Rendering recognizably unique textures

Janne V. Kujala and Tuomas J. Lukka
Hyperstructure Group
Agora Center, P.O. Box 35
FIN-40014 University of Jyväskylä
Finland
jvk@iki.fi, lukka@iki.fi

Abstract

We present a perceptually designed hardware-accelerated algorithm for generating unique background textures for distinguishing documents. To be recognizable, the texture should produce a random feature vector in the brain after visual feature extraction.

Our motivating example is a hypertext user interface which shows a fragment of a link's target in the margin. Upon traversing the link, the fragment expands to fill the screen. Our goal is to avoid user disorientation by texturing each document with a unique background so that a document can easily be recognized from a fragment. The user should be able to learn the textures of the most often visited documents, as per Zipf's law.

The results of an initial experiment show that the generated textures are indeed recognizable. We discuss a method for enhancing text readability by both providing fast, interactive zooming and unnoticeably bleaching the background around text.

1. Introduction

In this article, we introduce the use of procedurally generated unique backgrounds as a visualization of document identity. In our approach, each document has a different, easily distinguishable background texture. The user can thus identify an item at a glance, even if only a *fragment* of the item is shown, without reading the title (which the fragment may not even show).

In the following sections, we first review related work on texturing. Next, we discuss an example user interface to a hypertext structure. Then, we formulate general principles for designing recognizable backgrounds and present our hardware-accelerated implementation. Following this, we discuss enhancing text readability on such backgrounds and practical experiences and conclude.

2. Related work

The *texture* of a surface, taken literally, is its translation-invariant statistical microstructure. In computer graphics, the word *texturing* is used in a somewhat looser sense[6, 16]: it refers to mapping 2D arrays of numerical values onto graphics primitives such as polygons or Beziér patches, modifying their rendered appearance in some way (coloring[6], bump mapping[3], etc.). Textures have been synthesized in several ways: procedurally[8, 22, 26, 27, 29], using other textures as a starting point[17], perceptually, for visualizing surface orientation[19, 34, 35] and scalar or vector fields[37], and statistically, as samples from a probability distribution on a random field[9, 12, 36]. As a particular example, the Starfish[32] program uses procedural texturing for generating random wallpaper textures.

Psychophysical studies on texture perception have mostly concentrated on pre-attentive *visual texture discrimination*[1, 20], the ability of human observers to effortlessly discriminate pairs of certain textures. Julesz[21] proposed that texture discrimination could be explained by the densities of textons, fundamental texture elements, such as elongated blobs, line terminators, and line crossings. However, the textons are hard to define formally.

Simpler, filtering-based models can explain texture discrimination equally well[2]. There is also physiological evidence of the filtering processes: in the visual cortex, there are cells sensitive to different frequencies, orientations, and locations in the visual field[5].

On a higher level, the correlations between local features are combined by forming contours and possibly other higher-level constructions[31]. These higher levels are not yet thoroughly understood; some theories[18] assume certain primitive shapes whose structure facilitates recognition.

There have also been studies on mapping texture appearance to an Euclidian texture space[14]: in these experiments, three dimensions have been sufficient to explain most of the perceived differences for artificial textures. However, the texture stimuli have been somewhat simple (no color, lack of frequency-band interaction, etc.). For some natural texture sets, three dimensions have also been sufficient[28], but semantic connections can make it hard to assess the dimensionality.

