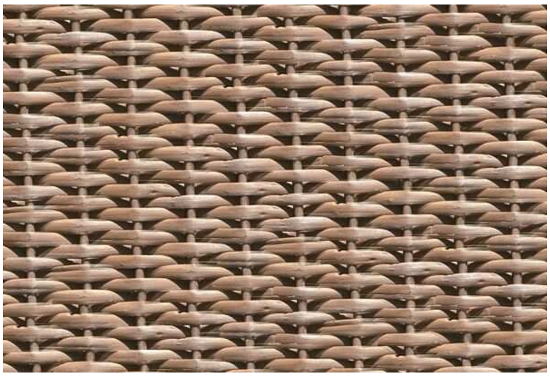
Figure 3: (a): input texture sample. (b): texture synthesis results from [11]. Straw pattern: one vertical line is terminated midway). (c): the texture synthesis results of our algorithm proposed in this paper.



(a) Input sample



(b) Output of [11]



(c) Our result

2 Regular Texture Analysis

A symmetry of a 2D periodic pattern P is a distance preserving mapping g: $R^2 \times I \Rightarrow R^2 \times I$ such that $g(P) = P$, where I can either be gray values in the range of $[0, 255]$ or RGB intensity values. It can be proven that all symmetries of P form its symmetry group. All the translation symmetries of a periodic pattern form its translation subgroup, a group generated by two linearly independent, shortest translation symmetries $\vec{t_1}, \vec{t_2}$ of P [32]. Mathematically speaking, symmetry groups are defined only for periodic patterns of infinite extent. In practice, we analyze a periodic pattern bounded within a finite image area, and thus use the concept of symmetry group G of P to mean G is the symmetry group of an infinite periodic pattern for which P is a finite region with more than one period.

Each 2D *regular texture* is a 2D periodic pattern that contains a non-empty parallelogram $\mathcal{T}$. The orbit of $\mathcal{T}$ under the action of its translation symmetry subgroup produces simultaneously a *covering* (no gaps) and a *packing* (no overlaps) of the original pattern [16, 32]. We call the smallest such parallelogram the **tile** of the texture. For a given regular texture its tile is uniquely defined in shape, size, and orientation but not in location, thus its pixelwise intensity and color content may vary, depending on where the lattice of the texture pattern is anchored.

A mature mathematical theory for wallpaper-like regular texture has been known for over 100 years [14, 16], namely the theory of wallpaper groups[1]. For monochrome planar periodic patterns, there are seventeen *wallpaper groups* describing patterns extended by two linearly independent translational gen-

[1] These groups are also called two dimensional Crystallographic groups [18]

erators. Despite the infinite variety of regular texture instantiations, this finite set of symmetry groups and their 17 corresponding lattice/tile structures completely characterize the possible structural symmetry of any 2D periodic pattern. There are only *five* possible lattice shapes [8], therefore *tile shapes*, and they form a shape hierarchy (Figure 4):

1. parallelogram,
2. rectangular,
3. rhombic,
4. square and
5. hexagonal.

Each lattice unit or tile shape is a parallelogram. A rectangular tile has angles of 90^o. A rhombic tile has equal-length edges. Square and hexagonal tiles are special cases of rectangle and rhombic, respectively.

Work in *structural texture analysis* [13, 28] is also based on the idea of a *unit pattern* together with a set of well-defined placement rules. However, its generality and computational tractability are limited: unit patterns are either regions centered about a local maximum that is bounded on all sides by local minima [13] or square texture regions with an unspecified window size [28]. The authors of [6] use mathematical tiling theory for the analysis of texture, but they do not take advantage of the complete characterization of lattice types and the inner structures of 2D regular texture afforded by wallpaper groups, and their characterization of pattern elements is dominated by the inertia feature alone.

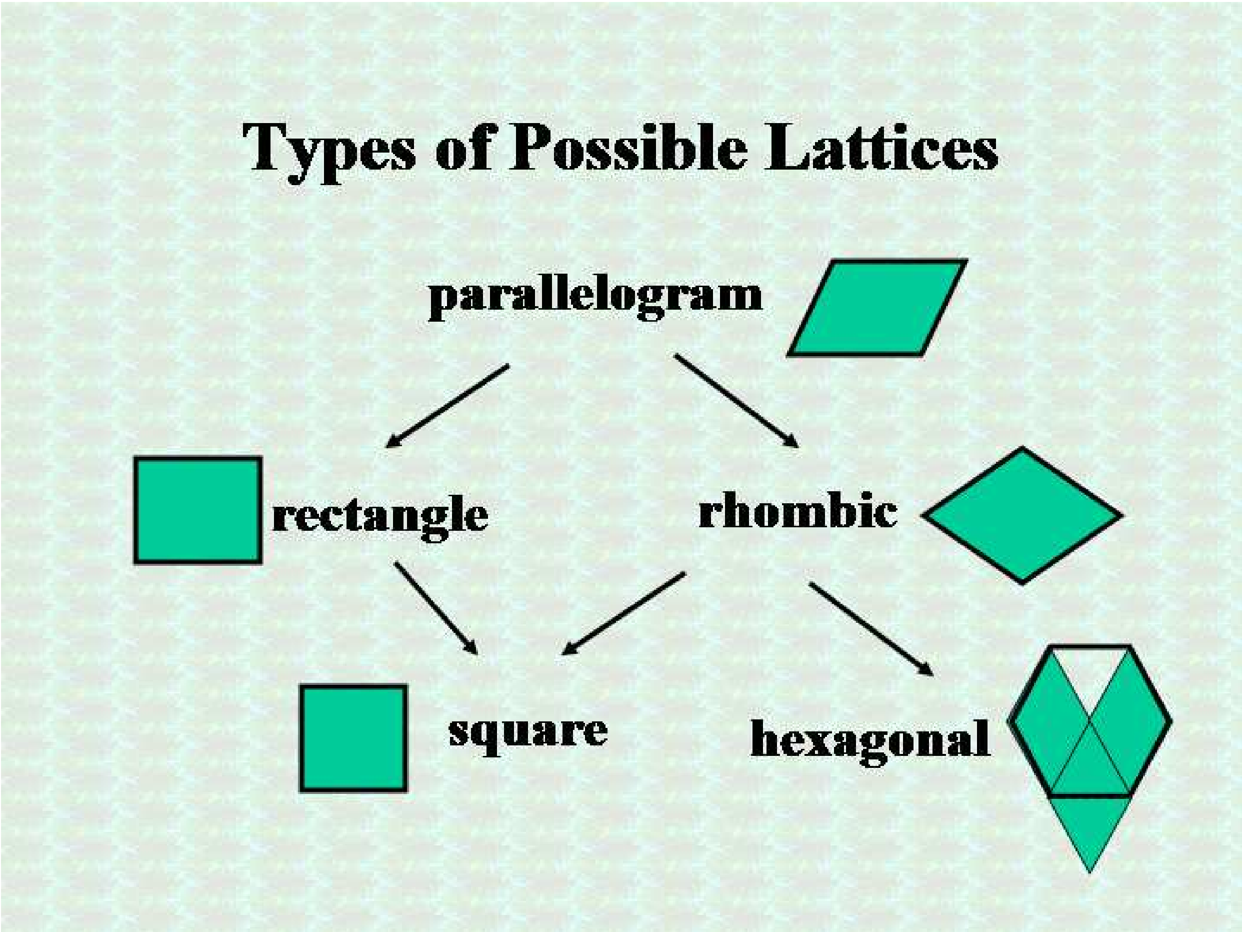
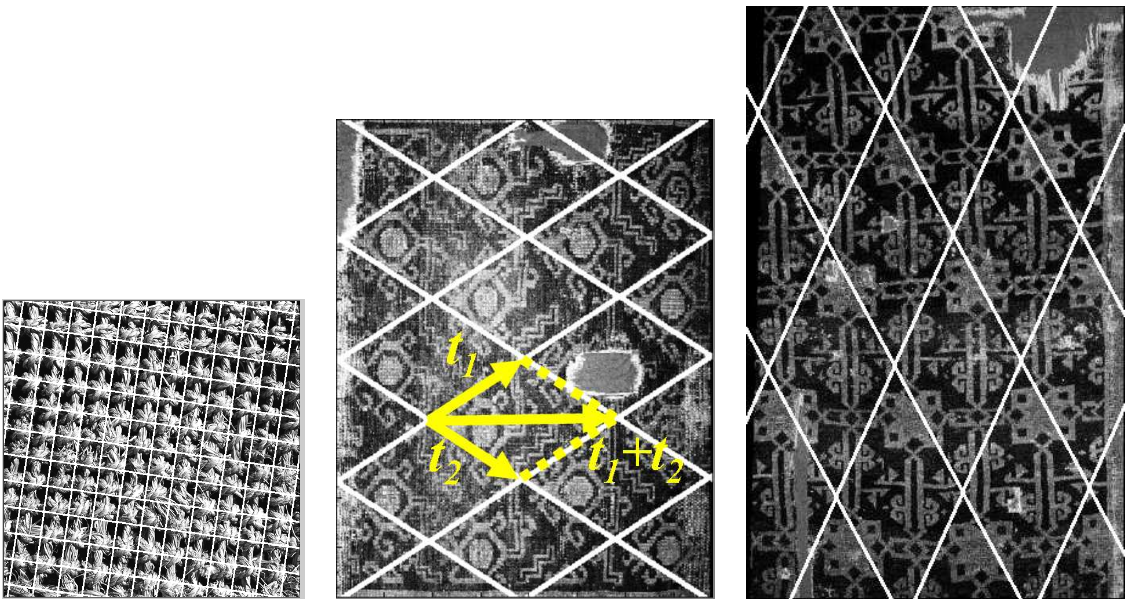


