An Alternative for Wang Tiles: Colored Edges versus Colored Corners

ARES LAGAE and PHILIP DUTRÉ
Katholieke Universiteit Leuven

In this article we revisit the concept of Wang tiles and introduce corner tiles, square tiles with colored corners. During past years, Wang tiles have become a valuable tool in computer graphics. Important applications of Wang tiles include texture synthesis, tile-based texture mapping, and generating Poisson disk distributions. Through their colored edges, Wang tiles enforce continuity with their direct neighbors. However, Wang tiles do not directly constrain their diagonal neighbors. This leads to continuity problems near tile corners, a problem commonly known as the corner problem. Corner tiles, on the other hand, do impose restrictions on their diagonal neighbors, and thus are not subject to the corner problem. In this article we show that previous applications of Wang tiles can also be done using corner tiles, but that corner tiles have distinct advantages for each of these applications. Compared to Wang tiles, corner tiles are easier to tile, textures synthesized with corner tiles contain more samples from the original texture, corner tiles reduce the required texture memory by a factor of two for tile-based texture mapping, and Poisson disk distributions generated with corner tiles have better spectral properties. Corner tiles result in cleaner, simpler, and more efficient applications.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Color, shading, shadowing, and texture*

General Terms: Algorithms

Additional Key Words and Phrases: Wang tiles, corner tiles, tiling, texture synthesis, tile-based texture mapping, Poisson disk distributions

1. INTRODUCTION

Computer graphics is often concerned with the synthesis of complex signals, such as textures and Poisson disk distributions. Wang tiles are an important tool to facilitate the generation of such signals. Wang tiles are square tiles with colored edges, placed randomly next to each other such that adjoining edges have matching colors. Instead of synthesizing a complex signal directly, the signal is constructed over a set of Wang tiles, consistent with the continuity constraints imposed by the colored edges. This is usually more difficult than synthesizing the signal directly, but once a tile set is constructed, arbitrary large nonperiodic quantities of this signal can be generated very efficiently by stochastically tiling the Wang tiles.

The colored edges, however, do not guarantee continuity of the signal near tile corners. Any two Wang tiles can be put diagonally to each other by adding two suitable tiles to complete the tiling.

A. Lagae is funded as a Research Assistant by the Fund of Scientific Research, Flanders, Belgium (Aspirant F.W.O., Vlaanderen).
Authors' addresses: A. Lagae, Ph. Dutré, Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, Heverlee; email: ares.lagae@cs.kuleuven.be.

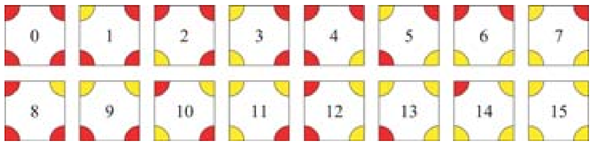

Fig. 1. The complete set of corner tiles over two colors.

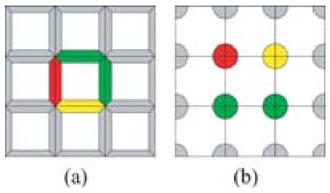

Fig. 2. The corner problem; (a) Wang tiles only enforce continuity with their four direct neighbors and do not constrain their diagonal neighbors; (b) corner tiles enforce continuity with all their neighbors. Note that there is also no analogous mid-edge problem.

This problem, called the corner problem, complicates construction methods and causes unwanted artifacts in the generated signals.

In this article we propose corner tiles, that is, square tiles with colored corners. Figure 1 shows a complete set of corner tiles over two colors. Corner tiles are similar to Wang tiles, but their colored corners ensure continuity of the signal over both tile edges and corners, thus avoiding the corner problem. This is illustrated in Figure 2. We revisit the most important applications of Wang tiles. We show that corner tiles have substantial advantages for each of these applications and that corner tiles result in cleaner, simpler, and more efficient applications.

1.1 Contributions

The main contributions of this work are the introduction of corner tiles, tiling algorithms for corner tiles, and an improvement of the important applications of Wang tiles. We show that in comparison with Wang tiles, corner tiles have the following advantages:

—Tiling algorithms for corner tiles are more efficient;

—textures synthesized with corner tiles contain more unique texture samples from the original texture;

—corner tiles reduce the required texture memory by a factor of two and improve the speed of tile-based texture mapping; and

—Poisson disk distributions generated with corner tiles have better spectral properties, even when smaller tile sets are used.

We hope that future tile-based applications will consider corner tiles as an alternative to Wang tiles.

1.2 Overview

The remainder of this article is structured as follows. In Section 2 we discuss related work. Section 3 introduces corner tiles. In Section 4 we present tiling algorithms for corner tiles. The next three sections discuss three applications in which the use of corner tiles proves beneficial: texture synthesis (Section 5), tile-based texture mapping (Section 6), and the generation of Poisson disk distributions (Section 7). We conclude and give directions for future work in Section 8.

2. BACKGROUND AND RELATED WORK

In this section we briefly sketch the background of Wang tiles, discuss previous applications of Wang tiles in the field of computer graphics, and review related work in Poisson disk distributions and texture synthesis.

2.1 Wang Tiles

Wang tiles originated in the field of discrete mathematics. They were first proposed by Hao Wang in 1961 [Wang1961, 1965]. In his article, Wang conjectured that aperiodic tile sets, tile sets that do not admit periodic tilings, did not exist. Berger refuted this conjecture in 1966, and constructed the first aperiodic set of Wang tiles, counting 20,426 tiles [Berger 1966]. This number was reduced repeatedly, often by well-known scientists such as Donald Knuth [1968]. The smallest aperiodic set of Wang tiles consists of 13 tiles over 5 colors [Culik 1996]. For more information about aperiodic tilings, we refer the reader to Grünbaum and Shepard [1986] and Glassner [1999, Chapter 12].

2.2 Applications of Wang Tiles in Computer Graphics

Wang tiles were introduced in the field of computer graphics by Stam [1997], who filled the tiles with procedurally generated textures. Shade et al. [2000] and Hiller et al. [2001] used Wang tiles to generate Poisson disk distributions. The latter approach was later adopted by Cohen and collaborators, in an article that popularized Wang tiles in the field of computer graphics [Cohen et al. 2003]. The same article introduced a method for texture synthesis using Wang tiles. Tile-based texture mapping on graphics hardware was first proposed by Wei [2004]. Fu and Leung [2005] recently extended texture tiling to arbitrary topological surfaces. Lagae and Dutré [2005a] presented an improved method for generating Poisson disk distributions and a procedural object distribution function based on Wang tiles.

2.3 Poisson Disk Distributions

Poisson disk distributions were introduced in the field of computer graphics to solve the aliasing problem. Dippé and Wold [1985], Cook [1986], and Mitchell [1987] introduced nonuniform sampling to turn structured aliasing artifacts into featureless noise. The Poisson disk distribution was identified as one of the best sampling patterns. Their work was based on studies of photoreceptor sampling in the retina [Yellot 1983]. Besides sampling, Poisson disk distributions are also used for geometry instancing [Deussen et al. 1998], object distribution for illustration [Secord et al. 2002], and procedural texturing [Lagae and Dutré 2005a].

Poisson disk distributions are traditionally generated using an expensive dart throwing algorithm [Cook 1986]. Fast methods that generate approximate Poisson disk distributions were proposed by various authors [Dippé and Wold 1985; Mitchell 1991]. McCool and Fiumé [1992] presented an approach that generates a hierarchical Poisson disk distribution and introduced Lloyd's relaxation scheme [1982] to optimize the generated distribution.

In 1985, Dippé and Wold suggested replicating a precomputed tile with Poisson disk distributed points across the plane. The first tile-based approach for generating Poisson disk distributions is an extension of dart throwing to Wang tiles, and is due to Shade et al. [2000]. Hiller et al. [2001] extended Lloyd's [1982] relaxation scheme to Wang tiles. This method was later adopted by Cohen et al. [2003]. Ostromoukhov et al. [2004] proposed a method to generate point distributions with blue noise properties over a given density based on Penrose tiles and Lloyd's [1982] relaxation scheme. Recently, Lagae and Dutré [2005a] presented an improved tile-based method for generating Poisson disk distributions, and studied the requirements for generating tiled distributions with good spectral properties [Lagae and Dutré 2005b].