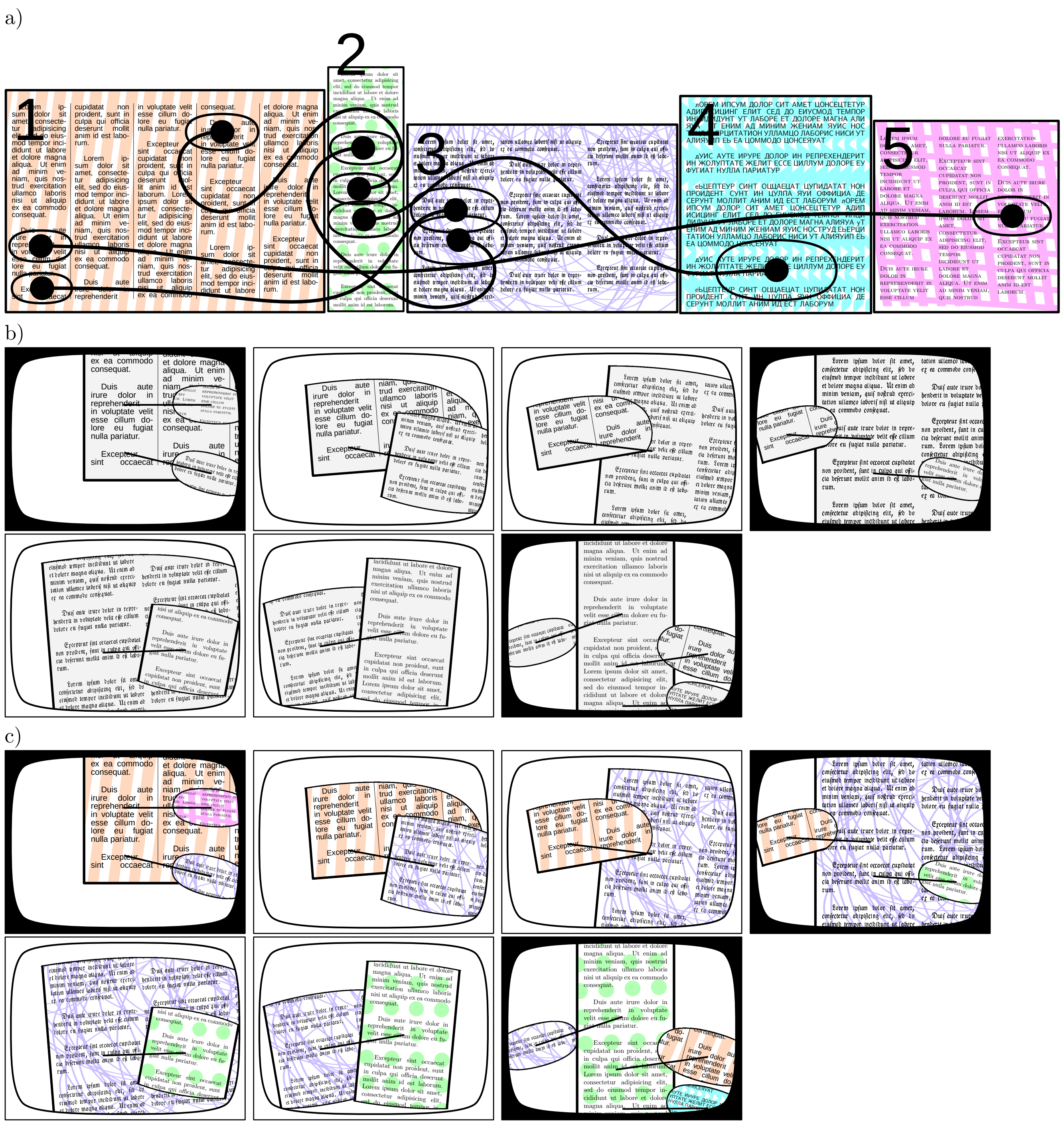


Figure 1. The motivating example for unique backgrounds: the BuoyOING focus+context interface for browsing bidirectionally hyperlinked documents. The interface shows the relevant fragments of the other ends of the links and animates them fluidly to the focus upon traversing the link. a) shows a small document network. b) and c) show what a user sees while browsing the network, b) without and c) with background texture. There are three keyframes where the animation stops. Two frames of each animation between the keyframes are shown. The unique backgrounds help the user notice that the upper right buoy in the last keyframe is actually a part of the same document (1) which was in the focus in the first keyframe. Our (as yet untested) hypothesis is that this will aid user orientation.

3. The motivation for Unique Backgrounds: the BuoyOING user interface

Focus+Context, or, fisheye views[10] are a paradigm for viewing large, structured information sets by showing the current area of interest (focus) magnified and the structurally connected but further-away elements peripherally, with less magnification. Much of the work on focus+context views has concentrated on tree structures or flat 2D images or maps.

The motivating example for unique backgrounds is the BuoyOING (Buoy-Oriented Interface, Next Generation) user interface, a focus+context interface for navigating hypertext. BuoyOING is a logical step from the earlier work on Fluid Links[39], hypercept animations[23], and transpointing windows[24]. BuoyOING has been designed from the ground up around the following three principles:

1. the user should always see all link targets ("you should see where you can go")
2. the link transition should be fluidly animated ("you should see where you do go")
3. the link transition and resulting view should make it obvious to the user how to go back, without an explicit back button ("once you get there, you should see how you can get back"). This implies bidirectional links.

The links are between specific areas of the 2D documents, and the *relevant fragments* of the target document is shown floating in the margin when the link anchor is close to the focus. Figure 1 shows a sample document structure and a traversal, with and without unique backgrounds.

4. Generating Unique Background Textures

In this section, we discuss the general principles we have used to derive our algorithm to generate unique background textures. The goals are that the unique backgrounds should be easily distinguishable and recognizable, and that they should not significantly impair the reading of black text drawn on top.

Our approach is an extension of the type of inversion approach that has been used by Ware and Knight[37] for inverting the earliest stage of the visual system in order to place a particular vector or scalar field of data in the texture "channel".

The ability to distinguish a particular texture from a large set depends on the distribution of textures in the set. For instance, it is intuitively clear that textures with independently random texel values would be a very bad choice: all such textures would look alike, being just noise. In order to design a distinguishable

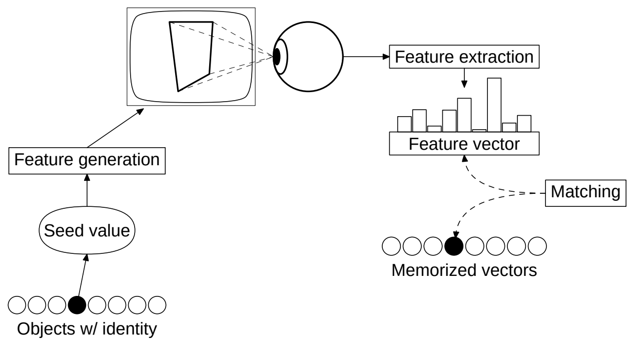


Figure 2. The qualitative model of visual perception used to create the algorithm. The visual input is transformed into a feature vector, which contains numbers (activation levels) corresponding to, e.g., colors, edges, curves and small patterns. The feature vector is matched against the memorized textures. In order to generate recognizable textures, random seed values should produce a distribution of feature vectors with maximum entropy.

distribution of textures, we have to take into account the properties of the human visual system.

The simple model of texture perception we use assumes that at some point, the results from the different pre-attentive feature detectors, such as different shapes and colors, are combined to form an abstract *feature vector* (see Fig. 2). As seen for example in [25], only a limited number of different features detected can be grouped into objects, indicating that the spatial resolution of the feature vector is quite low — as a well-known example, conjunction coding is not preattentive — red squares are hard to find among green squares and red and green circles.

The feature vector is then used to compute which concept the particular input corresponds to by comparing it to memorized models in a simple perceptron-like fashion[30, 38]. This configuration is commonly used in neural computation.

This rough, qualitative model is able to explain why uniformly random texels do not make easily distinguishable background textures: after the "preprocessing", different instances of noise would all yield *almost exactly the same feature vector* in the brain. Noise has no shape because there is no correlation between the pixels.