Figure 4: There are only five possible types of tiles in 2D regular textures.

One essential element in our method is to acknowledge the regularity in a near-regular texture by first locating the generating "tile" precisely. This computational effort is guided by the basic principles and understandings of tiles and their symmetries, as concisely summarized in their wallpaper groups. In order to find tiles in a given 2D near-regular pattern we developed an algorithm in [25, 27], based on *regions of dominance*, for locating the underlying lattice of a given pattern. Figure 5 shows the variations of shapes, sizes and orientations of lattices automatically generated from three real-world near-regular patterns. In addition, the generating translation vectors and a typical tile are indicated as an example on one of the three textures. The $\vec{t_1}, \vec{t_2}$ translational symmetries of a regular pattern alone fix the size, shape and orientation of the lattice, but leave open the question of where the lattice is located on the pattern. Any offset of the lattice on a pattern carves the pattern into a set of similar tiles, any one of which generates the whole 2D pattern. For perception purposes [26, 27], a motif (a representative tile) can be chosen that reflects the symmetry property of the whole patterns. For synthesis proposes, on the other hand, the tiles could be chosen to optimize the "blending" effects (Section 3.1).

Figure 5: Examples of imperfect, real-world near-regular patterns overlayed with automatically detected underlying lattices using an algorithm developed in [27]. Notice the different shape, size and orientations of the tiles. The arrows drawn on the middle image give an example of the two shortest generating translations $\vec{t}_1, \vec{t}_2$ for this texture pattern. The region bounded by the two vectors (enclosed by the two vectors and two dotted lines) indicates a tile for this pattern. For each near-regular texture, there exists a well-defined tile that is bounded by the two linearly independent translations of its wallpaper pattern.



3 Our Method for Texture Synthesis

Perfect regularities are rarely found in the real world, while varying degrees of deviations from regularity are common to observe. Our research interest is to capture both regularity and randomness by combining the mathematical theory of regular patterns with statistical modeling of data in texture analysis and synthesis.

We treat a set of tiles carved by the detected lattice as multiple samples of the same tile. We define these tiles as *minimum tiles* $\{t_i\}$ since by definition of regular patterns there are no 2D regions smaller than these tiles that can tile the whole texture pattern under its translation subgroup. Correspondingly, we define a set of *maximum tiles* $\{T_i\}$ by circumscribing each minimum tile t_i with the smallest rectangularly shaped convex hull. Note that depending on the shape and orientation of the t_is, maximum tiles T_i can be in any possible orientation and perspective ratio. The minimum (maximum) tile set also contains tiles centered on half-way shifted lattice points (i.e. at locations $((n + 1/2)\vec{t_1}, (m + 1/2)\vec{t_2})$ from the anchored lattice position, where m, n are integers). For texture synthesis, each time a tile is randomly chosen from these tile sets. This process provides the promise of capturing statistical color and intensity variations from different tiles, which can give the generated texture more natural appearance, while reproducing its regularity.

3.1 Algorithm for texture synthesis of near-regular patterns

Input: a sample near regular texture S

Output: a synthesized texture S' statistically similar to S.

Stage 1 (analysis):

- First determine the translational symmetry vectors $\vec{t_1}, \vec{t_2}$ from the given sample near-regular texture pattern. In our experiments, these vectors are either be (1) computed automatically [26, 27]; (2) indicated by the user by clicking on three nearest corresponding points of the texture, or (3) computed first and verified by the user.

- Determine where the lattice should be anchored such that all the *minimum tiles* t_i are uniquely defined. This is one parameter that the user can control to make the boundary of the tiles align with low frequency regions for the benefit of better blending results. In our experiments, most lattice locations have been hand located.

- For each t_i construct the corresponding *maximum tile* sets T and T_h. T contains all the T_is centered on the lattice points, and T_h contains all the T_is centered on the half-way shifted lattice points.

Figure 6 shows the brick wall sample as an example, each rhombic shaped tile t_i is enclosed by a rectangular maximum tile.

Stage 2 (synthesis):

1. Start from top left corner with a random tile chosen from T.

2. One tile is added at a time into the synthesized texture in a scanline order along the direction of $\vec{t_1}+\vec{t_2}$ with a step size of $|(\vec{t_1}+\vec{t_2})/2|$. When the process reaches the right boundary of the desired image size, one tile is placed in the direction of $\vec{t_2}-\vec{t_1}$ with a step size of $|(\vec{t_2}-\vec{t_1})/2|$ from the left most synthesized tile of the current row.