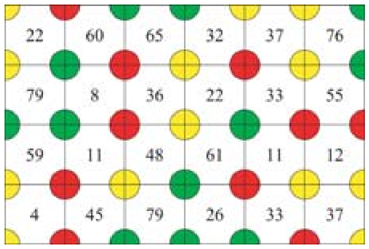

Fig. 3. A 4 × 6 tiling generated from the complete set of corner tiles over three colors.

2.4 Texture Synthesis

Texture synthesis has become a popular area of research within computer graphics, and a complete survey of related work is beyond the scope of this article. For an overview, we refer to Liu et al. [2004] and Kwatra et al. [2005]. Most techniques for texture synthesis are region growing methods, such as pixel-based [DeBonet 1997; Efros and Leung 1999; Wei and Levoy 2000] and patch-based texture synthesis [Efros and Freeman 2001; Liang et al. 2001; Cohen et al. 2003; Kwatra et al. 2003] or global methods, such as the work of Heeger and Bergen [1995] and Kwatra et al. [2005]. The texture synthesis technique presented in this article can be classified as a patch-based method.

2.5 Tiles with Colored Corners

Cohen et al. [2003] first identified the corner problem. They superimpose corner markings on a Wang tile set in an attempt to solve the problem. Although this allows the synthesis of textures with different densities, the corner problem remains: For a given corner marking, any two tiles can be put diagonally next to each other. They do not make the observation that the edge colors should be dropped altogether to adequately solve the corner problem.

Neyret and Cani [1999] use triangular tiles with edge and corner colors to generate pattern-based textures over a triangle mesh, in the spirit of Stam [1997].

Tiles with colored corners have, to our knowledge, not been used previously in computer graphics (or other domains) in the same way as in this article, except by Ng et al. [2005]. They presented a technique for assembling a set of tiles similar to corner tiles from an input texture to synthesize larger textures. We adopt their method in this work.

3. CORNER TILES

Corner tiles are defined analogous to Wang tiles. Corner tiles are unit square tiles with colored corners, and similarly to Wang tiles, have a fixed orientation. A corner tile set is a finite set of corner tiles, and the set is complete if it contains one corner tile for each possible combination of colors. Thus, a complete set of corner tiles over C colors consists of C^4 tiles. Figure 1 shows all 16 tiles of the complete corner tile set over two colors. A tiling is constructed by placing the tiles next to each other such that adjoining corners have matching colors. Each tile in the tile set can be used arbitrarily many times. Figure 3 shows a tiling using tiles from the complete set of corner tiles over three colors.

In this article we use the following conventions. Analogous to Wang tiles, we name the corners of corner tiles after the compass headings northeast (NE), southeast (SE), southwest (SW), and northwest (NW). The C colors are represented by the integers $0, 1, \ldots, C-1$. The illustrations in this article use the colors red, yellow, green, and cyan for, respectively, 0, 1, 2, and 3.

Most applications of corner tiles require an enumeration of all tiles in a tile set. We found the following scheme to be convenient. Corner tiles are uniquely determined by their corner colors c_{NE}, c_{SE}, c_{SW}, and

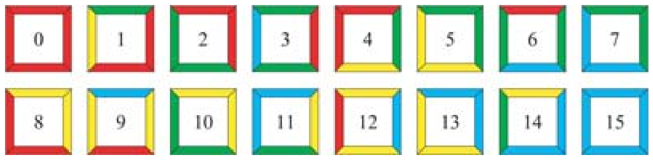

Fig. 4. A Wang tile set of 16 tiles over four colors equivalent to the complete set of corner tiles over two colors shown in Figure 1.

c_{NW}. Corner tiles can thus be represented as the 4-digit base-C numbers $c_{NE}\,c_{SE}\,c_{SW}\,c_{NW}$, or decimal integers $0, 1, \ldots, C^4 - 1$. A base conversion switches between the corner colors and tile index. For example, the tile index of the tile with corner colors c_{NE}, c_{SE}, c_{SW}, and c_{NW} is given by

$$((c_{NE}C + c_{SE})C + c_{SW})C + c_{NW}. \tag{1}$$

We use a similar scheme to enumerate Wang tiles.

Corner tiles are not just an ad hoc extension of Wang tiles. The tiles are sound also from a discrete mathematical point of view. For example, we have recently shown that it is possible to construct aperiodic sets of corner tiles [Lagae et al. 2006]. The smallest aperiodic set of corner tiles consists of 44 tiles over six colors. Corner tiles are also closely related to Wang tiles. Every corner tile set can be transformed into an equivalent Wang tile set by encoding any combination of two corner colors into an edge color (squaring the number of colors). Figure 4 shows an example. The opposite, however, in generally not possible.

4. TILING ALGORITHMS FOR CORNER TILES

Applications in computer graphics require random or stochastic tilings, such as that shown in Figure 3. Stochastic tilings are inherently nonperiodic. For practical applications, the stronger mathematical guarantee of (provable) aperiodicity is not needed. There are two kinds of stochastic tiling algorithms: scanline and direct.

4.1 Scanline Stochastic Tiling

A scanline stochastic tiling algorithm for Wang tiles was introduced by Cohen et al. [2003]. The tiles are placed in scanline order, from west to east, and from north to south. The Wang tile set is constructed to contain at least two tiles for each possible combination of north and west colors. Each time a tile is chosen from the tile set to fill out the next position in the scanline, the selection is made at random. Extending this algorithm to corner tiles is straightforward: A stochastic tiling is always possible if a complete corner tile set over (at least) two colors is used because the tile set contains at least two tiles for each combination of northeast, southwest, and northwest colors.

4.2 Direct Stochastic Tiling

The scanline stochastic tiling algorithm must store the complete tiling. However, many applications require the evaluation of a tiling locally, without explicitly constructing and storing the tiling up to this point. To solve this problem, direct stochastic tiling algorithms for Wang tiles have been proposed [Wei 2004; Lagae and Dutré 2005a].

We present the following algorithm for corner tiles. Without loss of generality, assume that the tiles are placed with their corners on the integer lattice points, and that the coordinates of a tile are the coordinates of its SW corner. Similar to direct stochastic tiling algorithms for Wang tiles, the algorithm for corner tiles is based on a hash function $h(x, y)$ that associates a random color with each integer lattice point. The corner colors of the tile at coordinates (x, y) are given by $h(x+1, y+1)$, $h(x+1, y)$,

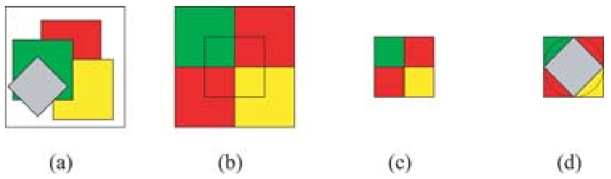

Fig. 5. Tile construction for texture synthesis; (a) for each color, a square patch is chosen in the input texture (the red, green, and yellow patch); (b) the patches are assembled according to corner colors; (c) the tile is cut out; and (d) the seam is covered with a new irregular patch from the input texture (the gray patch).

$h(x, y)$, and $h(x, y+1)$. The tile index is obtained with a base conversion, as explained in the previous section. We commonly use a hash function based on a permutation table [Perlin 2002; Ebert et al. 2002] because such hash functions are both easy to implement and efficient, but in practice, any hash function can be used. If P is a zero-based table containing a random permutation of the integers $0, 1, \ldots, N-1$, then the hash function is defined as

$$h(x, y) = P[(P[x\%N] + y)\%N]\%C, \tag{2}$$

in which % is the modulo division and N is the permutation table size. Moderate table sizes (256 or less) are commonly used. Note that with this particular choice of hash function, the tiling will have a period of (N, N). This is not necessarily a disadvantage, as it allows tilings over cylindric and toroidal topologies.