From the model we can see that to be distinguishable, a feature vector for a given texture should always be the same. Fragments of a non-repeating texture will be slightly different, resulting in slightly different vectors even if the local structure is the same. A repeating texture should thus be easier to recognize. Our anecdotal observations confirm this.

Additionally, the entropy of the feature vectors over the distribution of textures should be maximized. The distribution should contain occurrences of as many different features as possible, and the features should be distributed independently from each other. However, the results cited above[25] also indicate that in any *single* texture, only a limited range of features should be used.

In a sense, the model of perception should be *inverted* in order to produce a unique background from a random vector. Features that are orthogonal for human perception (e.g., color and the direction of the fastest luminance change) should be independently random, and features not orthogonal (e.g., colors of neighbouring pixels) should be correlated so as to maximize the entropy.

An important point in generating the backgrounds is that the texture appearance should have *no correlation* with any attribute or content of the document so that the textures of any hyperlinked documents are similar only by chance.

5. Hardware-accelerated implementation

In this section, we discuss our hardware-accelerated implementation (libpaper) of unique backgrounds (papers).

A. Parameter hierarchy

We have found that setting the parameters hierarchically produces the best results: the parameters for different passes should depend on hyperparameters randomly selected for the entire paper. This is in accordance with the discussion about [25] above: the hyperparameters limit the number of different features that are rendered onto one texture.

Our current parameter hierarchy is shown in Fig. 3. The individual parameters are explained in the subsections below.

B. Rationale for a Fragment-based implementation

One major goal for the implementation is to support complicated mappings between paper and screen coordinates, such as fisheye distortion. To make this simple, all processing when rendering the background texture must be done on the fragment level after the texture accesses, i.e., we cannot use procedural geometry except if pre-rendering the background into a texture.

However, pre-rendering each texture is likely to be too time- and memory-consuming if there are dozens of different documents visible at the same time, so we shall limit ourselves to pure fragment-based rendering in this article. However, if a single background covers

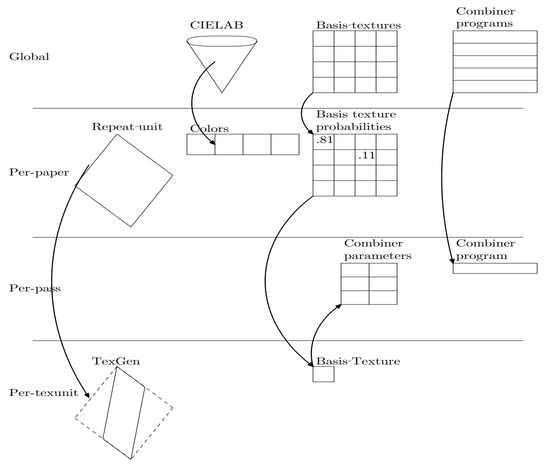


Figure 3. The parameter hierarchy for unique textures in our implementation. At the highest level, global hyperparameters define the distribution of the textures. For reasons explained in text, the parameters used by different rendering passes of the same background texture need to be correlated; this is achieved by having them depend on hyperparameters selected for each paper from the global distribution.

large areas of the screen, pre-rendering may increase performance considerably, and therefore our implementation does support two rendering modes with different tradeoffs. In the direct mode, small, static basis textures are used, which requires 2-3 passes with all texture units enabled. In the pre-rendered mode, rendering requires 1 pass with only one texture unit, but the repeating unit of the texture has to be pre-rendered into a texture of its own. To achieve a satisfactory image quality in zooming with pre-rendering, a relatively large texture has to be used for each background; 512x512 is not really sufficient.

Plain OpenGL 1.3 does not by itself provide enough flexibility in the fragment pipeline to allow for generating features nonlinearly from the basis textures. Because of this, and the availability of stable Linux drivers, our main platforms are NV10, i.e., OpenGL 1.3 + GL_NV_register_combiners, and NV25, i.e., NV10 + GL_NV_texture_shader3. We are working on an implementation based on GL_ARB_fragment_program once suitable hardware and Linux drivers emerge.

C. Colors

Color is the most dominant visual attribute of a texture. Therefore, it is essential that the overall colors of the backgrounds are maximally diverse with respect to color perception. However, we come again to the number-of-different-features arguments of Section 4: too many different colors in a *single* background are perceived just as a mix of many colors, making all

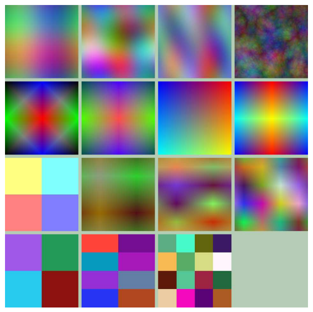


Figure 4. The complete set of 2D basis textures used by our implementation. All textures shown in this article are built from these textures and the corresponding HILO textures for offsetting.

such backgrounds look the same.

To maintain recognizability, we use a small palette of colors for each paper, selected randomly from a heuristic distribution. The final image contains convex combinations of the palette colors.

Even though the CIE $L^*a^*b^*$ [7] color space is fairly perceptually uniform, it is not good for selecting random colors (cf. [15]): using a uniformly chosen hue angle produces too much blue and too little yellow for high L^* (luminance). Because of this, we use a different color circle with seven evenly spaced "primary" hues (RGB, CMY and orange).

Hues are chosen from a distribution which usually clusters them fairly close together around a uniformly chosen mean but also allows far-away hues in the same palette occasionally. Because the shape of the RGB color gamut irregularly limits the saturations of light colors and linear interpolation can further reduce the saturation, the saturations of the palette colors are chosen from a distribution emphasizing saturated colors: unsaturated colors can easily cause a too multicolored palette because the adaptive effects of the eye shift them towards the complementary colors of the more saturated colors in the palette.