3. At each lattice or half-way lattice point, alternatively select the T or T_h tile set. For each tile in the selected tile set, we compute its color difference to the existing synthesized image in the overlapped region where the tile is going to be pasted. The error function is: $F_{error}(im_1, im_2) = \Sigma_{i,j}(dist(im_1(i,j), im_2(i,j)))$ where $dist(im_1(i,j), im_2(i,j)) = \Sigma(abs(R_{i_1} - R_{i_2}) + abs(G_{i_1} - G_{i_2}) + abs(B_{i_1} - B_{i_2}))$ and R, G, B are rgb values of a pixel. A candidate tile set is formed by selecting those tiles that have the RGB intensity difference less than a threshold. A tile is then randomly picked from the candidate tile set. If the candidate set is empty, we pick the tile with minimum error to paste to the synthesized image. The size of the candidate tile set varies at every lattice point.

4. Register the selected candidate tile using a correlation-based method such that small movements around the current lattice point are possible.

5. Use dynamic programming to "stitch" together the overlapping tiles in a similar manner as described in [11]. The dynamic programming technique is applied separately along the horizontal and vertical directions.

6. When pasting a tile to the existing image, blending is applied to the boundaries where the dynamic programming along horizontal and vertical directions may have conflicting decisions. In other words, all pixels on the boundary of the selected tile and existing synthesized texture are either results of dynamic programming or blending. The blending is done on a padded region around the boundary of the selected tile t_i and existing synthesized tile $\mathbf{t}$, based on this formula:

$w(i,j) \times t_i(i,j) + (1 - w(i,j)) \times \mathbf{t(i,j)}$ where $0 <= w(i,j) <= 1$ depending on the distance from the pixel (i,j) to the boundary.

7. Repeat steps 2 through 6 until the whole image is synthesized.

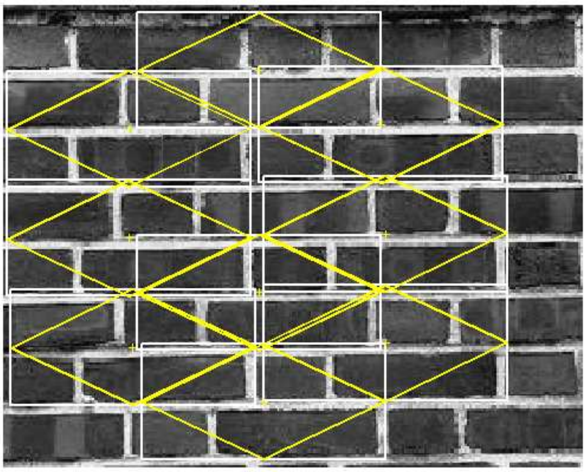
The reason we use maximum tiles instead of minimum tiles for synthesis is to have redundant overlapping regions for a smoother transition on the tile boundaries. Using this method, each tile has half to three quarters of overlap with the currently synthesized image. As a result, correlation-based registration can be done robustly. However, as departure from regularity in the texture increases, one can expect less coherence in the synthesized image. It takes about 20 seconds to synthesize an image of size 544 by 565 on a 2.2Ghz PC, using non-optimized Matlab code.

3.2 Experimental Results

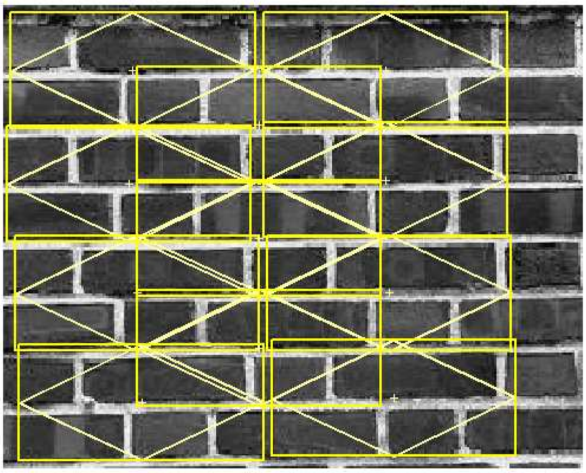
Images (c) in Figures 2 and 3 show our synthesized results in comparison with the corresponding results from [11]. Figure 6 shows both the minimum and the maximum tiles used in the brick wall example (Figure 2). Figure 7 demonstrates the difference between naive direct tiling and our random selection method. Figure 8 shows more sample results of our method. Figures 9, 10 and 11 demonstrate the synthesized results of three near regular textures where the lattices are automatically generated as shown in Figure 5. These experimental results reflect our intention of preserving the near-regular structure of the input texture *as well as* the statistical variations among and within the tiles.

Due to the blending procedure, the synthesized texture may appear not as sharp as the input texture (e.g. Image (c) of Figure 3). The top result shown in Figure 8 may appear more regular than the input texture as a result of using an error threshold that is too tight after a random candidate tile selection. There is a tradeoff between allowing more variations in the synthesized texture and keeping the textures over the stitching boundaries more similar to each other. However, there is a natural agreement between the probability of a tile appear in the input texture and its chance to appear in the synthesized texture. For example, tiles containing holes in the rug textures (Figure 5) have a lesser chance to be selected than those similar-looking tiles in the input texture, due to their oddness in the tile population. As a result, the holes may not appear in the output texture at all as shown in Figures 10 and 11. When the direction of $\vec{t_1} + \vec{t_2}$ is not parallel with horizontal and vertical axes of the image, for simplicity in our experiments the shape of the maximum tile remains to be an upright rectangle containing the the minimum tile (Figure 6). An alternative, perhaps better, choice is to use the coordinate system of the texture defined by the minimum tile shape to define the shape of the maximum tile.

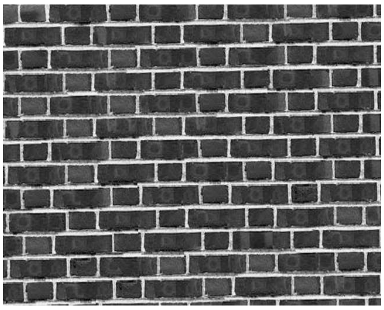
Figure 6: The sample tiles (rhombic shaped tiles are *minimum tiles* $\{t_i\}$ and rectangle shaped tiles are *maximum tiles* $\{T_i\}$) are shown, they are carved from the input brick texture. (a) and (b) show two different lattice positions.