Figure 3 shows a tiling obtained with the direct stochastic tiling algorithm, and the accompanying video[1] shows tilings of several thousands of tiles for tile sets over two, three and four colors.

4.3 Discussion

The direct stochastic tiling algorithm for corner tiles is cleaner and more efficient than similar algorithms for Wang tiles. One of the latter, presented by Lagae and Dutré [2005a], is also based on random colors generated at integer lattice points. However, the algorithm needs to compute edge colors from these corner colors. The algorithm for corner tiles uses the corner colors right away, with no additional computation, and is therefore faster. At first sight this difference might seem small, but it matters, for example, in GPU implementations and efficient implementations of texture basis functions.

5. TEXTURE SYNTHESIS

A technique for assembling a set of corner tiles from an input texture so as to synthesize larger textures was recently introduced by Ng et al. [2005]. We adopt their method in this article, and adjust it to our corner tile sets and tiling algorithms.

5.1 Tile Construction

For each corner color, a square patch is chosen at random from the input texture. Each tile is constructed by combining the four patches corresponding to the corner colors. This leaves a cross-shaped seam that is covered with a new diamond-shaped irregular patch from the input texture. This patch is optimized using the graph cut technique [Kwatra et al. 2003], and is restricted to lie in the circle inscribed in the tile. The process of tile construction is depicted in Figure 5. Figures 7 and 8 show several texture synthesis results. The accompanying video shows tilings of several thousands of tiles. Note that in contrast to [Ng et al. 2005], we generate complete sets of corner tiles.

[1]http://www.cs.kuleuven.be/∼ares/.

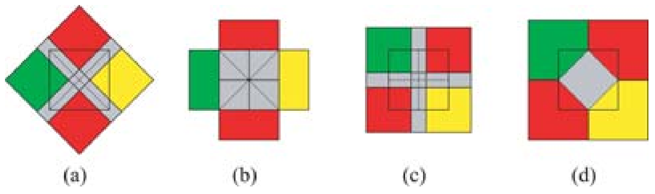

Fig. 6. Patch combination strategies; (a) the method of Cohen et al. [2003]; (b) a variant introduced by Burke; (c) a straightforward extension of (a) to corner tiles; and (d) the method used in this article. This last method is the only one that introduces new texture samples in each tile.

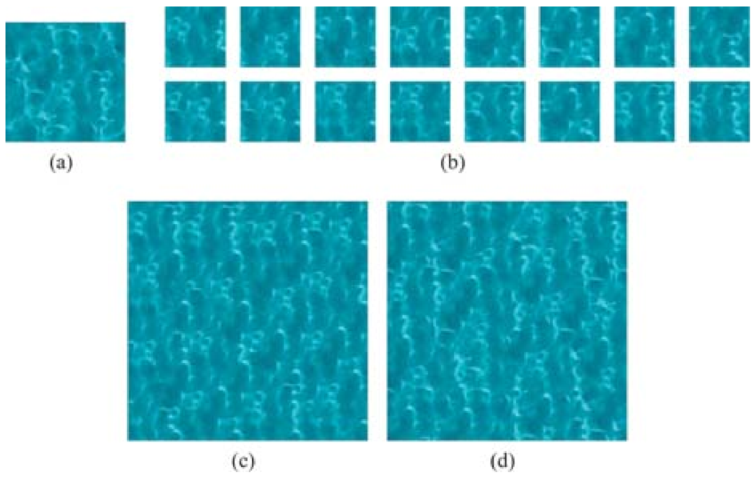

Fig. 7. Texture synthesis with corner tiles; (a) the input texture; (b) a complete set of corner tiles over two colors, constructed from the input texture (the tiles are in the same order as in Figure 1); (c) a new texture synthesized with this set; and (d) a texture synthesized with a complete set of corner tiles over three colors.

5.2 Discussion

This approach is simple and works well, and the quality of results is similar to those of other patch-based techniques. Textures synthesized with corner tiles are usually more similar to the input texture than those synthesized with Wang tiles. This is because the center of each corner tile is covered with a new irregular patch from the input texture. Therefore, each corner tile contains unique texture samples from the input texture. As illustrated in Figure 6, this is not the case for Wang tiles.

Unwanted artifacts in the synthesized textures are typically located where patches meet. This is at the corners for Wang tiles and in the middle of edges for corner tiles. In this respect, Wang and corner tiles are similar (see Ng et al. [2005] for a more detailed comparison).

An exciting advantage of tile-based texture synthesis is that its process is broken up into a preprocess in which the tiles are constructed, and a real-time part in which a new texture is synthesized. Once a set of tiles is constructed, an arbitrary large amount of textures can be synthesized very efficiently by generating a tiling. We explore this topic further in the next section.

6. TILE-BASED TEXTURE MAPPING

One of the advantages of tile-based texture synthesis is that a new texture can be synthesized in real time. In this section, we discuss the advantages of corner tiles over Wang tiles for tile-based texture mapping on graphics hardware, as introduced by Wei [2004].

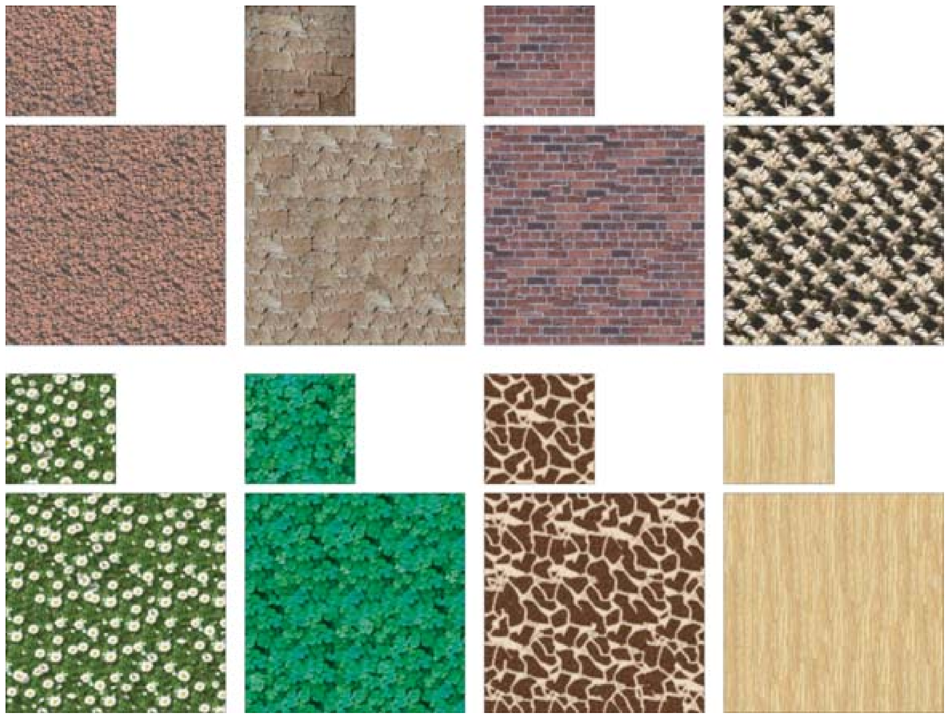

Fig. 8. Texture synthesis results. The textures are synthesized by tiling 4 × 4 tiles from a complete corner tile set over two or three colors.

6.1 Tile-Based Texture Mapping on the GPU

Tile-based texture mapping takes as input a complete set of tiles filled with a texture. A fragment program running on the GPU uses a direct stochastic tiling algorithm to generate an arbitrarily large and nonperiodic texture from the tiles in real time.

Because texture units are a scarce resource on graphics hardware, all the tiles of the tile set are packed into a single square texture, called a tile packing. To minimize texture memory usage, each tile in the tile set should appear exactly once in the tile packing, and in order to avoid the discontinuity artifacts introduced by texture filtering, the tile packing should also be a valid tiling. This is because texture sampling uses texels from adjacent tiles (see Wei [2004] for more details).

