EUROGRAPHICS 2017 / L. Barthe and B. Benes
(Guest Editors)

Volume 36 (2017), Number 2

A Subjective Evaluation of Texture Synthesis Methods

M.Kolář[1,2], K. Debattista[1] and A. Chalmers[1]

[1]University of Warwick
[2]Brno University of Technology

Abstract

This paper presents the results of a user study which quantifies the relative and absolute quality of example-based texture synthesis algorithms. In order to allow such evaluation, a list of texture properties is compiled, and a minimal representative set of textures is selected to cover these. Six texture synthesis methods are compared against each other and a reference on a selection of twelve textures by non-expert participants ($N = 67$). Results demonstrate certain algorithms successfully solve the problem of texture synthesis for certain textures, but there are no satisfactory results for other types of texture properties. The presented textures and results make it possible for future work to be subjectively compared, thus facilitating the development of future texture synthesis methods.

1. Introduction

Exemplar-based texture synthesis has numerous applications across computer graphics, such as inpainting, rendering, and manufacturing. Thirty years of research has resulted in a wide array of approaches. However, texture quality can be subjective, and independent evaluation of known methods is lacking. There is currently no structured way of evaluating the quality of texture synthesis algorithms, making it difficult to gain an understanding of which methods perform better, and under which circumstances.

In this work, a method for analyzing textures is proposed and used in a subjective experiment involving six representative texture synthesis methods and a reference (figure 1). A minimal set of twelve textures is created for this evaluation. These textures have been selected by compiling a list of 21 properties, and choosing textures such that the full range of each property is covered. This allows the analysis of texture synthesis quality for specific texture properties.

The benefit of this work is twofold: First, this work is the first to compare texture synthesis algorithms in a subjective study on textures selected to represent the wide variability of all textures. Statistically significant preferences are identified for algorithms, as well as over individual textures. Second, we offer a minimal set of textures with which future work may be subjectively compared, to promote the quality enhancement of future texture synthesis algorithms.

2. Background and Motivation

Texture Synthesis algorithms are typically compared on a small sample, possibly resulting in inconsistent evaluations of new methods and the misunderstanding of method properties. This work identifies textures which cover a wide range of properties, and uses them to synthesize outputs with six representative algorithms. A study using these outputs finds statistically significant user preferences between methods.

Unlike general images, textures must satisfy stationarity and locality. The first criterion requires any two patches to be visually similar, and the second criterion requires that any pixel be only related to a small set of neighboring pixels. As these criteria are satisfied to different degrees (for example a uniform noise image satisfies them perfectly), so must texture synthesis algorithms be able to handle textures with limited locality, stationarity, and other properties.

Many texture synthesis algorithms have been devised over the past 30 years [WLKT09], with a focus on various aspects: quality, speed, parallelism, use for animation, manufacturing quality, and others. In this paper, we focus on the quality of the synthesized texture. Despite attempts to quantify synthesized texture quality via texture energy metrics [KSE*03] or image statistics [Bal06], texture quality remains a subjective notion because optimizing these metrics does not guarantee textures of high visual quality [JFA*15]. However, no independent comparative user study has been performed to assess texture synthesis algorithms.

Several texture datasets have been previously created to demonstrate and evaluate texture synthesis: Brodatz Textures [Bro66], VisTex Textures [Gra95], DeBonet Textures [DB97], Colored Brodatz [ADC13], the PSU Near-Regular Texture Database [LL05], Simoncelli Textures [PS00], and many others. See Hossain and Serikawa [HS13], and Bianconi and Fernández [BF14] for a complete survey of Texture Databases. None of these cover the

(a) reference (b) resynthesizer (c) quilting (d) self-tuning (e) LazyFluids (f) Ashikhmin (g) CNNMRF

Figure 1: The *smarties* texture synthesized with selected methods, ordered by user preference

full range of published texture properties [Lou90, FH93, KSS97, LFTG97, BFH*98, FvDF*93] which the various methods claim to handle.

Texture synthesis quality evaluation is an inherently subjective problem. In order to clarify it, various schemes have focused on a subset of textures, by classifying textures by properties, so that appropriate synthesis algorithms may be created for specific applications. Some of the proposed classification schemes are the continuous texture spectrum from regular to stochastic [LHW*04a], and the A-score and G-score of Liu, Lin, and Hays [LLH04], promoting the creation of algorithms specifically for regular and near-regular textures.

Comparative studies have been performed for tone-mapping [LCTS05], inverse tone-mapping [BLD*09], image retargetting [RGSS10], and single image blind deblurring [LHH*]. There have also been comparative studies for texture synthesis, but only through individual subjective evaluation with example textures on which the algorithms do not fail [LHW*04b].

2.1. Selected Texture Synthesis Algorithms

In order to provide a comparison across texture synthesis methods, six methods were chosen. The selected algorithms belong to one of three categories: state-of-the-art methods with a focus on quality [KNL*15, JFA*15, LW16], well-known historic methods [EF01, Ash01], and methods whose code is public and known to be regarded, outside the research community, as useful and well-performing [CR11, Ash01]. The method will be referred to by the name described in brackets after its introduction in the following sections.

Texture Synthesis by image quilting [EF01] (**quilting**) places square subsamples of the exemplar onto the output texture, optimally choosing transitions by finding nearest matches according to the overlap. Each overlapping region is then optimally cut with a minimum cost path. Various implementation based on the original paper are available at `http://people.csail.mit.edu/thouis/efros-freeman/`.

Ashikhmin's Natural Texture Synthesis [Ash01] (**Ashikhmin**) is based on Wei and Levoy's Texture Synthesis [WL00], where pixels are individually added row-by-row by finding the best matching candidate according to the surrounding pixel similarity. Ashikhmin improves this search by focusing on several candidates, thus encouraging verbatim copying instead of blurring. An implementation based on the original paper are available at `http://www.cs.utah.edu/~michael/tscode/`.