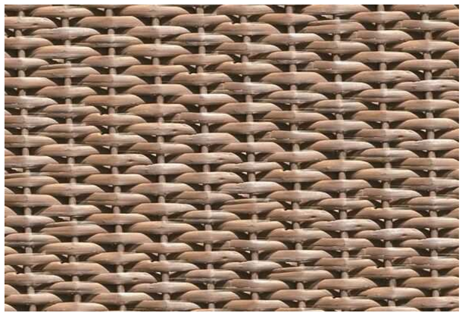
(a) a sample set of selected tiles



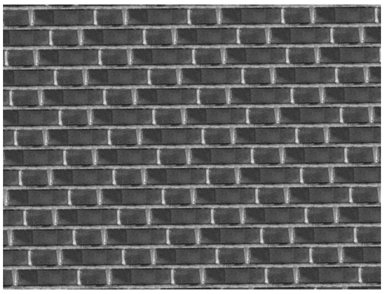
(b) another sample set of selected tiles centered on shifted lattice points



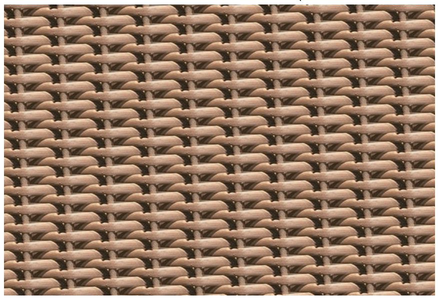
(a) synthesized texture (our method)



(b) synthesized texture (our method)



(c) naive direct tiling



(d) naive direct tiling

Figure 7: (a),(b): random sampling from tile sample sets using our texture synthesis method, which preserve both the near-regular nature of the texture **and** the variations across tiles. The symmetry group of both patterns is classified as *cmm* containing translation, rotation, reflection and glide-reflection symmetries [25]. (c),(d): direct tiling results. Though the regularity of the input texture is preserved, the synthesized texture does not reflect the intensity variations in the input texture.

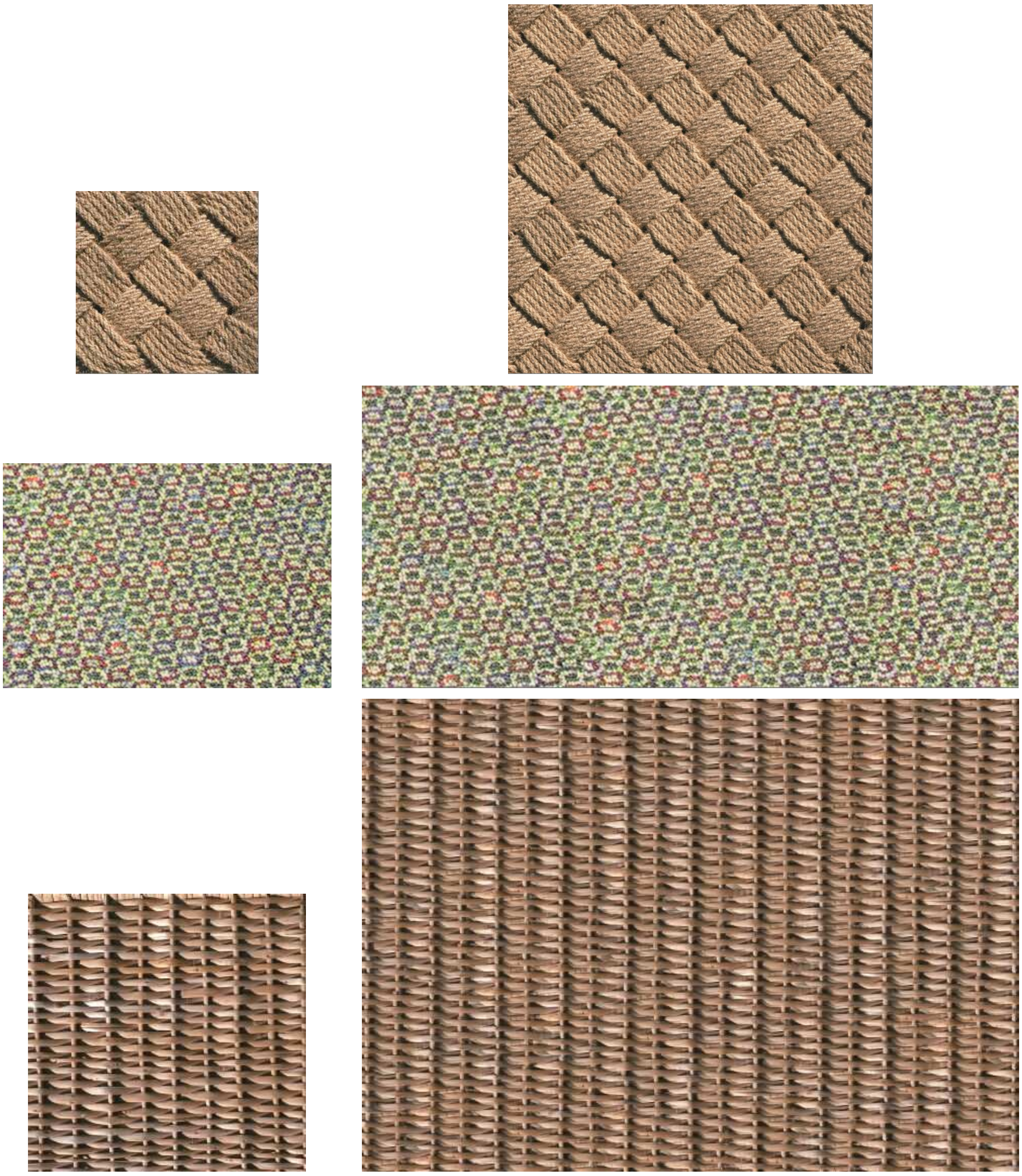
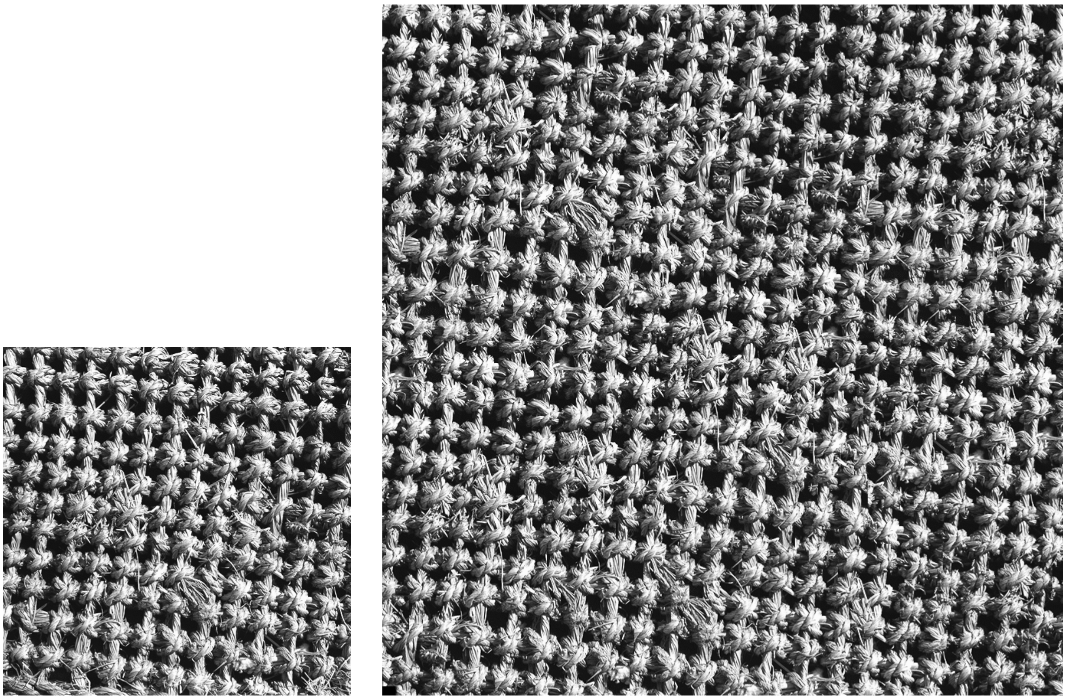


Figure 8: More examples on texture synthesis using our proposed approach.