For readability, we only use colors with the CIE L^* value over 80. Note that the display gamma interacts with the lightness computation: we have adjusted the computations for a display gamma of 2.2, which is typically the default in PC systems. For each background, we use colors from *both* the dark and light end of the 80–100 range in order to create some contrast. This makes the shapes in the texture apparent, and avoids the unpleasant, blurry appearance of images with only chroma changes in colors.

D. Basis textures

The shapes of the final background texture are generated entirely from a small set of static *basis textures*. Even though the basis textures are RGB textures, they contain no color information: they are simply treated as 3- or 4-vectors to be used in various ways to create shapes, and color is added by the register combiners using the palette selected as described above.

We have obtained good results with the basis textures in Fig. 4: a mix of small textures with uniformly random texels, larger textures with noise or turbulence[27], and simple geometric images (e.g., checkerboard).

As for the selection of basis textures for each paper, the principle of limiting the number of different features in each texture applies here as well: we use hyperparameters for each background texture to control the probabilities of selecting different basis textures so that each paper will mostly have only basis textures two or three types.

E. Texture coordinates

The choice of the geometry of the repeating unit (a parallelogram) fixes an absolute scale for the paper. The repeating unit should be fairly isotropic to avoid the degeneration of textures to diagonal lines, and the units for different textures should be relatively similar in size. The repeating unit is chosen from a heuristic distribution satisfying these criteria.

After a repeating unit is fixed, there is still freedom in choosing texture coordinates for each basis texture: any mapping of the texture is fine, as long as it repeats with the selected repeating unit. For example, a texture can repeat multiple times inside the repeating unit, or can be skewed with respect to the repeating unit. Again, a heuristic distribution is used which does not skew or scale the basis texture too much too often.

F. Texture shading

On the NV25 architecture, the texture accesses can be customized further by the use of texture shading: the texture coordinates used by a texture unit can be made to depend on the result of a previous texture unit. This can be used to create a large variety of shapes[27]. So far, we have only used offset textures with random offset matrices, but even they do improve the quality of the output.

G. Register combiners

The NVIDIA register combiners extension is used to combine the the 3- and 4-vectors obtained from the basis textures and the palette colors into the final fragment color. Our need for the combiners is rather unconventional: we want to lose most of the original shapes of the basis textures in order to create new, different shapes from the interaction of the basis texture values

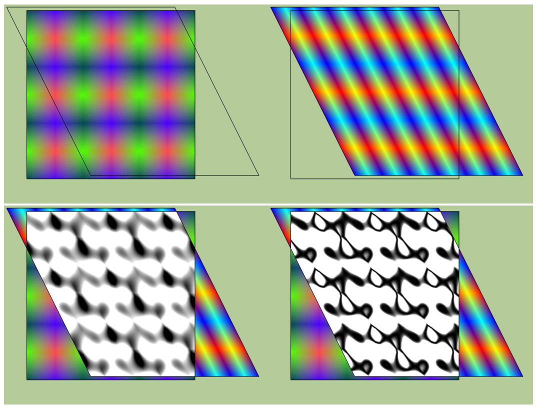


Figure 5. How the limited register combiners of the NV10 architecture can be used to generate shapes. Top: the two basis textures. Bottom left: dot product of the basis textures: $2(2a-1)\cdot(2b-1)+1/2$, where a and b are the texture RGB values. Bottom right: dot product of the basis textures squared: $32\left((2a-1)\cdot(2b-1)\right)^2$. This term can then be used to modulate between two colors.

and combiner parameters chosen randomly from the seed number. For this, we use dot products of texture values with each other and with random constant vectors, and scale up with the register combiner output mappings to sharpen the result (see Fig. 5). The resulting values are used for interpolating between the palette colors. Because some basis textures have blurrier edges than others, the output scalings need to be adjusted depending on the basis textures selected.

6. Experiences, evaluation

A. Overall appearance of the resulting textures

As seen in Fig. 7, the overall appearance of the textures is somewhat natural and the textures clearly do have unique distinguishing features.

They can be zoomed quite well, even though we have not tried to attain infinite zoomability[11] with the current implementation, but only the more modest goal of zooming within a range that would be reasonable for a single PDF document, i.e., approximately 20-fold difference between minimum and maximum zoom.

Although texture perception is scale-independent to some extent, the display resolution and certain spatial interactions in color perception limit the scale at which different features can be pre-attentively perceived. Therefore, we have chosen the texture coordinate mappings for different texture units so that the resulting textures have separate distinguishing features on both small and large scales.

It could be possible[11] to make the unique background look similar at different scales, but this would remove the use of the texture as a cue of scale.

Our nonlinear use of the register combiners does have some ill effects when zooming the texture out to a very small scale: mipmapping will not give the correct average color value. It may be possible to alleviate this by modeling the texture mathematically and calculating the correct average and placing corrective terms to the equations. However, in the intended zooming range the current system is quite satisfactory.

B. Text readability

The most common criticism against this work concerns text readability. Indeed, one of the most difficult aspects of this work *was* tuning the random color selection to produce acceptable results. However, in the current version, even relatively small text is quite legible on the backgrounds.

Experiments in [33] indicate that a background texture only affects readability when the text contrast is low. Also, as discussed there, subjective assessments of readability often have low correlation with objective measurements.

Additionally, text readability on the generated backgrounds depends strongly on the text scale; making it easy for the user to zoom fluidly in and out helps.