Resynthesizer [CR11, Har05] is an open-source texture synthesis plugin. In this algorithm, pixel values are chosen one at a time in a random order. When choosing the value of a pixel, the n nearest pixels that already have values are located, and the input image is searched for a good match to the pattern these pixels form. Once a good match is found, the appropriate pixel value is copied from the input texture to the output. To increase quality, some earlier chosen pixel values are re-chosen after later pixel values have been chosen. The source code and precompiled binary are available at `http://www.logarithmic.net/pfh/resynthesizer`.

Self Tuning Texture Optimization [KNL*15] (**self tuning**) is a general-purpose and fully automatic self-tuning non-parametric texture synthesis method. Various parameters and weights are tuned by focusing on three aspects of texture synthesis: irregular large scale structures are faithfully reproduced through the use of automatically generated and weighted guidance channels, repetition and smoothing of texture patches is avoided by new spatial uniformity constraints, and a smart initialization strategy is used in order to improve the synthesis of regular and near-regular textures [LLH04] without affecting textures that do not exhibit regularities. The Matlab code is available from the authors.

LazyFluids Appearance Transfer [JFA*15] (**LazyFluids**) extends Graphcut Textures [KSE*03] which minimizes Texture Energy

$$E(Z,X) = \sum_{p \in X} \min_{q \in Z} ||x_p - z_q||^2 \quad (1)$$

where Z is the source texture, and X is the output. However, this is known to create an output image which matches only a portion of the input, for example, blurred parts, and quality is highly dependent on selected parameters. LazyFluids resolves these issues by using a nearest-neighbor field to assure uniform source patch usage.

CNNMRF [LW16] is based on Neural Style [GEB15], which uses statistics of higher levels of a pre-trained Convolutional Neural

Network, and iteratively adjusts a random noise image to match the statistics of a texture exemplar. However, CNNMRF also fits a Markov Random Field over neuron activations, in order to better match the texture properties. The implementation code is available at https://github.com/chuanli11/CNNMRF.

Note that Self Tuning [KNL*15] and CNNMRF [LW16] require hours for large textures, while the other algorithms run in seconds or minutes. Exact times are not reported here because they vary with implementations, and this work focuses on quality. For the data used in the experiment, we performed synthesis for all algorithms except LazyFluids. The code for this method is not public, and synthesis was performed on our request by the authors of the method.

3. Texture Dataset

This section describes the process taken to establish a small representative set of textures. First, the shortcomings of the most popular datasets and the goals of this dataset are discussed in section 3.1. Then, texture properties are individually listed and explained in section 3.2, and section 3.3 lists and shows chosen textures.

3.1. Dataset Goals

Previous texture datasets have been devised with various goals, from the creation of comprehensive tileable textures for computer games to demonstrations of artistic renderings. However, despite the number and size of available datasets, none of them fulfill requirements for a minimal set of textures which are representative of known texture properties. None of these datasets attempt to cover texture properties as listed below in a structured way.

Certain datasets contain non-textures as well as textures, making them an interesting tool for understanding the inner workings and limitations of texture synthesis algorithms. However, non-texture images are not informative when assessing quality.

Most datasets contain more than 50 textures. This makes it challenging to perform a detailed evaluation, as evaluation requires human observation of the results. It also leads necessarily to manual selection of representative textures when publishing, thereby making it challenging to compare algorithms where authors have chosen different exemplars.

Numerous datasets also lack the variety of properties which is necessary for a robust evaluation: all textures are at the same scale, at the same resolution, or produced with the same camera. This creates a bias toward certain properties, making the results of algorithm evaluation challenging to extrapolate.

Lastly, no existing application-independent texture dataset contains exemplars as well as reference textures. By making this available, new methods of evaluation are possible, namely comparison to a ground truth.

3.2. Texture Properties

Textures are expected to exhibit stationarity and locality [WLKT09], but only a random noise image can satisfy these perfectly. Texture synthesis algorithms must be able to handle textures with varying locality, stationarity, and various other properties described here, and in tables 1 and 2. Ordinal properties are listed individually in sections 3.2.1 to 3.2.14, followed by binary properties in section 3.2.15. The selection combines properties discussed in other texture synthesis publications. Whenever possible, a clear definition of how each property is measured is included, but in some cases it is subjective and relative.

3.2.1. Scale

Scale refers to the size of texture elements in relation to the size of the entire example. Textures with a small repeating element are considered fine-grained (low scale), such as the *rough* texture. Conversely, textures where the repeating element is large are considered blown-up (high scale), such as *straw* or *green marble*, where certain texture elements continue across the entire texture. Furthermore, textures may be Multi-Scale, exhibiting texture elements at various scales simultaneously. Note that this property is independent of resolution.

3.2.2. Stochasticity

Stochasticity, or randomness, refers to the random variability within a texture. This can be formulated as follows *Given information of other pixels in the texture, how predictable is another pixel?* Therefore, even textures which are entirely random globally (such as the *smarties* texture, where each element is placed randomly) do not exhibit perfect stochasticity, because nearby pixels are likely to belong to the same texture element. Similarly, textures where pixels are likely to change locally are not entirely stochastic if the structure is periodic (regular), such as the *grid* texture.

3.2.3. Stationarity

Stationarity is the property which defines to what degree variance is linked with neighborhood. A stationary pattern is similar to a local pattern, in that texture elements are dependent on local pixel neighborhoods. However, a non-stationary pattern may be local or non-local, because both local and global underlying patterns can affect pixels to varying degrees. Note that texture stationarity is independent of texture locality, because a non-stationary pattern may simultaneously be highly local, such as the flow-guided texture *ink*.

3.2.4. Locality

Locality refers to the property of textures to depend on a local neighborhood. Regular textures, those which fit onto a repeating lattice, exhibit low locality, because pixel values depend on the entire texture, such as the *straw* or *grid* textures. Textures exhibiting locality are for example the *blades* and *smarties* textures, because small neighborhoods are locally independent from the rest of the texture.