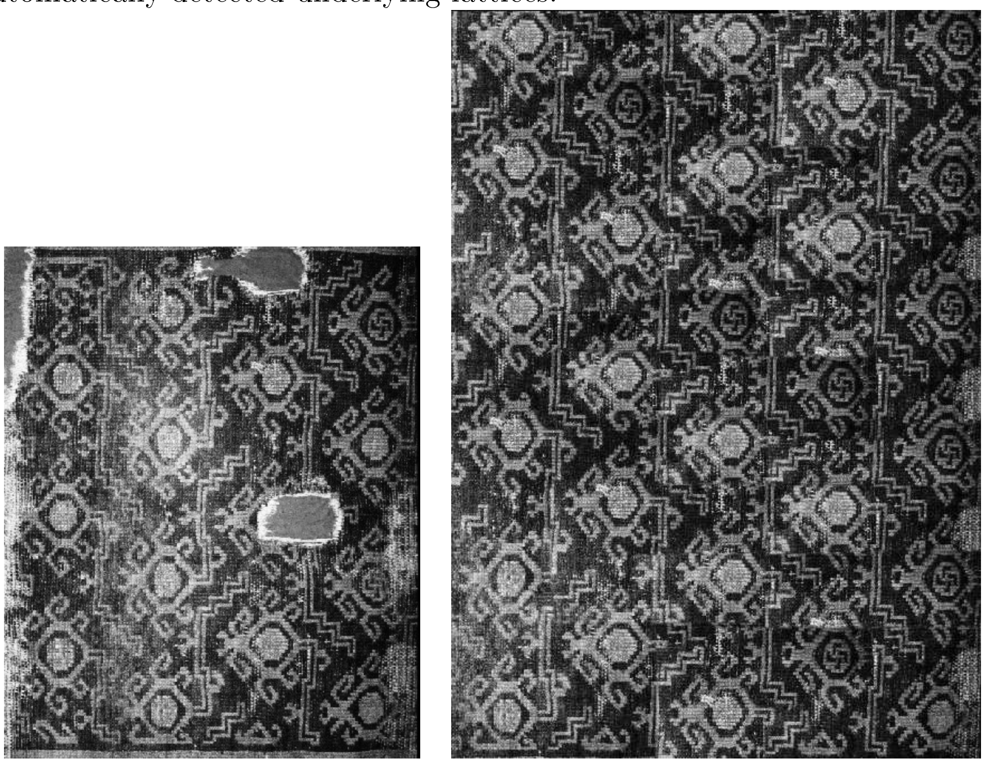
Figure 9: The synthesized result from one of the texture samples given in Figure 5, where imperfect, real-world near-regular patterns overlayed with automatically detected underlying lattices, using an algorithm developed in [27].



Original Texture

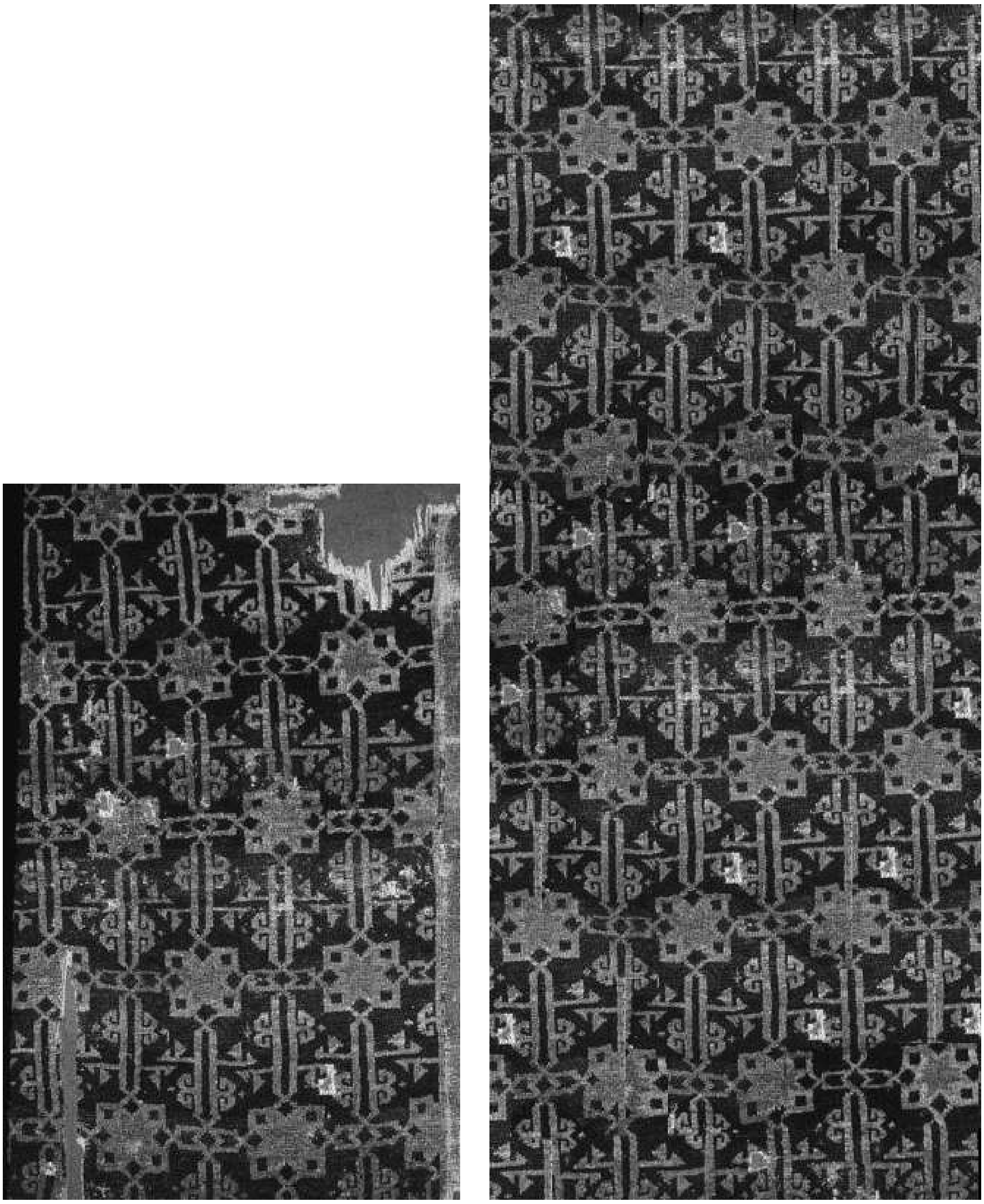
Synthesized Texture

Figure 10: The synthesized result from one of the texture samples given in Figure 5, where imperfect, real-world near-regular patterns overlayed with automatically detected underlying lattices.