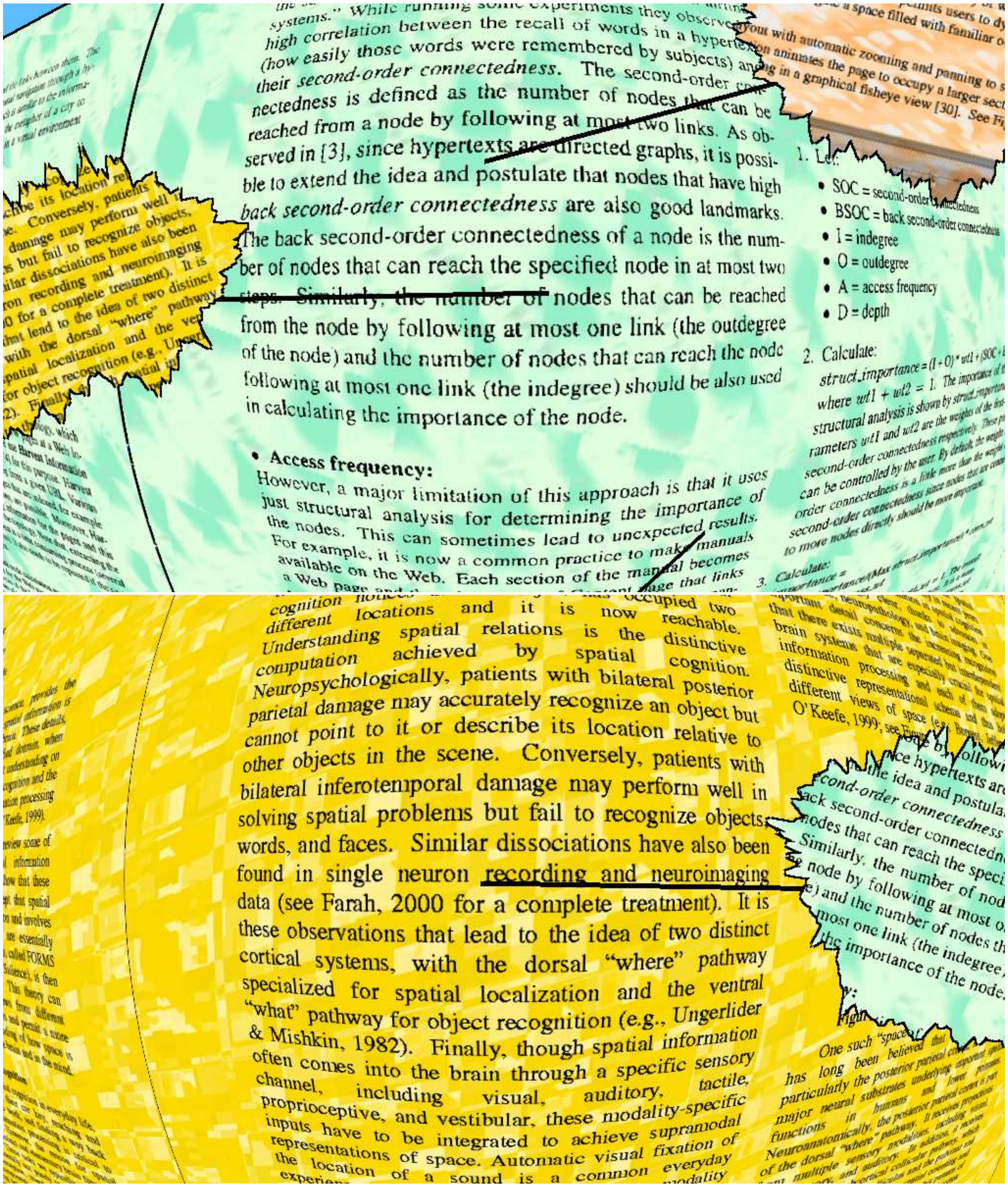
It is also possible to enhance the text readability explicitly, by lightening ("bleaching") the background slightly around the area where the text is drawn. The bleaching can be made almost unnoticeable on a computer screen and works well because we start from a fairly light background.

C. An Initial Recognizability Experiment

Experiments on black-and-white ink blots and snow crystals [13] have shown that complex, unfamiliar pictures can be remembered and recognized and that recognition performance decreases very little over time.

In order to evaluate the recognizability of our procedurally generated textures, we need to have an appropriate comparison point. Pictures of natural objects would not be appropriate, because they cannot be generated in infinite amounts from seed numbers and they would easily yield undesirable semantic associations. Lacking a better example, we shall use plain solid color backgrounds as a baseline even though the colors of even a small set of randomly chosen colors would most likely not be discriminable.

Another question is how many textures would the user have to remember for it to be useful. Studies of web cache statistics (see, e.g., [4]) have shown that file popularity approximately follows Zipf's law so that a small number of documents accounts for most of the



systems." While running some experiments they observed high correlation between the recall of words in a hypertext (how easily those words were remembered by subjects) and their *second-order connectedness*. The second-order connectedness is defined as the number of nodes that can be reached from a node by following at most two links. As observed in [3], since hypertexts are directed graphs, it is possible to extend the idea and postulate that nodes that have high *back second-order connectedness* are also good landmarks. The back second-order connectedness of a node is the number of nodes that can reach the specified node in at most two steps. Similarly, the number of nodes that can be reached from the node by following at most one link (the outdegree of the node) and the number of nodes that can reach the node following at most one link (the indegree) should be also used in calculating the importance of the node.

- **Access frequency:**

However, a major limitation of this approach is that it uses just structural analysis for determining the importance of the nodes. This can sometimes lead to unexpected results. For example, it is now a common practice to make manuals available on the Web. Each section of the manual becomes a Web page and ... that links

a space filled with familiar o... layout with automatic zooming and panning to s... animates the page to occupy a larger sect... in a graphical fisheye view [30]. See Fi...