3.2.5. Flow-guided

If a texture was generated in a process involving motion with some continuity, it is said to be flow-guided. This property can be seen to varying degrees, so the flow-guided property is ordinal, rather than binary.

Property	Definition	Low	Mid	High
Scale	fine-grained(fine) to blown-up(coarse)	rough	ink	pebbles
Stochasticity	entirely deterministic to entirely random	grid	straw	ink
Stationarity	different regions are perceived to be similar	smarties	orange marble	chicago
Locality	pixel values depend on small neighborhood	ink	pebbles	rough
Flow-guided	generative process includes flow	pebbles	straw	green marble
Shape variance	shapes do not vary to wide shape variance	grid	smarties	ink
Color variance	color does not vary to high color variance	blades	green marble	smarties
Natural	natural versus simulated	ink	chicago	orange marble
Absolute Texel size (resolution)	number of pixels in a texel	rough	blades	smarties
Texels per sample	number of texels in example	pebbles	chicago	blades
A-score	appearance regularity	chicago	straw	rough
G-score	geometric regularity	blades	chicago	grid
Scale variance	variation in scale of elements within texture	smarties	ink	green marble
Rotation	texture rotation invariance	grid	straw	blades
Example resolution	tiny to huge	straw	ink	smarties

Table 1: Ordinal properties of selected textures. Note that the noise and checkerboard textures are not required for completeness.

Property	true	false
Historic	straw	ink
Regular color, irregular shape	pebbles	chicago
Irregular color, regular shape	smarties	straw
Multi-scale	chicago	blades
Global variance	green marble	rough
Overlapping multiple textures	chicago	grid
Color	blades	straw

Table 2: Binary properties of selected textures

Texture	Source	Size
 blades	[ADC13]	640 × 640
 grid	[Bro66]	640 × 640
 pebbles	[Bro66]	640 × 640
 rough	[ADC13]	640 × 640
 orange marble	own work	1024 × 1340
 green marble	own work	657 × 876
 straw	[Bro66]	256 × 256
 chicago	[cre14]	2266 × 2267
 ink	own work	1281 × 653
 smarties	[Sch12]	3543 × 2362
 checkerboard	own work	256 × 256
 noise	own work	1000 × 1000

Table 3: Selected textures, with images of the exemplar. All textures are freely available under various licenses at full resolution.

3.2.6. Shape Variance

Variance refers to variation of the shape of texture elements. For example, the *smarties* texture has low shape variation of its elements, therefore exhibiting high regularity. Conversely, low shape regularity or lack of shape altogether mean high shape variance, such as textures *ink* or *blades*. Because this property is not related to the concept of a loose lattice, it is different from the G-score of section 3.2.11. However, whenever a loose lattice can be fitted onto a texture, shape variance inversely corresponds to G-regularity.

3.2.7. Color Variance

Similarly to shape variance, color variance refers to the variation of the color across elements of a texture. This can be attributed to variability between elements, and variability within a given element. For example, the *blades* texture displays only two major colors, with little variation while the *smarties* texture shows both types of color variation.

3.2.8. Natural

This property refers to the degree to which a texture has been produced in a natural process, rather than being the product of a computer simulation. The marble textures show unedited images of stone, thus being entirely natural. The ink texture has been generated in an entirely artificial way. Others, such as *chicago* and *smarties* are produced in a partly natural process, consisting of both a repeatable generative process as well as an element of natural randomness. This property, together with the Flow-guided property, can serve as a helpful indicator of repeatability of texture synthesis results across natural and synthetic textures. Because arbitrarily large samples from synthetic textures can be drawn, it is a potential benefit for texture synthesis algorithm development.

3.2.9. Absolute Texel Size

This property refers to the size in pixels of a repeating texture element. This is included to shift focus from algorithms which work at a given scale, and used to evaluate texture synthesis methods which perform well on textures with repeating elements at any scale.

3.2.10. Texels per Sample

The number of texels per sample refers to the repeating texture elements of sections 3.2.1 and 3.2.9. These three properties are inevitably interlinked, and are listed here for completeness.

3.2.11. A-score and G-score

As introduced by Liu, Lin, and Hays [LLH04], Geometric (G) Regularity and Appearance (A) Regularity of near-regular textures provide a quantitative measure of deviation from a perfectly regular texture in terms of shape, and color. These differ from shape variance and color variance defined earlier by requiring a texture to be near-regular. Textures which cannot fit into a loose lattice structure, such as the *smarties* texture of figure 1 are not near-regular, and therefore cannot have an A-score and G-score.

3.2.12. Scale Variance

The difference in scale between repeating elements in a texture. For instance, the *pebbles* texture contains pebbles of varying size, creating scale variance. On the other hand, the *grid* texture is scale invariant, because texture elements are of constant size.

3.2.13. Example Resolution

The absolute pixel size of the example texture. Similarly to Absolute Texel Size, this property enables the evaluation of algorithms which can handle small as well as large example images. Table 3 shows selected texture resolutions.

3.2.14. Rotation

Texture Rotation refers to rotation invariance of repeatable texture elements. If elements of a texture can be rotated, such as the *ink* texture, it exhibits high rotation. Conversely, if elements of the texture cannot be rotated, such as the *grid* texture, rotation is low. Note that this property does not refer to rotating the entire texture, because this is always assumed to be possible.

3.2.15. Binary Properties

In addition to the ordinal properties discussed here, it is important to consider certain categorical binary properties to quantify textures. The properties listed in table 2 are discussed in depth here, to clarify and justify their definition and selection.

Historic textures were selected because various past publications have already demonstrated their performance on them in sets of synthesized images available from the authors. Rather than requiring some textures to be old, the purpose of this property is to allow a clear link to past work.