In general, tile-based texture mapping algorithms only use complete tile sets. This is because a complete tile set over C colors counts C^4 tiles, which can easily be packed into a square texture using a $C^2 \times C^2$ tiling.

6.2 The Tile Packing Problem

Formally, the tile packing problem consists of arranging a complete set of C^4 tiles in a $C^2 \times C^2$ toroidal configuration such that adjoining edges or corners have matching colors. A tile packing is an essential ingredient of the tile-based texture mapping algorithm. If a tile packing is not used, texture filtering will introduce discontinuity artifacts [Wei 2004].

In a single dimension, Wang tiles (as well as corner tiles) can be seen as dominoes. The equivalent problem is to arrange all domino tiles into a single circular train. This problem is well-studied in the field of recreational mathematics. A solution based on graph theory is given in the classic work of

Ball [1926]. Wei [2004] observed that a solution for the Wang tile packing problem is given by the outer product of two 1D tile packings.

Although tiles with colored edges and problems similar to the Wang tile packing problem were studied before in the field of recreational mathematics [MacMahon 1921], corner tiles and the corner tile packing problem have not been examined. The method for constructing a Wang tile packing also does not seem to extend to corner tiles because the latter are more restrictive than Wang tiles. It was not clear whether the corner tile packing problem even had a solution. We therefore decided to tackle the problem using combinatorial search methods. A simple exhaustive search or generate-and-test method is not practical: For C colors, the tiles can be arranged in $C^4!$ ways, which equals approximately 2.09×10^{13} for $C = 2$ and 5.80×10^{120} for $C = 3$. Instead, we use a backtracking method that places one tile at a time until a dead-end is reached, at which point the previous steps are retraced. With the backtracking algorithm, we are able to compute Wang and corner tile packings for two, three, and four colors. A solution for C colors can often be found more quickly by starting from a solution of $C - 1$ colors. This way, a recursive tile packing is obtained. A recursive corner tile packing for four colors is shown in Figure 9. Some of these tile packings took almost one year of CPU time to compute on a cluster with 400 2.4 GHz CPUs. More solutions and a description of the implementation of our parallel backtracking algorithm can be found in Lagae and Dutré [2006b].

6.3 Implementation and Results

The tile-based texture mapping algorithm runs as a fragment program on the GPU. The tile coordinates of an incoming fragment with texture coordinates (s, t) are given by $(\lfloor s \rfloor, \lfloor t \rfloor)$. The direct stochastic tiling algorithm is used to determine the tile index. The permutation table and tile packing can be implemented as a 1D texture or constant array in the fragment program. The texture coordinates within the tile are given by $(s - \lfloor s \rfloor, t - \lfloor t \rfloor)$. The tile index and packing are used to compute texture coordinates for the tile packing texture. For more implementation details, we refer to Wei [2004] and Lefebvre and Neyret [2003]. Our interactive application runs at several hundred frames per second on a NVidia GeForce 7800 GTX graphics card. Figure 10 shows several results.

6.4 Discussion

To avoid the corner problem, the approach for tile-based texture mapping by Wei [2004] requires a second Wang tile packing that contains all possible corner configurations of the Wang tile set. This additional texture is used for texture lookups close to tile corners. Because corner tiles are not subject to the corner problem, only the texture that contains the tile packing is needed. Compared to the original method of Wei [2004], our algorithm thus reduces the required texture memory by a factor of two and saves one texture unit. This is an important advantage, as reducing texture memory usage is the main goal of tile-based texture mapping. Our algorithm also runs faster because the tiling algorithm for corner tiles is simpler and more efficient than equivalent algorithms for Wang tiles. Corner tiles reduce the cost of tile-based texture mapping to almost that of regular texture mapping. This is a significant saving for real-time and interactive applications.

Corner tile packings are more difficult to compute than Wang tile packings. Fortunately, they have to be computed only once. People who want to use the texture mapping algorithm based on corner tiles do not have to implement the algorithm to compute a tile packing; the solution shown in Figure 9 can be used.

7. POISSON DISK DISTRIBUTIONS

As a last application of corner tiles, we discuss the generation of Poisson disk distributions. Poisson disk distributions are uniform point distributions in which all points are separated by a minimum distance.

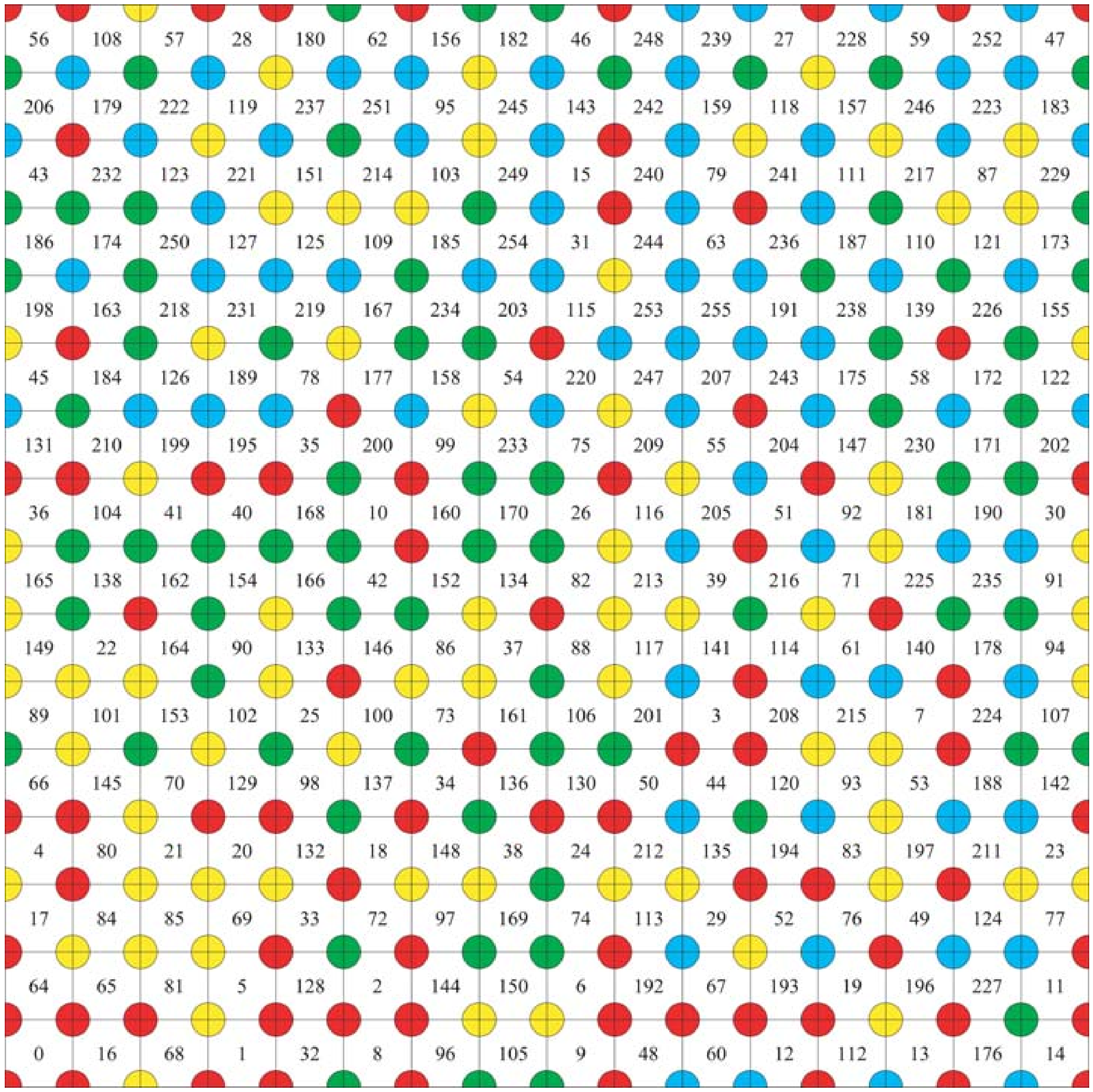

Fig. 9. A tile packing of the complete set of corner tiles over four colors. This tile packing is recursive: Solutions for one, two, and three colors are embedded in the lower left corner. Note that all these tile packings are toroidal.