1. Let:
   - SOC = second-order connectedness
   - BSOC = back second-order connectedness
   - I = indegree
   - O = outdegree
   - A = access frequency
   - D = depth
2. Calculate:
   struct_importance = (I + O) * *wt1* + (SOC + ...
   where *wt1* + *wt2* = 1. The importance of t... structural analysis is shown by *struct_importan*... rameters *wt1* and *wt2* are the weights of the first... second-order connectedness respectively. These p... can be controlled by the user. By default, the weight... order connectedness is a little more than the weight... second-order connectedness since nodes that are conn... to more nodes directly should be more important.
3. Calculate:
   importance = ...

cribe its location re... Conversely, patients ... damage may perform well ... s but fail to recognize objects, ... ilar dissociations have also been ... ron recording and neuroimaging ... 0 for a complete treatment). It is ... that lead to the idea of two distinct ... with the dorsal "where" pathway ... spatial localization and the ventral ... for object recognition (e.g., Unger... 2). Finally

cognition notices ... has occupied two different locations and it is now reachable. Understanding spatial relations is the distinctive computation achieved by spatial cognition. Neuropsychologically, patients with bilateral posterior parietal damage may accurately recognize an object but cannot point to it or describe its location relative to other objects in the scene. Conversely, patients with bilateral inferotemporal damage may perform well in solving spatial problems but fail to recognize objects, words, and faces. Similar dissociations have also been found in single neuron recording and neuroimaging data (see Farah, 2000 for a complete treatment). It is these observations that lead to the idea of two distinct cortical systems, with the dorsal "where" pathway specialized for spatial localization and the ventral "what" pathway for object recognition (e.g., Ungerlider & Mishkin, 1982). Finally, though spatial information often comes into the brain through a specific sensory channel, including visual, auditory, tactile, proprioceptive, and vestibular, these modality-specific inputs have to be integrated to achieve supramodal representations of space. Automatic visual fixation of the location of a sound is a common everyday experien... modality

portant detail concerns the neuropathology ... that there exists multiple separated but interdependent brain systems that are especially crucial for spatial information processing and each of them using distinctive representational schema and thus building different views of space (e.g., Burgess, Jeffery & O'Keefe, 1999; see Figure ...

...ce hypertexts ar... the idea and postula... cond-order connectedness ... ack second-order connectedn... nodes that can reach the speci... Similarly, the number of nod... node by following at most o... e) and the number of nodes th... most one link (the indegree, ... the importance of the node.

One such "space ... has long been believed that ... particularly the posterior parietal c... major neural substrates underlying important ... functions in humans and lower primates. Neuroanatomically, the posterior parietal cortex is part of the dorsal "where" pathway. It receives projections from multiple sensory modalities, including visual, ...

Figure 6. Two different screenshots of a structure of PDF documents viewed in a focus+context view. The user interface shows relationships between specific points in the documents. Each document has an unique background, which makes it easy to see that the fragment of a document on the right side of the bottom view is the document fully seen in the top view; without unique backgrounds, this would be relatively difficult and would require traversing the link.

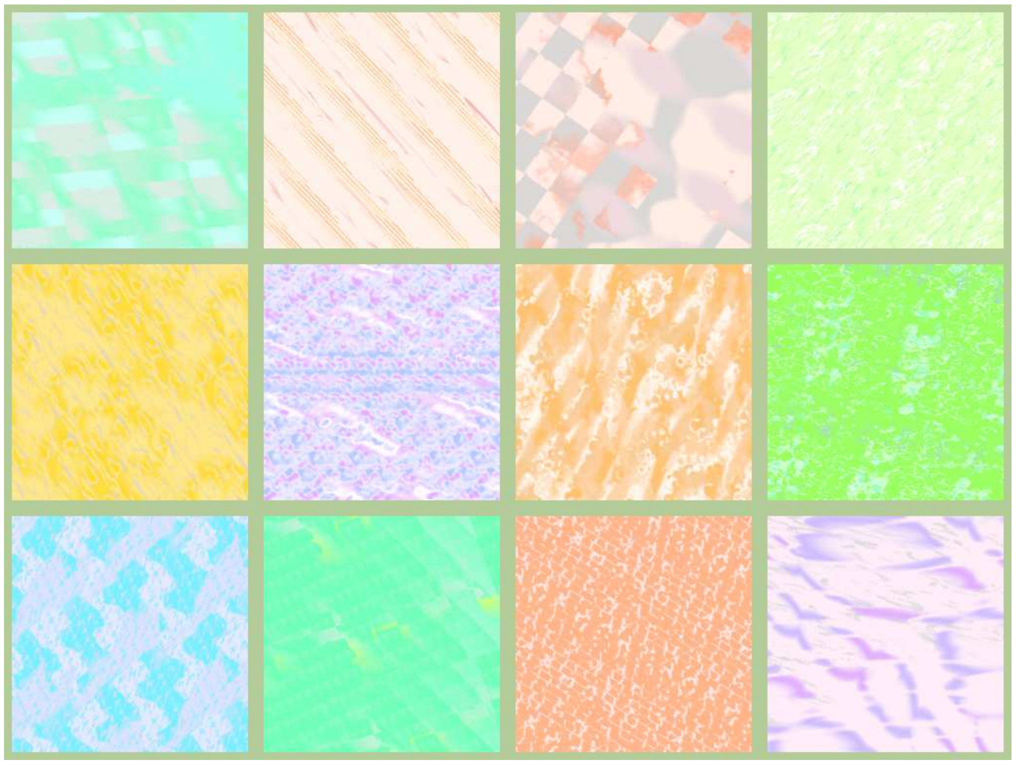


Figure 7. A number of unique backgrounds generated by our system. This view can be rendered, without pre-rendering the textures, in 20ms on a GeForce4 Ti 4200 in a 1024x768 window (fill-rate/bandwidth limited).

use, see Fig. 9.

Thus, we chose to measure the recognition of only 15 target textures or colors in the experiment. Our hypothesis is that the texture backgrounds are more recognizable than the solid colors.