The two **combinations of color and shape variance and regularity** have been included to match all four categories of near-regular textures according to their A-score and G-score. This produces the classification of Liu, Lin, and Hays [LLH04], where near-regular textures are divided into four types:

0 Regular Geometry, Regular Color
I Regular Geometry, Irregular Color
II Irregular Geometry, Regular Color
III Irregular Geometry, Irregular Color

As mentioned in section 3.2.1, **Multi-scale textures** contain texture elements at various scales, such that they interact with each other. For example, the *chicago* texture demonstrates this property: the road grid is a lower scale texture than the houses, but both of these properties satisfy the properties required to be a texture.

Global variance is the property that texture elements differ either by color or geometry across the texture in a global, predictable manner. While this is at odds with the basic requirement for locality, a texture demonstrating this property is included in the dataset to judge how this affects texture synthesis algorithms.

An additional property of textures is that there may be a **combination of overlapping textures**. Unlike Multi-resolution, this property requires independence of the combined textures, instead of interaction between a higher-scale repeating texture element with a lower-scale one. The texture element may even be on the same scale. The *ink* texture demonstrates this property, because there are different independent overlapping generative processes involved.

Finally, textures in both **color and grayscale** are included, to facilitate testing of single-channel texture synthesis algorithms. Single-channel texture synthesis algorithms can be applied on textures with three or more color channels as well, by simply converting colors to a single measure, or for certain methods by providing a similarity metric. However, these results would be skewed by the chosen color mapping, hence the benefit of providing two grayscale textures (*straw* and *pebbles*). For work with single-channel methods, we recommend the method of Smith et al [SLTM08] which produces accurate and perceptually preferred color to grayscale conversions according to a prior comparative study [Č08].

3.3. Texture Selection

Textures were selected such that at least one covers the low, mid, and high values of every ordinal property listed in table 1, and each case of the binary properties 2. See Table 3 for exemplar images and sources of these textures.

To allow an example and reference output to be produced from the selected textures, it is required that a subimage of $1/3$ width and $1/3$ height is a sufficient sample of the texture, as per figure 2. Therefore, some popular textures which have been widely applied by the community had to be discarded because such a small subimage did not contain a representative patch.

4. Experiment

This section describes the experimental method, including design, materials and algorithm parameter configuration, and the procedure. The interface source code and textures used in the experiment are available at `https://github.com/mrmartin/ordering_study`.

4.1. Design

The experiment compares six algorithmic methods across twelve textures. The experiment is based around a ranking design based

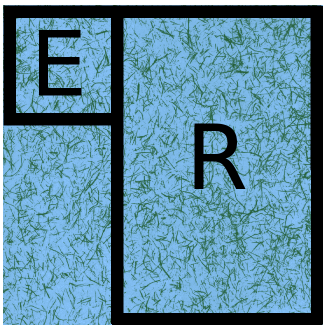


Figure 2: Process of selection of the input exemplar E and the reference output R from a square sample texture.

on similar experiments for HDR video compression comparisons [MDBR*16]. The ranking procedure can be seen in figure 3 where a reference exemplar, acting as ground truth is presented. The participant ranks the stimuli interactively according to how close to the exemplar they believe the stimuli to be. Ranking gives participants the opportunity to be able to view all stimuli an equal number of times, and be able to view them concurrently within a reasonable amount of time reducing the fatigue associated to other design choices such as pairwise comparisons. The *texture* variable corresponds to a within-participants independent variable encompassing the different texture stimuli. The *method* variable is also a within-participants independent variable referring to the distinct texture synthesis algorithms. The *method* variable also includes a hidden reference besides the texture synthesis generated textures. The hidden reference is added to help identify differences between the reference and texture synthesis methods and to see how close these are to the ground truth. The experimental question was explicitly formulated to allow multiple interpretations, asking users to "Sort [...] by how much they look like the original.", "Sort [...] by order of realism.", and "Order [...] according to how similar to the reference you think they are."

4.2. Materials

The selection of textures and texture synthesis algorithms for the study has been guided by various constraints. First, in order to compare synthesis outputs to a ground truth texture, it was necessary to choose textures large enough to be divided into an input and output sample. Furthermore, this output sample needed to be larger than the input, in order to clearly demonstrate algorithmic properties. An output aspect ratio of $2:3$ was chosen, and because it was desirable to present all textures equally on one row, the number of outputs was limited to seven, comprising the six methods and the hidden reference (see user interface in figure 3).

The input and output images were generated by first taking a texture, and cropping it to a square. Then, the top-left corner of $1/3$ width and $1/3$ height is extracted to give the exemplar. The right band of $2/3$ width and full height was used as the ground truth. See figure 2.

All selected algorithms were run with the same parameters across all textures, to simulate a non-expert user environment. Default parameters were set according to the cited publications. Note that certain algorithms required no tuning at all (self-tuning and resynthesizer), while for some finding default configurations was challenging and error-prone. Ashikhmin was executed in three passes over a 7×7 neighborhood, and Quilting was performed with a patch size half of the input, and left and top overlaps one third of the input.

4.2.1. User Interface

Figure 3 shows the GUI used for selection. The experiment was run entirely inside a browser to provide online availability; this permits easy of use and the ability to recruit a wide variety of users with various monitors, resolutions, preferences, and lighting conditions. The GUI presents the exemplar as a ground truth reference in the top center and the six methods plus hidden reference underneath. The GUI allows the selection and movement of any of the stimuli corresponding to the methods along the x-axis to be ordered according to participant preferences.

The background is a neutral gray, and the user is requested to maximize the window. All images scale to fill as much of the screen as possible, and the aspect ratio between the input and output are maintained.

4.3. Participants and Procedure

Participation was anonymous, and available to students at two international universities, and interested members of the public. Screen resolution was recorded, and participants with less than HD 720p were discarded. Participants who did not change the order of any textures, and those who did not reach the end of the experiment were discarded. In total, the subjective study was performed by 67 participants.

A mass e-mail was sent out to students and staff of two universities to recruit participants on a voluntary basis. Those who responded were sent a URL corresponding to the experiment. No personal data was collected. The experiment was run entirely in browser and results stored on a server.