Original Texture Synthesized Texture

Figure 11: The synthesized result from one of the texture samples given in Figure 5, where imperfect, real-world near-regular patterns overlayed with automatically detected underlying lattices.



Original Texture Synthesized Texture

4 Discussion

One obvious limitation of this work is its focus on near-regular texture alone. Nevertheless, several fundamental issues for texture understanding and synthesis seem to be related. Firstly, with respect to the very different properties of different textures (random versus regular) should we treat all textures uniformly? If not, how should we combine different methodologies together? Secondly, almost all the texture synthesis algorithms have to define a window, sometime called patch, and which we call a tile, to sample the original input texture. In this paper, we use the word window, patch and tile interchangeably. What are the basic variables involved in choosing a window, and what are their impacts? Thirdly, for near-regular textures, can we do better than what has been proposed here? What will happen if we go beyond translation symmetry and investigate the effect of rotation, reflection and glide-reflection symmetries? and how this may be related to the concept of texton for near-regular texture?

In the following, under the context of texture synthesis, we shall elaborate on each of these topics in more detail.

4.1 Regular Texture versus Random Texture

It is beyond the topic of this paper to investigate the definition of texture and whether regular or near-regular patterns should be considered as texture. However, near-regular texture does have its own special properties and relevant mathematical theories that can be, and actually has to be, taken into consideration in order to carry out the texture synthesis properly. On the other hand, it is also a reasonable concern that if regularity dominates the synthesized texture to the extreme of tiling, there will be no meaning in texture synthesis.

One fundamental principle in texture synthesis we are following is to be faithful to the input sample texture by respecting both its regularity and statistical randomness. One of the perils when dealing with near-regular texture is the temptation to use direct tiling (of a tile) to fill the whole 2D image. Though tiling is the central theme and appropriate means for many artistic and design tasks [35, 16], it is usually not suited for providing natural visual effects in the context of texture synthesis. The results from simple tiling are overly regular, usually more so than the original input sample (Figure 7).

The two perils of near-regular texture are:

1. random treatment: ignoring the special property of regularity, thus regularity (a global property) is no longer preserved (images (b) in Figures 2,3);

2. regular treatment: only recognizing that the texture is regular, thus ended up repeating a single tile (Figure 7).

Alternatively, one can avoid both of these two potential traps. There are many ways to combine the treatment of near-regular texture proposed here with existing local-neighborhood methods that are known to be particularly effective for random textures. One way is to build a texture regularity classifier F. Given a sample texture T, if $F(T) = 1$ exceeds a certain threshold, use our near-regular texture algorithm, otherwise resort to one of the local-neighborhood methods. People have already experimented with such classifiers. For example, [4] provides a score for a textured pattern that seems to be consistent with human perception. See Figure 12 from [5] for an example of regularity scores. Our lattice detection algorithm [25] or other future robust lattice extraction algorithms can also serve as a periodicity measure. In this manner, both the near-regular end and the random end of the textures will be well covered. The question then will become: how to treat those textures that are in the middle of the spectrum. We foresee a continuous spectrum from regular to random texture, but where to draw the line between regular, near-regular, near-random and random remains to be an open problem. It would be interesting to quantify how strong the regularity in a texture should be for the proposed texture synthesis method to be most effective.

4.2 Will the regularity in the input texture be preserved by increasing window size during texture synthesis?

Despite common belief, the answer to the above question is no. Figure 13 from [12] does show a trend towards regularity with the increase of the window size, however, it does not show that the regularity of the *input texture* can be reproduced with the increase of the window size. As a matter of fact, if one looks at the right-most synthesized result in Figure 13 carefully, it becomes obvious that the regularity produced in the synthesized texture with the larger window size is **not** the same kind of structural regularity presented in the input texture. Even though the smaller sized bricks occur in the input due to cut-offs at image borders, there is sufficient evidence to indicate that the input texture has bricks of only one size.

A local neighborhood-based approach is incapable of perceiving texture structure beyond the image borders, therefore it is not surprising that the synthesized texture reproduces what it can observe, and thus we see the mix of short bricks with the longer ones. Texture quilting [11] as a typical local

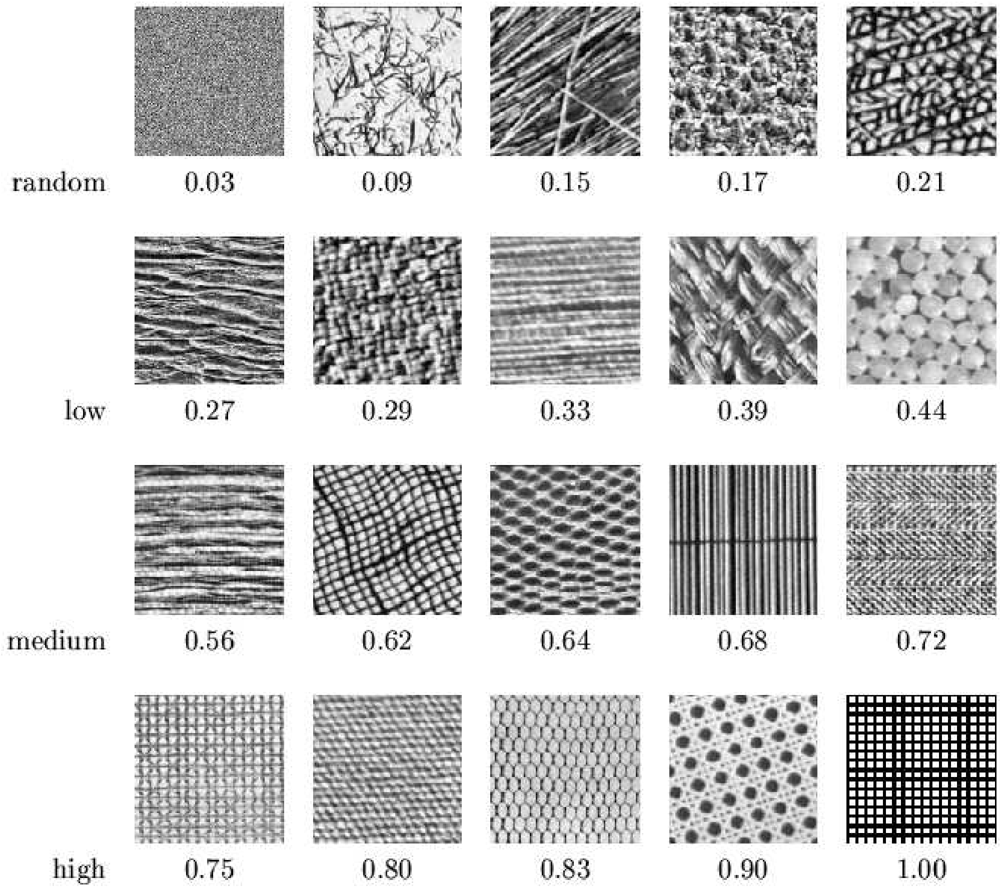


Figure 12: Image from [5] shows the regularity scores for various types of textures.