Half this distance is called the radius r of the distribution. If a disk of this radius is placed at each point, then no two disks overlap.

Poisson disk distributions have many applications in computer graphics. Unfortunately, they are expensive to generate. Tile-based approaches such as that we present here try to solve this problem. Once a Poisson disk distribution is generated over a tile set, arbitrary large Poisson disk distributions can be generated in real time. However, generating a Poisson disk distribution over a tile set is typically very hard.

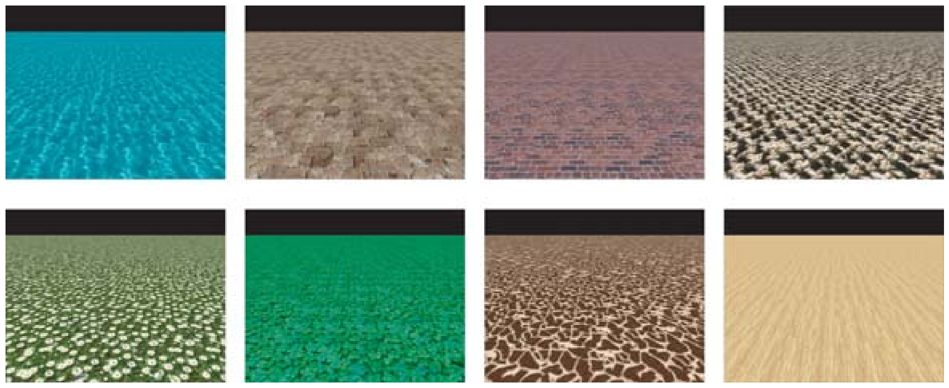

Fig. 10. Tile-based texture mapping on graphics hardware. Screenshots from our interactive tile-based texture mapping application based on corner tiles. Texture filtering does not introduce discontinuity artifacts because a tile packing was used.

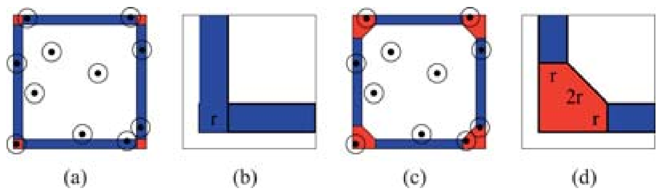

Fig. 11. Tile regions; (a) the Poisson disk radius determines corner and edge regions; (b) a close-up of (a); (c) the regions are modified so that points in different edge regions do not affect each other; (d) a close-up of (c).

In this section we present our method for tile-based generation of Poisson disk distributions. We show how to construct a Poisson disk distribution over a set of corner tiles, and compare our method with previous techniques.

7.1 Tile Construction

For constructing a Poisson disk distribution over a set of corner tiles, we have adapted the approach of Lagae and Dutré [2005a].

The points near edges of tiles constrain points in one neighboring tile, and points near corners of tiles affect points in three neighboring tiles. The Poisson disk radius divides a tile into corner regions, edge regions, and an interior region. As illustrated in Figure 11, the corner regions are slightly enlarged in order to make the distance between edge regions $2r$. Hence, points in edge regions only affect points in corner regions.

A set of corner tiles marked with tile regions produces a new type of tiling, illustrated in Figure 12. There are three different types of tiles: octagonal corner tiles that correspond to the union of four corner regions, edge tiles that correspond to the union of two edge regions, and interior tiles that correspond to interior regions. A complete tile set over C colors results in C octagonal corner tiles, C^2 horizontal and vertical edge tiles, and C^4 interior tiles. In contrast to Lagae and Dutré [2005a], we will first construct a Poisson disk distribution over the corner tiles, and then over edge tiles. Figure 15 illustrates the process.

First, a Poisson disk distribution is constructed over a corner (Figure 13(a)). The number of points per tile N is chosen. For each corner tile, a toroidal Poisson disk distribution of N points is generated

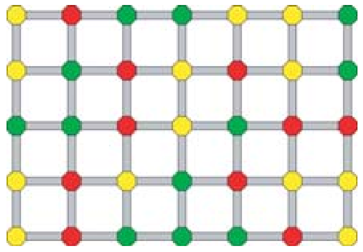
Fig. 12. A tiling constructed by interpreting the tile regions as markings on a set of corner tiles, generated from the tiling shown in Figure 3.

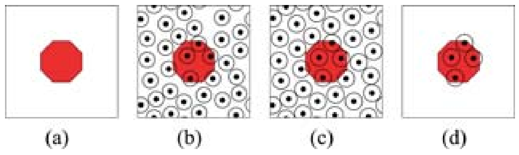

Fig. 13. Construction of a Poisson disk distribution over an octagonal corner tile.

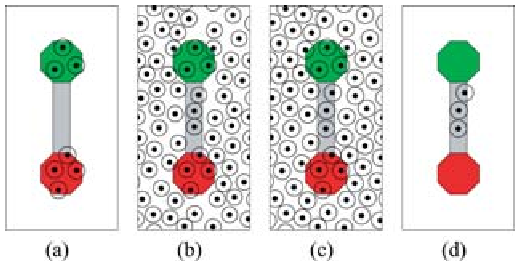

Fig. 14. Construction of a Poisson disk distribution over an edge tile.

using dart throwing (Figure 13(b)), optionally followed by Lloyd's relaxation [1982] (Figure 13(c)). The corner tile is then cut out of the distribution (Figure 13(d)).

The edge tiles are constructed by starting from appropriate corner tiles (Figure 14(a)), and generating a toroidal Poisson disk distribution of the same density using dart throwing (Figure 14(b)), also optionally followed by Lloyd's relaxation (Figure 14(c)). No new points are added in the corner tiles, and during relaxation, points are prohibited to enter the corner tiles. The edge tile is then cut out of the distribution (Figure 14(d)).

The final tiles are constructed by assembling the appropriate configuration of four octagonal corner tiles and four edge tiles (Figure 15(a)). This assembly is embedded in a toroidal Poisson disk distribution generated with dart throwing (Figure 15(b)), optionally followed by Lloyd's relaxation (Figure 15(c)). The tile is then cut out of the Poisson disk distribution (Figure 15(d)). The number of points added to the interior of the tile is chosen to bring the total number of points in the tile to N. The points outside the tile serve as a buffer for the relaxation. Figure 16 shows a Poisson disk distribution generated using corner tiles.

Although the difference between this approach and that described in Lagae and Dutré [2005a] is relatively small, the increase in quality of the resulting Poisson disk distributions is large. Tile sets obtained with this method are also significantly smaller than those obtained with the method of Lagae and Dutré [2005a] (C^4 versus C^{12}).

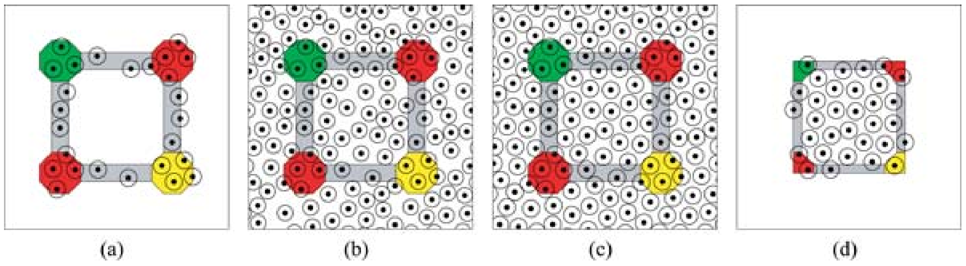
Fig. 15. Construction of a Poisson disk distribution over a tile.