5. Results and Analysis

Analysis was conducted using the non-parametric Kendall test for Concordance. Kendall's test for Concordance (W) provides a statistic between 0 and 1 conforming to the agreement across participants. A W of 1 indicates perfect agreement among participants and 0 means perfect disagreement. The overall W of this study is 0.402, which is compares well with other subjective ordering studies in Computer Graphics: 0.12 [LHH*], 0.095 [RGSS10], 0.282 [BLD*09]. W is also be tested for significance, and its p-value is below 10^{-10} for every texture, due to the high number of participants [Gwe14].

Pairwise comparisons among all the methods for each of the textures were also conducted, these give an indication of whether there are any significant differences across methods for a given texture or in the overall experiment.

Results for each texture, as well as across all textures using collapsed scores, are shown in table 4. The groupings in each row

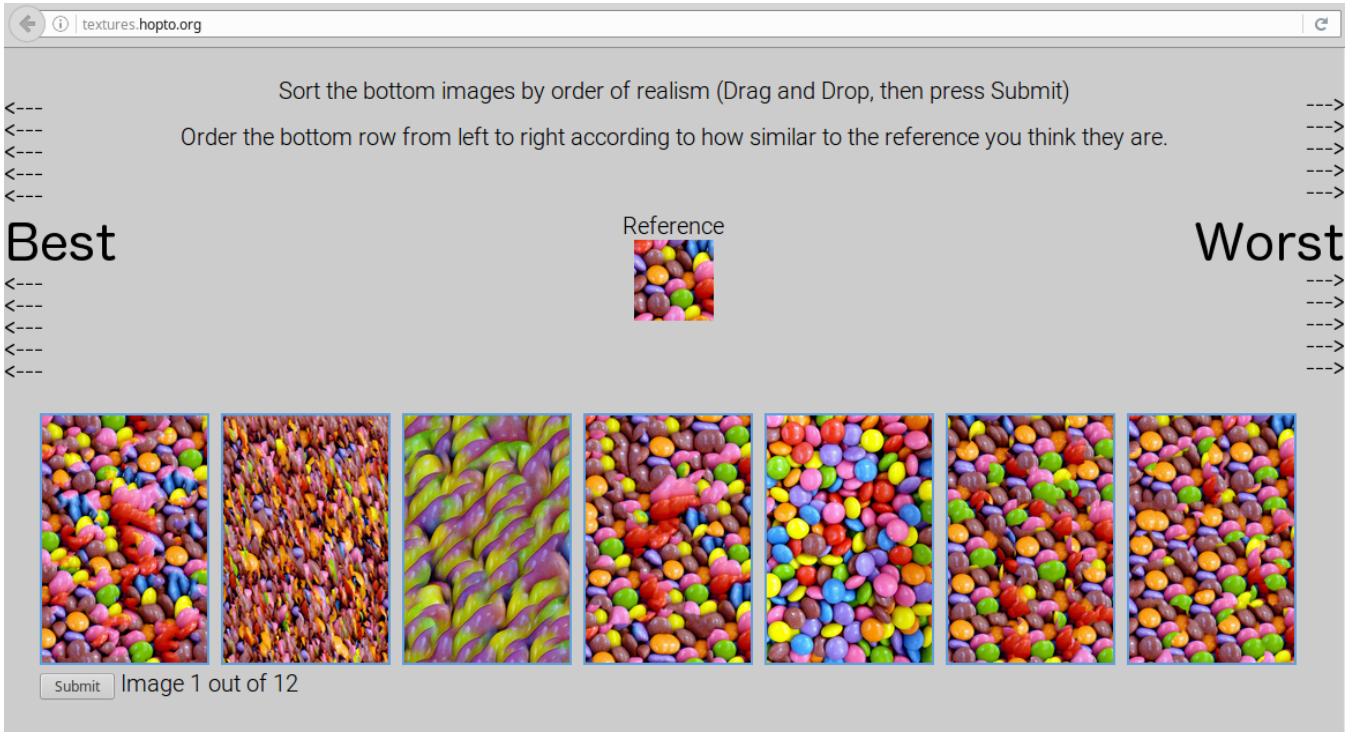


Figure 3: The experiment's user interface. The textures can be reordered interactively, and the entire page scales up to maintain correct aspect ratios between the reference and the synthesized textures.

demonstrate non-significant differences. Methods not grouped together indicate significant differences across those methods for that particular texture.

Results on the noise and checkerboard textures serve as a sanity check, verifying assumptions about the experiment. Consistently with assumptions, the noise texture is easy to synthesize correctly, because four out of six methods create results statistically indistinguishable from the reference. Results with the checkerboard texture are visually clearest (this can be seen in figure 5), they show a very high Kendall's coefficient of concordance.

Other textures cause wide disagreement. This is possibly caused by differing views on what constitutes an ideal output. For several textures, participants consistently agree that the reference is not the best (*pebbles*, *rough*, *chicago*, *green marble*). We hypothesize that this is because the reference image contains greater variance than the example, while synthesized textures closely match the inputs' visual properties (figure 6).

There are seven textures for which some methods produce results not significantly worse than the reference (*blades*, *pebbles*, *ink*, *noise*, *green marble*, *orange marble*, *straw*). The texture synthesis methods for these textures may be broadly considered successful.

Out of the remaining five textures (*smarties*, *checkerboard*, *grid*, *rough*, *chicago*), two reference textures are evaluated as significantly worse than synthesized outputs (*rough*, *chicago*), and in the other three, the reference is significantly better than synthesized textures.

5.1. Analysis of Texture Properties

There are three textures for which synthesis is not fully solved by any of the evaluated methods. These are the *smarties* texture (figure 1), *checkerboard* (figure 5), and *grid*. These share some common properties: low-to-mid shape variance, mid-to-low stochasticity, and low locality. These properties, together with complex shape patterns, form patterns which are hard to recreate without considering the underlying generative process.