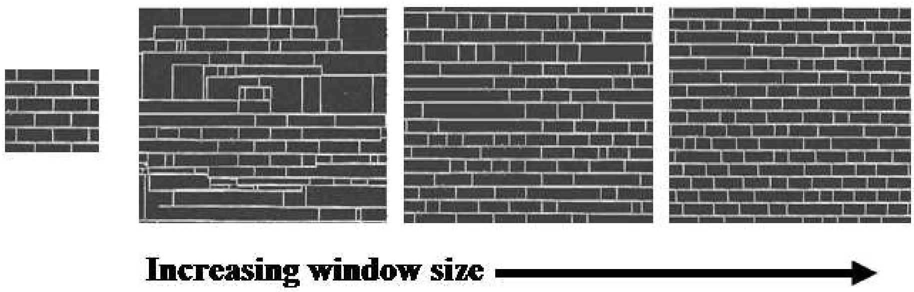


Figure 13: Image from [12] shows the synthesized texture becomes more regular with the increase of the window size. The question is: whether the regularity in the *input sample* will be reproduced by increasing the window size? The answer is: in general, **not really**.

approach allows variations around the overlapping boundary regions. When two patches are placed too close to each other due to an inappropriate window size, the algorithm can only maximize the similarity and smoothness locally to achieve a better looking local boundary, even though a global replacement is required to reproduce the similar spacing indicated in the original pattern. One can push the window size argument to the extreme: imagine using the whole input texture as the largest possible patch, even then the regularity of the input texture still will not be preserved, unless the cut-offs happen right at the matching line (e.g. the short bricks happen to have half-brick length in Figure 13). The input texture has to be a super tile, otherwise by putting its own copies together the regular pattern in the original texture (with a smaller generating tile) still can not be reproduced without discontinuity.

4.3 How to determine the sample window shape used for texture synthesis?

It is stated in [11] that:

> Determining precisely what are the patches for a given texture and how they are put together is still an open problem ... let us define the ... (patch) ... to be a square block of user-specified size ...

For lack of a better choice of window shapes, an upright square window is a common choice for many texture synthesis algorithms [11, 36, 37, 23].

The reason that local texture synthesis algorithms work on certain near-regular textures (patterns of dots or knots, for example) is due to a judicious choice of the window size and shape that happens to match the tile shape and orientation of the input sample. In the case of dots and knots, it is an upright square; and in the case of soup cans and rows of windows, it is an upright rectangle such that a proper square can serve as a super tile (Figure 4 in [11]). Conversely, an improper choice of window size and shape usually causes failure in faithful texture synthesis (e.g. images (b) in Figures 2,3 from [11]).

A key factor in reproducing regularity is to recognize, simultaneously, the shape, orientation and size of a basic tile of the input near-regular texture. This is the attempt we make in our texture synthesis method (Figure 5). One advantage of our approach is that the tile shape (not necessarily a square), orientation (not necessarily upright), and size are determined up front, explicitly, and customized to each input near-regular input texture pattern (Figure 5). Even though the input texture's color and intensity may vary randomly, recognizing the underlying structural regularity provides a skeleton for the appropriate texture displacement while allowing color and intensity variations. We have demonstrated the feasibility of this approach in Section 3.2.

4.4 Go Beyond Translational Symmetry and how it is related to the Concept of Texton

When one really understands the making of a periodic pattern and its generating regions [32], modifications can be made to direct tiling such that more natural appearance can be achieved. In particular, we have only used translational symmetry in this paper, rotation, reflection and glide reflection symmetries can also be used to generate patterns from much smaller tiles (Figure 14). This means that a much larger sample set of observed statistical variations can be obtained in a principled and controlled manner.

In many texture related papers (e.g. [39]) the concept of a "texton" has been suggested. Texton is referring to the atomic element in a texture. There

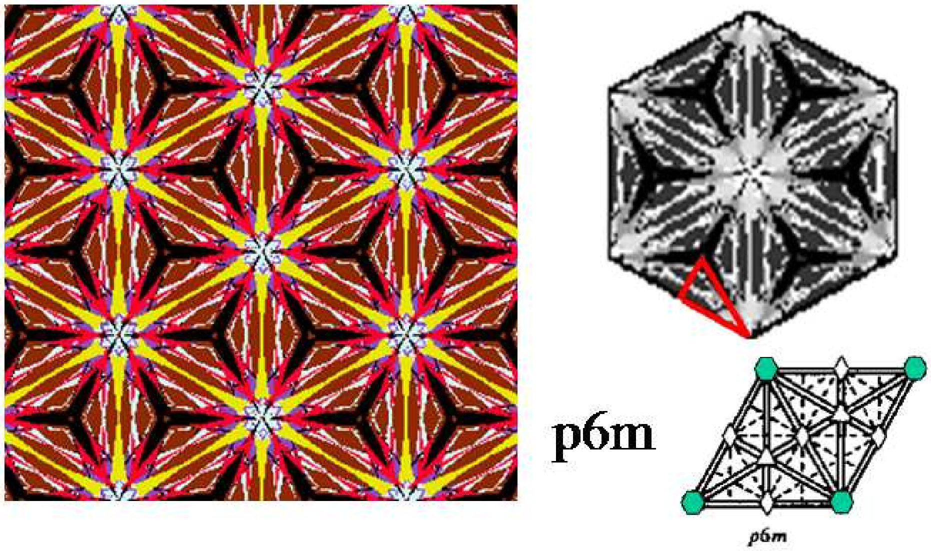


Figure 14: For this pattern (p6m, one of the 17 wallpaper groups patterns), only the triangle region is needed to recover the whole image through rotation, reflection and translations. Thus much smaller tile sizes and more sample numbers can be used for texture synthesis of better quality.

exists an interesting interplay between what is a texton and which group of transformations that one is considering. In the case at hand (Figure 14), if we only consider the translation subgroup of the symmetry group of the texture pattern, the texton would be the parallelogram for hexagonal shape shown in the lower right corner of Figure 4. If we consider the whole symmetry group of the texture pattern, the corresponding texton then becomes the small triangle indicated in Figure 14. Near-regular textures provide a more structured environment for exploring the elusive concept of multi-layered textons.