C.1 Method

Participants. Five participants naïve to the hypothesis and our texturing work performed the experiment in both conditions.

Stimuli. 15 target backgrounds and 15 distractor backgrounds were randomly chosen for both the texture and solid color conditions. The distribution of the solid colors was the same that is used for the texture colors except that the highest lightness tail was de-emphasized to increase the otherwise low discriminability of very light, unsaturated colors.

Procedure. Each participant performed the test individually in both conditions in a random order. First, the 15 target backgrounds were shown sequentially, in a random order, 5 seconds each. Then, recognition was tested by showing the 15 target backgrounds and the 15 distractor backgrounds in a random order and having the participant answer "seen" or "not seen" for each one. The time for answers was not limited.

Table 1. Results of a recognition experiment with 15 previously seen textures and light colors to be picked out from 15 not previously seen instances. The numbers give the percentage of trials and the average reaction time in seconds for the five subjects. This shows that the textures are quite recognizable even after just one previous viewing.

	TEXTURES		COLORS	
	correct	incorrect	correct	incorrect
seen	72 (2.6s)	28 (5.5s)	71 (2.8s)	29 (4.1s)
not seen	84 (3.0s)	16 (5.6s)	43 (3.4s)	57 (4.0s)
total	78 (2.8s)	22 (5.5s)	57 (3.1s)	43 (4.1s)

C.2 Results

The results are summarized in Table 1. An analysis of variance indicates that the average recognition performance is significantly better for the textures than for the solid colors [F(1,4) = 19.0, p = .012]. The large number of false positives shows that solid colors do not have enough variation for unambiguous recognition.

Of course, solid colors could facilitate recognition in the presence of other cues, such as the text of the document. However, the recognition of similar-looking fragments of documents would still depend on conscious effort on the user's part — exactly what the unique background textures were designed to avoid.

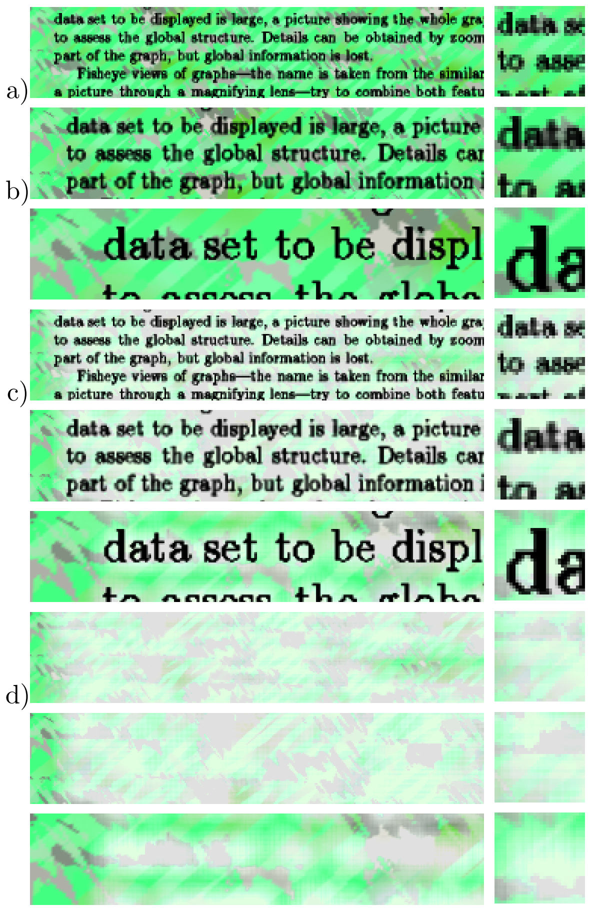


Figure 8. Enhancing text readability on colored backgrounds: a) Original, b) Zooming, c) Bleaching, d) The bleached version without drawing the black text, showing the blurred lightening.

7. Conclusions

We have presented a perceptually designed hardware-accelerated algorithm for generating recognizably unique backgrounds. The motivating example, the BuoyOING user interface demonstrates that the method is at its most useful when the same document can be reached through several ways and fragments of documents are seen.

Of course, we cannot hope to match in quality a unique graphical appearance designed by a human designer; magazines and web sites have long used skillfully designed graphical elements to make themselves recognizable. However, our algorithm is able to generate an unlimited amount of unique backgrounds cheaply, making it possible, e.g., for all academic articles with similar typography to have their own background.

So far, we have concentrated mostly on low-end hardware, and have not even tapped the full potential of the NV25 architecture.

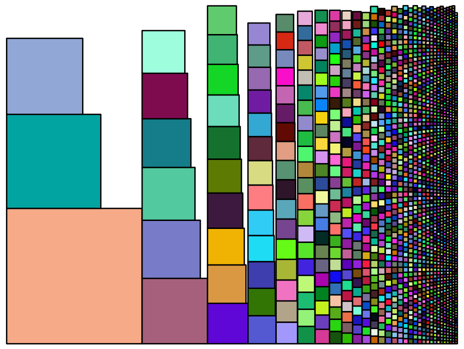


Figure 9. Zipf's law concretized: why distinguishing 15 textures from a large number of others helps. In real life, some documents get accessed far more often than most. The diagram shows 2000 documents weighted with Zipf's law with exponent 1.1. Each square represents a document, and the area of each square is scaled to its rate of accesses. Under these conditions, the 15 most important documents account for approximately half of the accesses. Of course, it is impossible to know which documents will be important and that will also change with time so all documents should be textured stably from the first viewing onwards.

We have conducted an initial recognition experiment showing that the textures generated by our algorithm are indeed recognizable. Carrying out more usability tests is necessary, both to measure how well textures can be remembered and to make the ad hoc distributions more experimentally based.