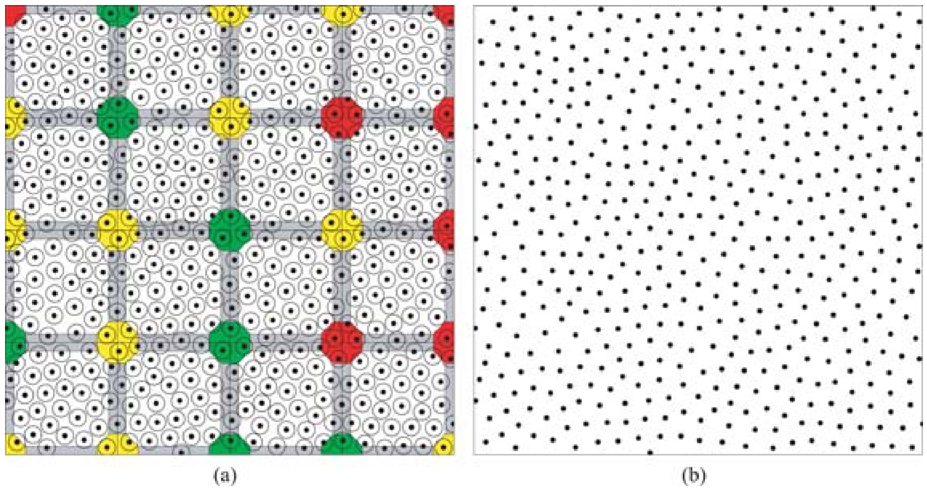
Fig. 16. A 4 × 4 tiling of corner-based Poisson disk tiles, and the resulting Poisson disk distribution.

7.2 Analysis

Analyzing Poisson disk distributions is typically done using radius statistics and spectral analysis. These measures cover a wide range of applications.

7.2.1 *Radius Specification.* Rather than providing the radius as an absolute value, it is expressed as a fraction $\rho \in [0 \cdots 1]$ of the maximum radius r_{max} that can be attained when distributing N points over the (toroidal) unit square [Lagae and Dutré 2005a]. The maximum radius is given by

$$r_{max} = (2\sqrt{3}\,N)^{-1/2}. \tag{3}$$

The relative radius ρ is independent of the number of points and size of the domain of the Poisson disk distribution. Poisson disk distributions have a relative radius that is large ($\rho \geq 0.65$), but not ($\rho \leq 0.85$) because regularity must be avoided.

7.2.2 *Spectral Analysis.* We analyze Poisson disk distributions with the technique introduced by Ulichney [1987]. The power spectrum of the distribution is estimated by averaging periodograms. The periodogram of a Poisson disk distribution is the magnitude squared of the Fourier transform of the distribution. Two 1D statistics are derived from the power spectrum. The first is the radially averaged power spectrum obtained by averaging the power spectrum estimate in concentric annuli. This power spectrum should follow the typical blue noise spectrum. The second statistic is the anisotropy, defined as the sample variance of the frequency samples within an annulus. The anisotropy should be low, indicating good radial symmetry. The spectral analysis of a Poisson disk distribution generated with dart throwing and Lloyd's relaxation [1982] is shown in Figure 18(a).

7.3 Discussion

In this subsection we compare our corner-based Poisson disk tiles with the two previous approaches, Hiller's Poisson disk tiles [Hiller et al. 2001] (adopted in Cohen et al. [2003]) and edge-based Poisson disk tiles [Lagae and Dutré 2005a], in terms of radius, number of points per tile, number of tiles, and spectral properties.

7.3.1 *Radius.* We have reviewed the original data of Hiller et al. [2001] and found that the radius of Poisson disk distributions generated with these tiles is relatively low (often $\rho \approx 0.40$ or even less). This low radius is caused by samples near the tile boundary, indicating convergence problems with the relaxation scheme. Edge-based Poisson disk tiles [Lagae and Dutré 2005a] and our corner-based Poisson disk tiles are capable of generating Poisson disk distributions with a high radius (up to $\rho = 0.85$).

7.3.2 *Number of Tiles and Number of Points per Tile.* Recent studies [Lagae and Dutré 2005b, 2006a] showed that 32 points per tile and 32 tiles are needed to produce Poisson disk distributions with acceptable spectral properties, and that 256 tiles usually result in Poisson disk distributions with good spectral properties. Hiller et al. [2001] use a set of eight Poisson disk tiles, significantly lower than the aforementioned bounds. Their construction method will also not handle larger tile sets due to the previously described convergence problems. A set of edge-based Poisson disk tiles [Lagae and Dutré 2005a] over C colors consists of C^{12} tiles. The only practical choice for C is two. This results in a set of 4,096 tiles, which is much more than needed. A set of corner-based Poisson disk tiles over C colors consists of only C^4 tiles. For three, four, and five colors this results in 81, 256, and 525 tiles, respectively. This is close to the ideal number of tiles, and leaves room for trading spectral quality for tile set size. The difference between C^4 for corner tiles and C^{12} for Wang tiles is a clear example of how the corner problem complicates construction methods and causes combinatorial explosions.

7.3.3 *Spectral Analysis.* Figures 17 and 18 show the spectral analysis of corner tiles and several techniques for generating Poisson disk distributions. As in Ulichney [1987], the spectral estimates in this article were produced by averaging ten periodograms. Therefore, an anisotropy of $-10\,dB$ should be considered background noise, and a reference line at $-10\,dB$ appears in all anisotropy plots. Each distribution counts 16,384 (or approximately) points. All spectra and plots are at the same scale. The width of the annuli is one sample, and the radially averaged power spectrum is normalized. The high-magnitude DC peak has been removed from all plots. The power spectrum images were tone mapped with a logarithmic tone mapper, using the same settings for all images.

The spectral analysis of a Poisson disk distribution generated with dart throwing and Lloyd's relaxation (Figure 18(a)) and a uniform distribution (Figure 18(b)) are included as reference.

Compared to the reference power spectrum, the power spectra of our corner-based Poisson disk tiles (Figure 17(a)–(j)) are significantly better than the power spectra of Hiller et al.'s Poisson disk tiles [2001] (Figure 18(c) and (d)) and edge-based Poisson disk tiles [Lagae and Dutré 2005a] (Figure 18(g) and (h)).

The power spectra of tiled Poisson disk distributions reveal the underlying tiling through a regular pattern of spikes. These are caused by using only a limited number of tiles, and by tiles that have points at the same location near the tile boundary. Note that in previous work, the spectral analysis was often performed with annuli that were too wide, effectively smoothing graphs and hiding spikes. For corner-based Poisson disk tiles, the spikes are much lower than those of alternate approaches. This is because corner-based Poisson disk tiles allow additional different edge configurations. A set