5 Conclusion and Research Directions

In this paper, we provide a new method for near-regular texture synthesis. Our method differs from most local-neighborhood approaches to texture synthesis in that it first does a texture structure analysis by identifying the specific tile shape of the given texture. Our approach also separates the treatment of spatial layout regularity (tiles) from the intensity/color variations (the content of a tile). A special treatment for near-regular texture in texture synthesis has been a missing piece in the texture synthesis puzzle.

We point out that it is actually a misconception that the regularity in the input sample will be reproduced when the window size is large enough. It should be realized by now that the regularity preservation problem can not be solved by adjusting window size alone.

We are investigating the use of a richer set of symmetries residing in near regular texture beyond translation. Our long term goal is to model a continuous spectrum from regular to near-regular to chaotic patterns, and to study texture variations along different dimensions of symmetry [24].

6 Acknowledgement

This research is partially funded by an NSF research grant # IIS-0099597. We highly appreciate the constructive comments given by two anonymous reviewers. Dr. Liu thanks Professor W. T. Freeman, Dr. A.A. Efros and Dr. R.T. Collins for helpful discussions.

References

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

[2] L. Bieberbach. Über die Bewegungsgruppen der n-dimensional en Euklidischen Räume mit einem endlichen Fundamental bereich. *Göttinger Nachrichten*, pages 75–84, 1910.

[3] D.A. Chambers, editor. *DNA: the double helix: Perspective and Prospective at Forty Years.* New York Academy of Sciences, New York, 1995.

[4] D. Chetverikov. Pattern regularity as a visual key. *Image and Vision Computing*, 18:975–986, 2000.

[5] D. Chetverikov. *Fundamental Structural Properties of Textures.* PhD thesis, MTA SZTAKI, Budapest, 2002.

[6] R.W. Conners and C.A. Harlow. Toward a structural textural analyzer based on statistical methods. *CGIP*, 12(3):224–256, March 1980.

[7] H.S.M. Coxeter. *Introduction to Geometry.* Wiley, New York, second edition, 1980.

[8] H.S.M. Coxeter and W.O.J. Moser. *Generators and Relations for Discrete Groups.* Springer-Verlag, New York, fourth edition, 1980.

[9] G.R. Cross and A.K. Jain. Markov random field texture models. *IEEE Trans. PAMI*, 5:25–39, 1983.

[10] J.S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture image. In *SIGGRAPH Proceedings*, pages 361–368, 1997.

[11] A.A. Efros and W.T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, pages 35–42, 2001.

[12] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *International Conference on Computer Vision*, 1999.

[13] R. Enrich and J.P. Foith. A view of texture topology and texture description. *Computer Graphics Image Processing*, 8:174–202, 1978.

[14] E.S. Fedorov. The elements of the study of figures. [Russian] (2) 21. In *Zapiski Imperatorskogo S. Peterburgskogo Mineralogichesgo Obshchestva [Proc. S. Peterb. Mineral. Soc.]*, pages 1–289, 1885.

[15] R.P. Feynman. *Six Not-so-easy pieses: Einstein's relativity, Symmetry, and Space-Time.* Perseus Press, Cambridge, Mass, 1998.

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

[17] István Hargittai and Magdolna Hargittai. *In Our Own Image: Personal Symmetry in Discovery.* Kluwer Academic Publishers, 2000.

[18] N.F.M. Henry and K. Lonsdale, editors. *International Tables for X-ray Crystallography, Volume 1, Symmetry Groups.* The Kynoch Press, England, 1969. The International Union of Crystallography.

[19] A. Hertzmann, C.E. Jacobs, N. Oliver, B. Curless, and D.H. Salesin. Image analogies. *SIGGRAPH*, 2001.

[20] T.I. Hsu and R. Wilson. A two-component model of texture for analysis and synthesis. *IEEE Trans. on Image Processing*, 7(10):1466–1476, October 1998.

[21] C.S. Kaplan and D.H. Salesin. Escherization. In *the 27th International Conference on Computer Graphics and Interactive Techniques*, July 2000.

[22] M. Leyton. *Symmetry, Causality, Mind.* The MIT Press, Cambridge,Massachusetts, 1992.

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

[24] Y. Liu. Computational Symmetry. In I. Hargittai and T.C. Laurent, editors, *Symmetry 2000*, volume 80, chapter 21, pages 231–245. Wenner-Gren International Series, Portland, London, ISBN I 85578 149 2, 2001.

[25] Y. Liu and R. T. Collins. A Computational Model for Repeated Pattern Perception using Frieze and Wallpaper Groups. In *Computer Vision and Pattern Recognition Conference (CVPR'00)*, pages 537–544, Hilton Head,SC, June 2000. IEEE Computer Society Press. (http://www.ri.cmu.edu/pubs/pub_3302.html).

[26] Y. Liu and R. T. Collins. Skewed Symmetry Groups. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, pages 872,879, Kauai,HI, December 2001. IEEE Computer Society Press. (http://www.ri.cmu.edu/pubs/pub_3815.html).

[27] Y. Liu, R.T. Collins, and Y. Tsin. A computational model for periodic pattern perception based on frieze and wallpaper groups. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, to appear, 2003.

[28] S.Y. Lu and K.S. Fu. A syntactic approach to texture analysis. *Computer Graphics Image Processing*, 7:303–330, November 1978.

[29] W. Miller Jr. *Symmetry Groups and Their Applications.* Academic Press, New York, 1972.

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

[31] A.R. Rao and G.L. Lohse. Identifying high level features of texture perception. *CVGIP: Image Processing*, 55:218–233, 1993.

[32] D. Schattschneider. The plane symmetry groups: their recognition and notation. *American Mathematical Monthly*, 85:439–450, 1978.

[33] M. Senechal. *Quasicrystals and Geometry.* Cambridge Universy Press, 1995.

[34] Y. Tsin, Y. Liu, and V. Ramesh. Texture replacement in real images. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01). (http://www-2.cs.cmu.edu/ ytsin/research/texture_replacement/index.html)*, Kauai, December 2001. IEEE Computer Society Press.

[35] D.K. Washburn and D.W. Crowe. *Symmetries of Culture: Theory and Practice of Plane Pattern Analysis.* University of Washington Press, 1991.

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

[37] Y.Q. Xu, S.C. Zhu, B.N. Guo, and H.Y. Shum. Asymptotically admissible texture synthesis. In *International Workshop on Statistical and Computational Theories of Vision*, July 2001.

[38] Anthony Zee. *Fearful Symmetry.* Princeton University Press, 1999.

[39] S.C. Zhu, C.E. Guo, Y.N. Wu, and Y.Z. Wang. What are textons. In *Proceeding of European Conference on Computer Vision*, Copenhagen,Denmark, June 2002.

[40] S.C. Zhu, X. Liu, and Y. Wu. Exploring texture ensembles by efficitn markov chain monte carlo. *IEEE Transaction on PAMI*, 22(6), 2000.

[41] S.C. Zhu, Y. Wu, and D.B. Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9:1627–1660, 1997.

[42] S.W. Zucker. Toward a model of texture. *CGIP*, 5(2):190–202, June 1976.