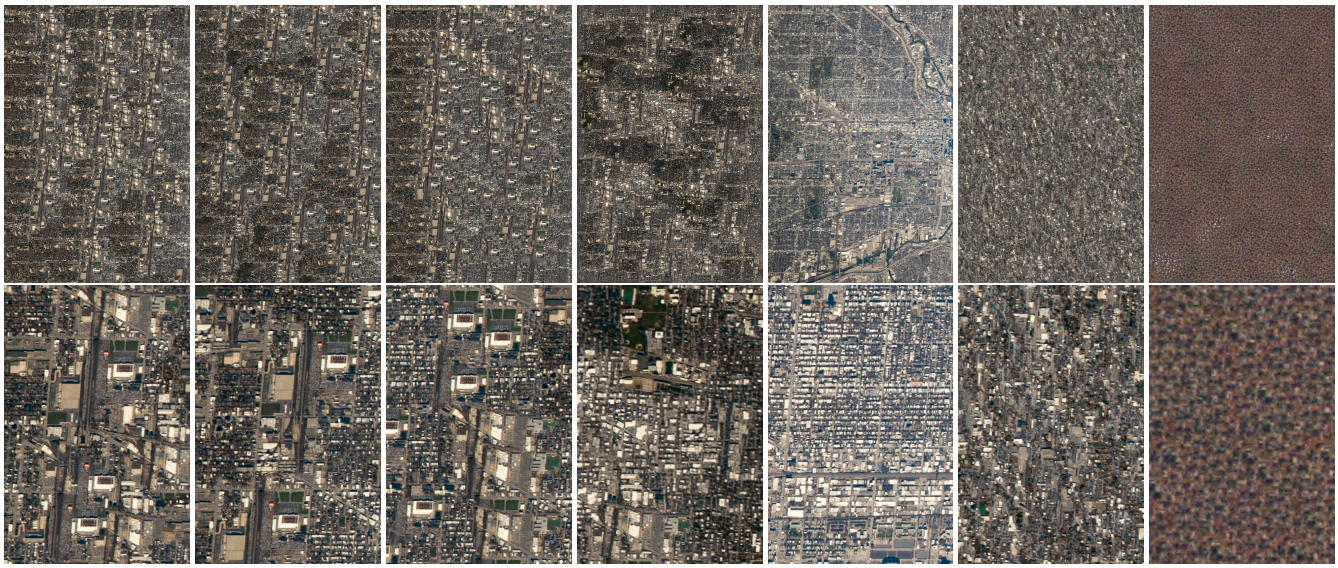
Low shape variance poses challenges which none of the algorithms handle well. Regular texture elements are not replicated perfectly (figure 5) in the output, and a human observer readily identifies even minute flaws, so the relative perceived quality is significantly worse than the reference.

The results demonstrate that numerous complex properties are well handled by the top four methods: texel size, example resolution, texels per sample, scale, scale variance, stochasticity, and shape. Furthermore, the orderings of table 4 show constructive disagreement among textures, demonstrating each offers novel information. Therefore, they must exhibit uncorrelated properties, validating the selection of section 3.2.

Considering the shared properties of textures which are not well synthesized, and the fact that reference images contain greater variance than the exemplar, the major focus of texture synthesis re-

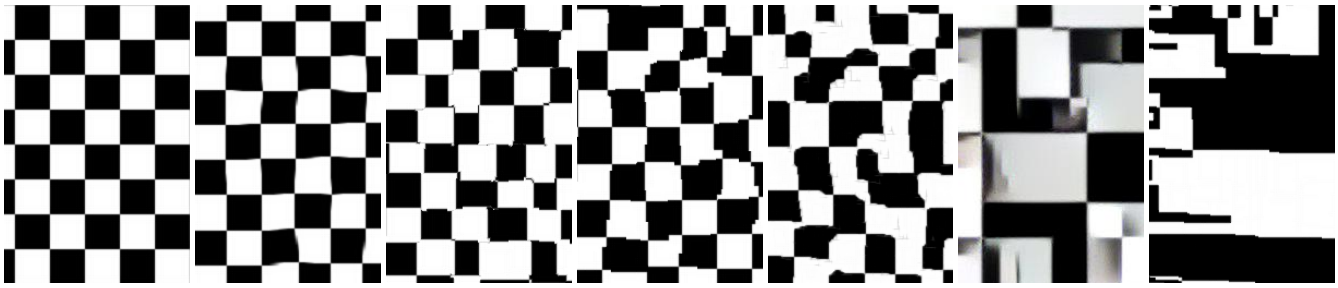
Texture	ranks							Kendall's W
smarties	reference	resynthesizer	quilting	self tuning	LazyFluids	Ashikhmin	CNNMRF	0.737
checkerboard	reference	self tuning	quilting	resynthesizer	LazyFluids	CNNMRF	Ashikhmin	0.819
blades	reference	resynthesizer	LazyFluids	self tuning	quilting	Ashikhmin	CNNMRF	0.44
grid	reference	quilting	self tuning	LazyFluids	resynthesizer	Ashikhmin	CNNMRF	0.811
pebbles	LazyFluids	reference	self tuning	resynthesizer	quilting	Ashikhmin	CNNMRF	0.461
ink	self tuning	LazyFluids	reference	resynthesizer	Ashikhmin	quilting	CNNMRF	0.558
rough	quilting	resynthesizer	self tuning	LazyFluids	reference	Ashikhmin	CNNMRF	0.447
chicago	resynthesizer	self tuning	quilting	LazyFluids	reference	Ashikhmin	CNNMRF	0.619
noise	reference	LazyFluids	quilting	resynthesizer	self tuning	Ashikhmin	CNNMRF	0.361
green marble	self tuning	LazyFluids	reference	resynthesizer	quilting	Ashikhmin	CNNMRF	0.517
orange marble	LazyFluids	self tuning	reference	resynthesizer	CNNMRF	Ashikhmin	quilting	0.266
straw	self tuning	quilting	resynthesizer	LazyFluids	reference	Ashikhmin	CNNMRF	0.363
ALL	reference	self tuning	resynthesizer	LazyFluids	quilting	Ashikhmin	CNNMRF	0.402

Table 4: Subjective ranks with Kendall W, for each texture separately, and over all textures. Ranks are from left to right. All orderings are significant to $p < 0.01$, except orderings within each group. Kendall's W coefficient of concordance ranges from 0 (no agreement) to 1 (complete agreement).