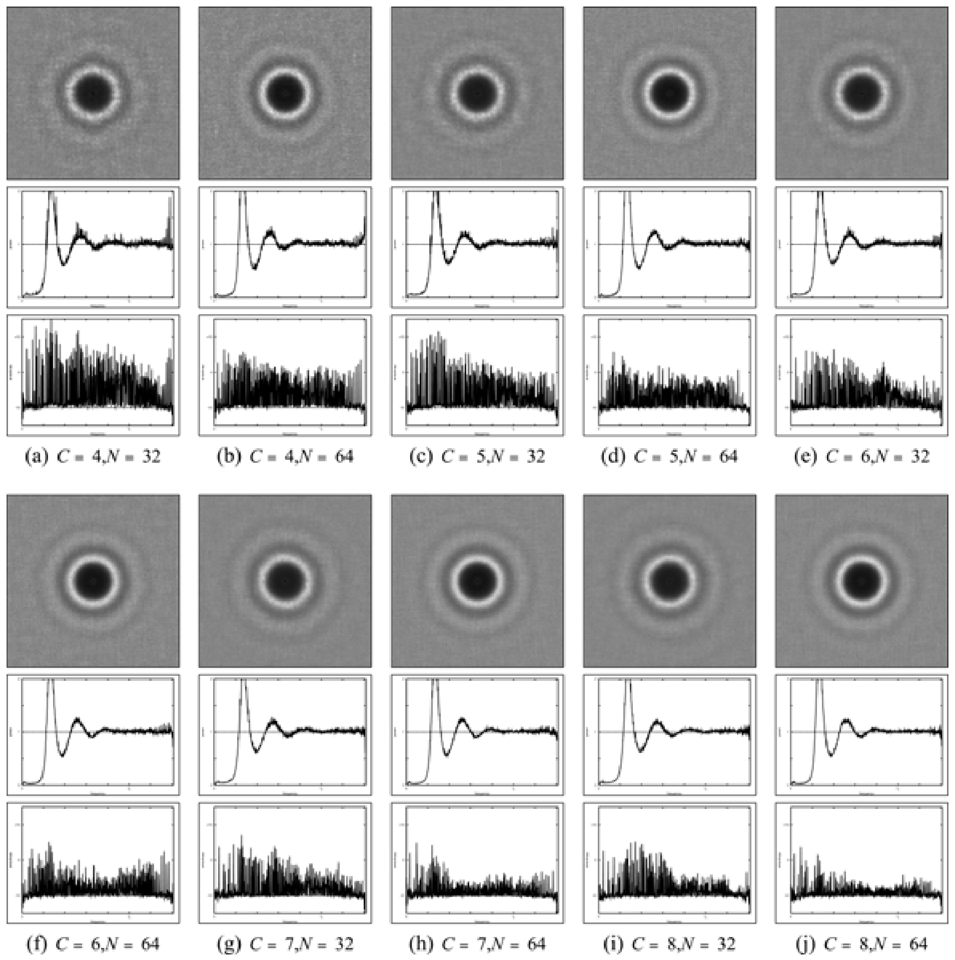

Fig. 17. Spectral analysis of tiled Poisson disk distributions generated with corner tiles. The number of colors C and number of points per tile N are indicated. Tile sets over four, five, six, seven, and eight colors consist of, respectively, 256, 625, 1296, 2401, and 4096 tiles. The tiled Poisson disk distributions with 32 and 64 points per tile were generated by tiling, respectively, 23×23 and 16×16 tiles.

of edge-based Poisson disk tiles over C colors results in only C different edge configurations and C^4 different corner configurations, while a set of corner-based Poisson disk tiles over C colors results in C^2 edge configurations and C corner configurations. Because the expected number of points in corner regions is much smaller than in edge regions (e.g., ≈ 3 points in corner regions versus ≈ 10 points in edge regions for $N = 64$ and $\rho = 0.75$), more variation in edge configurations is preferable to more variation in corner configurations. The fact that edge-based Poisson disk tile sets are much larger than those of corner-based for the same number of colors amplifies this difference significantly.

We have also included a comparison with the approach of Ostromoukhov et al. [2004]. Because their method is deterministic, we could only compute a periodogram, instead of a power spectrum estimate (Figure 18(e)). The spikes therein form a star-like pattern, revealing the underlying Penrose tiling. We have included a periodogram of a Poisson disk distribution generated with our corner-based Poisson disk tiles as a reference (Figure 18(f)). The periodogram of Ostromoukhov et al.'s [2004] approach is not

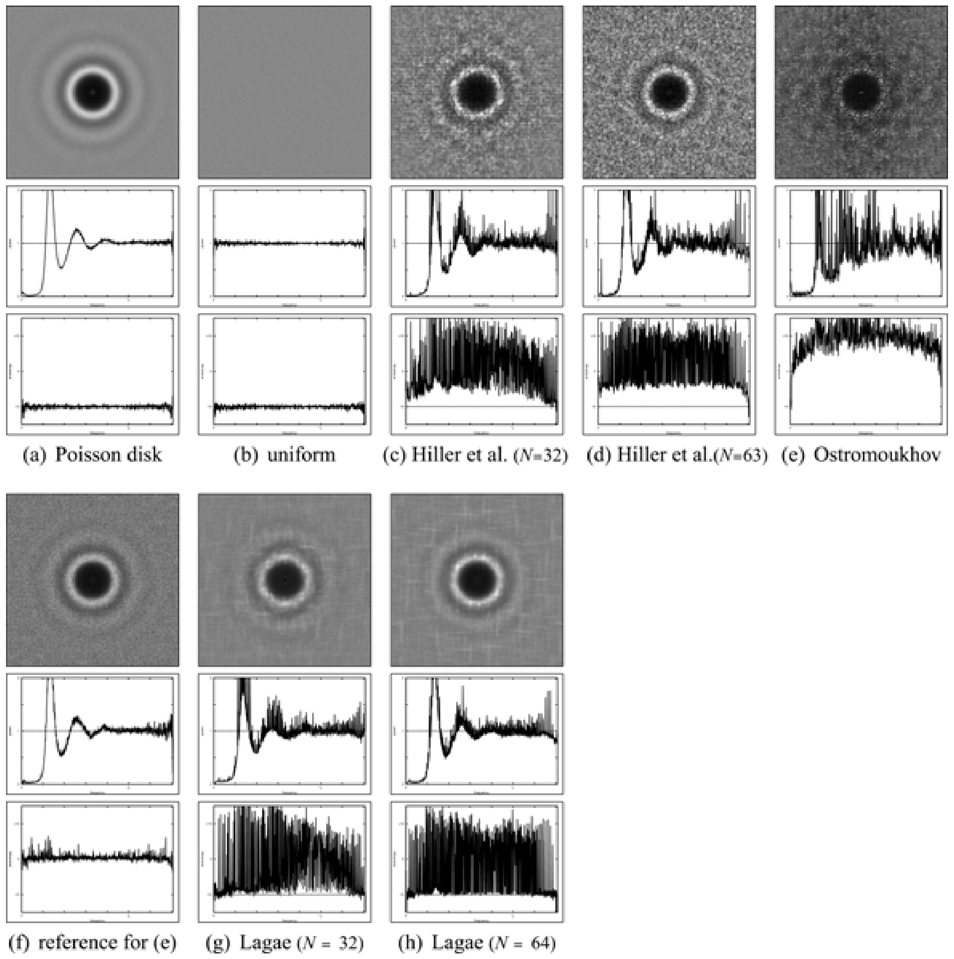

Fig. 18. Spectral analysis of Poisson disk distributions generated with (a) dart throwing and Lloyd's relaxation; (c) and (d) Hiller et al.'s Poisson disk tiles; (e) Ostromoukhov et al.'s method, and (g) and (h) edge-based Poisson disk tiles. Also shown is (b) the spectral analysis of a uniform distribution, and (f) a reference for (e). For tiled Poisson disk distributions, the number of points per tile N is indicated. The tiled Poisson disk distributions with 32, 63, and 64 points per tile were generated by tiling, respectively, 23×23, 16×16, and 16×16 tiles. The edge-based tile set over two colors was used for (g) and (e) counts 4,096 tiles.

that good, but the method does allow us to generate Poisson disk distributions with varying density. More details about the comparison of methods for generating Poisson disk distributions can be found in Lagae and Dutré [2006a].

The corner-based Poisson disk tiles we present produce Poisson disk distributions with a high radius and good spectral properties. They can be used to improve existing applications of Poisson disk distributions, such as sampling, geometry instancing [Deussen et al. 1998], object distribution for illustration [Secord et al. 2002], and procedural texturing [Lagae and Dutré 2005a].

8. CONCLUSION

In this article we have introduced corner tiles as an alternative for Wang tiles. In contrast to Wang tiles, corner tiles guarantee continuity over tile corners, as well as tile edges, and are not subject to the corner problem. We have proposed a simple and efficient direct stochastic tiling algorithm for corner tiles. We also have revisited existing applications of Wang tiles, and we have shown that corner tiles have significant advantages over Wang tiles. Textures synthesized with corner tiles contain more samples from the original texture, corner tiles reduce the required texture memory by a factor of two for tile-based texture mapping, and Poisson disk distributions generated with corner tiles have better spectral properties, even when using smaller tile sets. Corner tiles result in cleaner, simpler, and more efficient applications. We hope that future tile-based applications will consider corner tiles as an alternative for Wang tiles.