It can of course be argued that the backgrounds clutter the display visually, making the user interface more confusing, and may not even be helpful. Indeed, many aspects of the system are a compromise between recognizability and other goals such as readability, rendering performance, or user comfortability.

Overall, however, we feel that the backgrounds can greatly improve user orientation, enabling more efficient views to hyperstructured content, and therefore that the benefits can be made to outweigh the costs.

8. Acknowledgments

The authors would like to thank Marketta Niemelä for help with ANOVA, and John Canny, Benja Fallenstein, Marketta Niemelä and Pertti Saariluoma for discussions. Benja Fallenstein, Matti Katila, and Asko Soukka have contributed to the development of the BuoyOING interface in aspects not related to the unique background textures.

References

[1] J. R. Bergen. Theories of visual texture perception. In D. Regan, editor, *Vision and Visual Dysfunction,*

volume 10B, pages 114–134. Macmillian, New York, 1991.

[2] J. R. Bergen and E. H. Adelson. Early vision and texture perception. *Nature*, pages 363–364, May 1988.

[3] J. F. Blinn. Simulation of wrinkled surfaces. In *Proc. SIGGRAPH'78*, pages 286–292. ACM Press, 1978.

[4] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. In *Infocom'99 proceedings*, pages 126–134, 1999.

[5] V. Bruce, P. R. Green, and M. A. Georgeson. *Visual Perception: Physiology, Psychology, and Ecology, 3rd edition*. Psychology Press, 1996.

[6] E. E. Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, Dept. of CS, U. of Utah, Dec. 1974.

[7] Colorimetry. CIE Publication No. 15.2, 2nd Edition, 1986.

[8] R. L. Cook. Shade trees. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 223–231, 1984.

[9] G. C. Cross and A. K. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1):25–39, 1983.

[10] G. W. Furnas. Generalized fisheye views. In *Proceedings of CHI '86*, pages 16–23. ACM Press, 1986.

[11] G. W. Furnas and X. Zhang. Illusions of infinity: feedback for infinite worlds. In *Proc. UIST'00*, pages 237–238. ACM Press, 2000.

[12] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–742, 1984.

[13] A. G. Goldstein and J. E. Chance. Visual recognition memory for complex configurations. *Perception and Psychophysics*, 9(2B):237–241, 71.

[14] R. Gurnsey and D. J. Fleet. Texture space. *Vision Research*, 41:745–757, 2001.

[15] C. G. Healey. Choosing effective colours for data visualization. In *Proc. IEEE Visualization '96*, pages 263–270, 1996.

[16] P. S. Heckbert. Survey of texture mapping. In M. Green, editor, *Proceedings of Graphics Interface '86*, pages 207–212, 1986.

[17] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proc. SIGGRAPH'95*, pages 229–238. ACM Press, 1995.

[18] B. I. Recognition–by–components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[19] V. Interrante. Illustrating surface shape in volume data via principal direction-driven 3d line integral convolution. In *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 109–116, 1997.

[20] B. Julesz. Visual pattern discrimination. *IRE Transactions on Information Theory*, 8(2):84–92, 1962.

[21] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, Mar 1981.

[22] E. M. H. Ken Perlin. Hypertexture. *Computer Graphics*, 23(3):253–262, 1989.

[23] D. Milgram and W. B. Cowan. Hypercept: behavioural linkage in hypertext environments. In *Proc. NPIVM'99*, pages 49–55. ACM Press, 1999.

[24] T. H. Nelson. Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Computing Surveys*, 31(4es), 1999.

[25] I. R. Olson and Y. Jiang. Is visual short-term memory object based? rejection of the "strong-object" hypothesis. *Perception & Psychophysics*, 64(7):1055–1067, 2002.

[26] D. R. Peachey. Solid texturing of complex surfaces. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 279–286. ACM Press, 1985.

[27] K. Perlin. An image synthesizer. *Computer Graphics (SIGGRAPH '85 Proceedings)*, 19(3):287–296, 1985.

[28] A. R. Rao and G. L. Lohse. Towards a texture naming system: Identifying relevant dimensions of texture. *Vision Research*, 36:1649–1669, Jun 1996.

[29] J. Rhoades, G. Turk, A. Bell, A. State, U. Neumann, and A. Varshney. Real-time procedural textures. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 95–100. ACM Press, 1992.

[30] F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanics*. Washington, DC: Spartan, 1962.

[31] J. Saarinen, D. M. Levi, and B. Shen. Integration of local pattern elements into a global shape in human vision. *Proceedings of the National Academy of Sciences U.S.A.*, 94:8267–8271, Jul 1997.

[32] M. Saxman. The Starfish program. http://www.redplanetsw.com/starfi sh/.

[33] L. F. V. Scharff, A. L. Hill, and A. J. Ahumada, Jr. Discriminability measures for predicting readability of text on textured backgrounds. *Optics Express*, 6(4):81–91, 2000.

[34] D. Schweitzer. Artificial texturing: An aid to surface visualization. In *Proc. SIGGRAPH'83*, pages 23–29, 1983.

[35] D. Stalling. LIC on surfaces. In *Texture Synthesis with Line Integral Convolution*, pages 51–64, 1997.

[36] G. Turk. Generating textures on arbitrary surfaces using reaction-diffusion. In *Proc. SIGGRAPH'91*, pages 289–298. ACM Press, 1991.

[37] C. Ware and W. Knight. Using visual texture for information display. *ACM Transactions on Graphics (TOG)*, 14(1):3–20, 1995.

[38] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *IRE WESCON Convention Record*, volume 4, pages 96–104, 1960.

[39] P. T. Zellweger, B.-W. Chang, and J. D. Mackinlay. Fluid links for informed and incremental link transitions. In *Proc. HyperText'98*, pages 50–57. ACM Press, 1998.