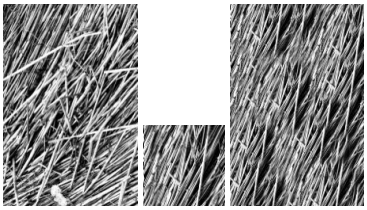
(a) resynthesizer (b) self-tuning (c) quilting (d) LazyFluids (e) reference (f) Ashikhmin (g) CNNMRF

Figure 4: The *chicago* texture synthesized with selected methods, ordered by user preference. Top row is the full synthesized texture as seen by participants, and the bottom row is a center crop of size $1/4 \times 1/4$ of the full output.



(a) reference (b) self-tuning (c) quilting (d) resynthesizer (e) LazyFluids (f) CNNMRF (g) Ashikhmin

Figure 5: The checkerboard texture, ordered by user preference



(a) reference (b) input (c) self-tuning

Figure 6: For this texture, the reference shows greater variance than the exemplar, which is not replicated in synthesis

search should be regularity mapping, learning to replicate the texture's generative process, and a structured approach to estimate variability to increase it in the output.

5.2. Analysis of Algorithm Performance

Overall, the study showed that people prefer the quality of the reference over any algorithm, conclusively demonstrating that none of the available algorithms in general produce convincing textures. Nonetheless, when ordering algorithms in relation to each other, **self tuning** is the best available method, and should be used in applications where quality is key.

Despite its simplicity, **resynthesizer** is indistinguishable from the far more computationally intensive **self tuning** on nine out of twelve textures, demonstrating that it is a powerful and flexible method, whose open-source implementation makes it ideal for portability.

LazyFluids appears best at generating novel textures of the scale of the exemplar, but does not perform well when generating a large sample. Figure 4 shows the full-scale vs the details of the *chicago* texture, revealing additional clues regarding method quality. Optimizing the quality metric of LazyFluids, which is an enhancement of Texture Energy [KSE*03], yield textures of comparatively low quality. Therefore, these quality metrics are not appropriate for quantitative quality evaluation.

Ashikhmin and **CNNMRF** have not performed as well as the other methods. Although these methods are known to produce impressive results, this demonstrates that their usefulness is limited to certain textures, and that they require careful parameter selection. Despite being the oldest tested method, **quilting** was consistently preferred over both of these, demonstrating the flexibility with which it achieves quality.

6. Conclusion

The quality of existing texture synthesis algorithms has been quantified by means of a subjective experiment, demonstrating that the problem of example-based texture synthesis is not yet fully solved. It appears there are types of textures and some texture properties where existing algorithms fail; although certain types of texture are found to be very well synthesized. The quality of previously published methods has been validated, and new findings regarding properties of textures which are not well synthesized have been made.

A set of textures which are representative of the desired properties have been proposed, with the hope that the performance of any method may be judged over all textures simply by observing results on these. Nine of these textures can be considered solved by the best algorithm, **self tuning**, and three show significant flaws with all state-of-the-art methods, allowing for compact and straightforward analysis of new methods.

A shortcoming of our final analysis is that the collapsed scores assume equal weight across the selected textures, but in reality this weighting is application-specific. We have attempted to allow application-specific analysis by showing rankings for each texture, but the overall ranking will not be representative of all applications. A limitation of the current work is that all participants were shown the same rendering, but independent renderings would have produced a more representative evaluation of methods.

By identifying properties of textures which have consistently not been synthesized as well as the reference, we hope to help advance research in the field of example-based texture synthesis, so that algorithms may be devised which handle any texture. Conclusions drawn regarding relationships between the texture properties and algorithm performance are informative, and hint to an area where additional studies may make statistically significant findings.

In future work, it will be beneficial to take into account user's preferences in application-specific environments, by performing the study with printed textures, ceramic tiles, or inside virtual envi-

ronments. The evaluation procedure with the proposed dataset requires human interaction, and it would be beneficial if quality assessment could be performed automatically. Finally, chosen properties and textures may need to be extended, either because future improvement is likely to solve the 12 proposed textures, or because certain applications may require texture properties not explored here, such as embossing or linear features. New textures may be found by first preparing a large dataset, then finding a representative subset.

Acknowledgment

The authors would like to thank the authors of compared methods for their code and cooperation. Debattista is partially supported by a Royal Society Industrial Fellowship (IF130053) and Kolář by the EPSRC (EP/I01585X/1).

References

[ADC13] ABDELMOUNAIME S., DONG-CHEN H.: New brodatz-based image databases for grayscale color and multiband texture analysis. *ISRN Machine Vision 2013* (2013). 1, 4

[Ash01] ASHIKHMIN M.: Synthesizing natural textures. In *Symposium on Interactive 3D graphics* (2001), ACM, pp. 217–226. 2

[Bal06] BALAS B. J.: Texture synthesis and perception: Using computational models to study texture representations in the human visual system. *Vision research 46*, 3 (2006), 299–309. 1

[BF14] BIANCONI F., FERNÁNDEZ A.: Appendix to texture databases-a comprehensive survey. *Pattern Recognition Letters 45* (2014). 1

[BFH*98] BUHMANN J. M., FELLNER D. W., HELD M., KETTERER J., PUZICHA J.: Dithered color quantization. *CGF 17*, 3 (1998). 2

[BLD*09] BANTERLE F., LEDDA P., DEBATTISTA K., BLOJ M., ARTUSI A., CHALMERS A.: A psychophysical evaluation of inverse tone mapping techniques. In *CGF* (2009), vol. 28. 2, 6