There are several opportunities for future work. It would be interesting to construct tiles that allow efficient generation of Poisson disk distributions over a given density, in the spirit of Ostromoukhov et al. [2004]. We would also like to extend corner tiles to 3D. A possible application is a solid version of the texture basis function of Lagae and Dutré [2005a]. We are also interested in employing advanced texture synthesis methods such as those of Liu et al. [2004] and Kwatra et al. [2005] for synthesizing a texture over a set of corner tiles. Finally, we would like to find a closed-form solution for the corner tile packing problem.

ACKNOWLEDGMENTS

We would like to thank Wang Yue, Tuen-Young Ng, and Tiow-Seng Tan from the National University of Singapore for generating the corner tile sets for texture synthesis. Thanks to Stefan Hiller for sharing the data of Hiller et al. [2001]. We are grateful to the authors of Ostromoukhov et al. [2004] for making example code available.

REFERENCES

Ball, W. R. 1926. *Mathematical Recreations and Essays*. MacMillan, Indianapolis, IN.

Berger, R. 1966. *The Undecidability of the Domino Problem.* American Mathematical Society.

Cohen, M. F., Shade, J., Hiller, S., and Deussen, O. 2003. Wang tiles for image and texture generation. *ACM Trans. Graph.*, 287–294.

Cook, R. L. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graph. 5*, 1, 51–72.

Culik, II, K. 1996. An aperiodic set of 13 Wang tiles. *Discrete Math. 160,* 1–3, 245–251.

DeBonet, J. S. 1997. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Computer Graphics Proceedings*. Annual Conference Series, 361–368.

Deussen, O., Hanrahan, P., Lintermann, B., Měch, R., Pharr, M., and Prusinkiewicz, P. 1998. Realistic modeling and rendering of plant ecosystems. *Computer Graphics Proceedings.* Annual Conference Series, 275–286.

Dippé, M. A. Z. and Wold, E. H. 1985. Antialiasing through stochastic sampling. In *Proceedings of the 12th Annual Conference Computer Graphics and Interactive Technology. 19*, 3, 69–78.

Ebert, D. S., Musgrave, F. K., Peachey, D., Perlin, K., and Worley, S. 2002. *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann, San Fransisco, CA.

Efros, A. A. and Freeman, W. T. 2001. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th Annual Conference Computer Graphics and Interactive Technology.* 341–346.

Efros, A. A. and Leung, T. K. 1999. Texture synthesis by non-parametric sampling. In *Proceedings of the International Conference on Computer Vision*. 1033–1038.

Fu, C.-W. and Leung, M.-K. 2005. Texture tiling on arbitrary topological surfaces using Wang tiles. In *Rendering Techniques*. 99–104.

Glassner, A. 1999. *Andrew Glassner's Notebook: Recreational Computer Graphics*. Morgan Kaufmann, San Fransisco, CA .

Grünbaum, B. and Shepard, G. C. 1986. *Tilings and Patterns*. W. H. Freeman.

Heeger, D. J. and Bergen, J. R. 1995. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd Annual Conference Computer Graphics and Interactive Techniques*. 229–238.

Hiller, S., Deussen, O., and Keller, A. 2001. Tiled blue noise samples. In *Proceedings of the Vision Modeling Visualization*, 265–272.

Knuth, D. E. 1968. *The Art of Computer Programming Conference*. vol. 1. Addison-Wesley, Reading, MA.

KWATRA, V., ESSA, I., BOBICK, A., AND KWATRA, N. 2005. Texture optimization for example-based synthesis. *ACM Trans. Graph. 24*, 3, 795–802.

KWATRA, V., SCHÖDL, A., ESSA, I., TURK, G., AND BOBICK, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Trans. Graph. 22*, 3, 277–286.

LAGAE, A. AND DUTRÉ, P. 2005a. A procedural object distribution function. *ACM Tran. Graph. 24,* 4.

LAGAE, A. AND DUTRÉ, P. 2005b. Template Poisson disk tiles. Report CW 413, Department of Computer Science, K.U. Leuven, Leuven, Belgium. May.

LAGAE, A. AND DUTRÉ, P. 2006a. A comparison of methods for generating Poisson disk distributions. Report CW 459, Department of Computer Science, K.U. Leuven, Leuven, Belgium. Aug.

LAGAE, A. AND DUTRÉ, P. 2006b. The tile packing problem. Report CW 461, Department of Computer Science, K.U. Leuven, Leuven, Belgium. Aug.

LAGAE, A., KARI, J., AND DUTRÉ, P. 2006. Aperiodic sets of square tiles with colored corners. Report CW 460, Department of Computer Science, K.U. Leuven, Leuven, Belgium. Aug.

LEFEBVRE, S. AND NEYRET, F. 2003. Pattern-Based procedural textures. In *Proceedings of the Symposium on Interactive 3D Graphics*. 203–212.

LIANG, L., LIU, C., XU, Y.-Q., GUO, B., AND SHUM, H.-Y. 2001. Real-Time texture synthesis by patch-based sampling. *ACM Trans. Graph. 20*, 3, 127–150.

LIU, Y., LIN, W.-C., AND HAYS, J. 2004. Near-Regular texture analysis and manipulation. *ACM Trans. Graph. 23*, 3, 368–376.

LLOYD, S. P. 1982. Least squares quantization in PCM. *IEEE Trans. Info. Theory 28,* 2, 129–137.

MACMAHON, M. P. A. 1921. *New Mathematical Pastimes*. Cambridge University Press, New York.

MCCOOL, M. AND FIUME, E. 1992. Hierarchical Poisson disk sampling distributions. *Graphics Interface*, 94–105.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques.* 65–72.

MITCHELL, D. P. 1991. Spectrally optimal sampling for distribution ray tracing. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques.* 157–164.

NEYRET, F. AND CANI, M.-P. 1999. Pattern-Based texturing revisited. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques.* 235–242.

NG, T.-Y., WEN, C., TAN, T.-S., ZHANG, X., AND KIM, Y. J. 2005. Generating an ω-tile set for texture synthesis. In *Proceedings of the Computer Graphics International Conference*. 177–184.

OSTROMOUKHOV, V., DONOHUE, C., AND JODOIN, P.-M. 2004. Fast hierarchical importance sampling with blue noise properties. *ACM Trans. Graph. 23*, 3, 488–495.

PERLIN, K. 2002. Improving noise. *ACM Trans. Graph.*, 681–682.

SECORD, A., HEIDRICH, W., AND STREIT, L. 2002. Fast primitive distribution for illustration. In *EGRW: Proceedings of the 13th Eurographics Workshop on Rendering*. 215–226.

SHADE, J., COHEN, M. F., AND MITCHELL, D. P. 2000. Tiling layered depth images. Tech. Rep., University of Washington, Department of Computer Science and Engineering.

STAM, J. 1997. Aperiodic texture mapping. Tech. Rep. ERCIM-01/97-R046, European Research Consortium for Informatics and Mathematics (ECRIM).

ULICHNEY, R. 1987. *Digital Halftoning*. MIT Press, Cambridge, MA.

WANG, H. 1961. Proving theorems by pattern recognition II. *Bell Syst. Tech. J. 40*, 1–42.

WANG, H. 1965. Games, logic and computers. *Sci. Amer. 213*, 5, 98–106.

WEI, L.-Y. 2004. Tile-Based texture mapping on graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*. 55–63.

WEI, L.-Y. AND LEVOY, M. 2000. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques.* 479–488.

YELLOT, J. I. 1983. Spectral consequences of photoreceptor sampling in the rhesus retina. *Sci. 221*, 382–385.

Received April 2006; accepted August 2006