[Bro66] BRODATZ P.: *Textures: a photographic album for artists and designers*. Dover Pubns, 1966. 1, 4

[CR11] CORNET E., ROUQUIER J.-B.: Resynthesizer plugin for the gimp, 2011. 2

[cre14] CREW I. E. .: Image courtesy of the earth science and remote sensing unit, nasa johnson space center, 2014. 4

[DB97] DE BONET J. S.: Multiresolution sampling procedure for analysis and synthesis of texture images. In *Computer graphics and interactive techniques* (1997). 1

[EF01] EFROS A. A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. In *Computer graphics and interactive techniques* (2001), ACM. 2

[FH93] FELLNER D. W., HELMBERG C.: Robust rendering of general ellipses and elliptical arcs. *ACM TOG 12*, 3 (July 1993), 251–276. 2

[FvDF*93] FOLEY J. D., VAN DAM A., FEINER S. K., HUGHES J. F., PHILLIPS R.: *Introduction to Computer Graphics*. Addison-Wesley, 1993. 2

[GEB15] GATYS L. A., ECKER A. S., BETHGE M.: A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* (2015). 2

[Gra95] GRACZYK C.: Vistex vision texture database, 1995. 1

[Gwe14] GWET K. L.: *Handbook of inter-rater reliability: The definitive guide to measuring the extent of agreement among raters*. Advanced Analytics, LLC, 2014. 6

[Har05] HARRISON P.: *Image texture tools*. PhD thesis, PhD thesis, Monash University, 2005. 2

[HS13] HOSSAIN S., SERIKAWA S.: Texture databases–a comprehensive survey. *pattern recognition letters 34*, 15 (2013), 2007–2022. 1

[JFA*15] JAMRIŠKA O., FIŠER J., ASENTE P., LU J., SHECHTMAN E., SÝKORA D.: Lazyfluids: Appearance transfer for fluid animations. *ACM Transactions on Graphics 34*, 4 (2015). 1, 2

[KNL*15] KASPAR A., NEUBERT B., LISCHINSKI D., PAULY M., KOPF J.: Self tuning texture optimization. In *CGF* (2015), no. 2. 2, 3

[KSE*03] KWATRA V., SCHÖDL A., ESSA I., TURK G., BOBICK A.: Graphcut textures: image and video synthesis using graph cuts. In *ACM Transactions on Graphics (ToG)* (2003), vol. 22, ACM, pp. 277–286. 1, 2, 9

[KSS97] KOBBELT L., STAMMINGER M., SEIDEL H.-P.: Using subdivision on hierarchical data to reconstruct radiosity distribution. *CGF 16*, 3 (1997), C347–C355. 2

[LCTS05] LEDDA P., CHALMERS A., TROSCIANKO T., SEETZEN H.: Evaluation of tone mapping operators using a high dynamic range display. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, ACM, pp. 640–648. 2

[LFTG97] LAFORTUNE E. P., FOO S.-C., TORRANCE K. E., GREENBERG D. P.: Non-linear approximation of reflectance functions. In *Proc. SIGGRAPH '97* (1997), vol. 31, pp. 117–126. 2

[LHH*] LAI W.-S., HUANG J.-B., HU Z., AHUJA N., YANG M.-H.: A comparative study for single image blind deblurring. 2, 6

[LHW*04a] LIN W.-C., HAYS J., WU C., KWATRA V., LIU Y.: *A comparison study of four texture synthesis algorithms on near-regular textures*. Tech. Rep. CMU-RI-TR-04-01, 2004. 2

[LHW*04b] LIN W.-C., HAYS J., WU C., KWATRA V., LIU Y.: A comparison study of four texture synthesis algorithms on near-regular textures. In *ACM SIGGRAPH 2004 Posters* (2004), ACM, p. 16. 2

[LL05] LEE S., LIU Y.: Psu near-regular texture database. 1

[LLH04] LIU Y., LIN W.-C., HAYS J.: Near-regular texture analysis and manipulation. In *ACM Transactions on Graphics (TOG)* (2004), vol. 23, ACM, pp. 368–376. 2, 5

[Lou90] LOUS Y. L.: Report on the First Eurographics Workshop on Visualization in Scientific Computing. *Computer Graphics Forum 9*, 4 (Dec. 1990), 371–372. 2

[LW16] LI C., WAND M.: Combining markov random fields and convolutional neural networks for image synthesis. *arXiv preprint arXiv:1601.04589* (2016). 2, 3

[MDBR*16] MUKHERJEE R., DEBATTISTA K., BASHFORD-ROGERS T., VANGORP P., MANTIUK R., BESSA M., WATERFIELD B., CHALMERS A.: Objective and subjective evaluation of high dynamic range video compression. *Signal Processing: Image Communication 47* (2016), 426–437. 6

[PS00] PORTILLA J., SIMONCELLI E. P.: A parametric texture model based on joint statistics of complex wavelet coefficients. *International journal of computer vision 40*, 1 (2000), 49–70. 1

[RGSS10] RUBINSTEIN M., GUTIERREZ D., SORKINE O., SHAMIR A.: A comparative study of image retargeting. In *ACM transactions on graphics (TOG)* (2010), vol. 29, ACM, p. 160. 2, 6

[Sch12] SCHWARZKOPF H.: Public domain photograph, 2012. 4

[SLTM08] SMITH K., LANDES P.-E., THOLLOT J., MYSZKOWSKI K.: Apparent greyscale: A simple and fast conversion to perceptually accurate images and video. In *CGF* (2008), vol. 27. 5

[Č08] ČADÍK M.: Perceptual evaluation of color-to-grayscale image conversions. In *Computer Graphics Forum* (2008), vol. 27. 5

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *Computer graphics and interactive techniques* (2000). 2

[WLKT09] WEI L.-Y., LEFEBVRE S., KWATRA V., TURK G.: State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report* (2009). 1, 3