# Stabilized Noise

Jong-Chul Yoon [†] and In-Kwon Lee [‡]

Dept. of Computer Science, Yonsei University

**Abstract**

*Perlin noise is generated by interpolation of a pre-defined random number table. However since the random number table is generated without considering the ideal properties of noise for computer graphics, the resulting noise function does not have these properties. We improve the properties of noise for Perlin's and similar algorithms, by stabilizing the random number table itself. We create the noise function using well-known statistical tools that measure the degree of stability of a random number table. These tools are used within an optimization procedure to create a random number table with a uniform random distribution, without periodicity, and having a band-limited property.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: Color, shading, shadowing, and texture

## 1. Introduction

Perlin's noise function [Per84, Per85], is now a well-established tool for creating procedural texture and shading. Although Perlin noise is defined as a continuous series of values which can be generated from random numbers, it is different from white noise, which consists of unconstrained random numbers. Computer graphics applications require noise with ideal properties: meaning that it is reproducible, bounded, band-limited, non-periodic, stationary and isotropic periodicities [EMP*02].

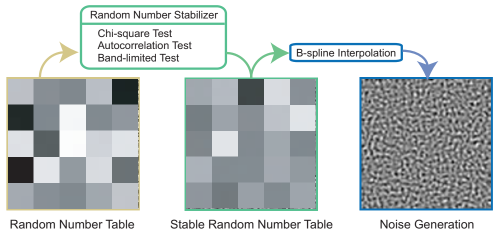

**Figure 1:** *Overview of our approach to generating stable noise.*

Perlin's original noise function [Per85] does not exhibit these ideal properties fully. There has been research on improving the quality of noise function, for instance by eliminating bias [Per02] and by making the noise fully band-limited property [CD05]. However, each of these improvements only considers an individual property. Furthermore, because of its simplicity of implementation, the original Perlin noise function still in general use, and has even implemented in hardware [Har01].

In this paper, we propose a novel way to generate a stable random number table which can be used to produce an stabilized noise function. This leads to a noise function with ideal properties, which can nevertheless be generated using Perlin's original straightforward mechanism. Conventional methods of random number generation are based on simple arithmetic. But these methods take no account of the properties of an ideal noise function, making it the weak point in Perlin's procedure. But we deliberately generate a random number table with the required properties. Various statistical tools are commonly used to test the properties of random number [LK91]. From these, we select three tests. The chi-squared and auto-correlation tests, measure the uniformity of a random distribution and the periodic degree of a table of random numbers, while procedural band-pass pyramid measures its band-limited degree. Based on these three tools, we can design an optimization problem, the solution of which stabilizes a random number table so that it can be used to

[†] media19@cs.yonsei.ac.kr
[‡] iklee@yonsei.ac.kr

create an ideal noise function. A stabilized table is reusable and can also be employed with other noise generation technique.

Figure 1 is an overview of our noise generation method. The noise function that we use is a simple B-spline interpolation, but our random number table can be generated by any other noise mechanism which is based on predefined random numbers.

## 2. Related Work

After Perlin [Per84, Per85] introduced the noise function, it was used to generate procedural textures [EMP*02, Lew89], various natural phenomena such as water, fire, clouds, woodgrain etc. [PN01], and to make animated motions more natural [Per95]. Additionally, a noise function can provide a way of reducing the high cost of simulation-based methods [EMP*02]. Another approach to natural pattern generation was introduced by Lewis [Lew89]. His method, which is based on the Wiener interpolation, can generate noise with an arbitrary energy spectrum, but it is more complex than Perlin's method. Recently, Perlin noise has even been implemented in hardware for real-time graphics [Har01]. Ebert et al. [EMP*02] have dealt with various aspects of noise and provide details of many different types of implementation of the noise function.

Perlin's original noise function has been improved to overcome two kinds of problem. Perlin himself [Per02] introduced a new algorithm that solved two problems with his noise function. He suggested a new interpolation method that gives $C^2$ continuity everywhere in the domain, and a new gradient table, containing only 16 vectors, which speeds up computation and prevents directional bias. Cook et al. [CD05] introduced Wavelet noise, which perfectly band-limited because unnecessary low frequencies are eliminated. Our method of generating an stable random number table inherits these improvements. We inherit these improvements, in generating an ideal random number table which can be used with any noise function.

## 3. Random Number Stabilization

We generate a table of random numbers using an objective function which measures the ideal properties of the noise function numerically. This objective function is divided into three terms: a chi-squared test term ($CS(x)$), an autocorrelation term ($AC(x)$), and a band-limited term ($BL(x)$).

### 3.1. Chi-squared Test Term

The chi-squared test [Knu98] is used to compare observed data with a specific distribution. Let $X$ be a set of observed data. We divide the domain of $X$ into $K$ buckets of uniform size, so that the number of data items in the $j^{\text{th}}$ bucket is $m_j$.

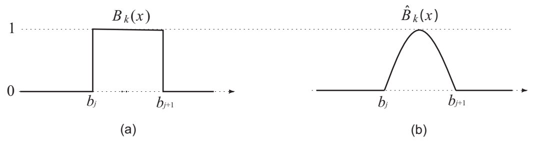

**Figure 2:** *(a) $B_k(h)$ is the boxcar function, and (b) $\hat{B}_k(h)$ is an approximation to the boxcar function.*

If $Z_j$ is the expected number of data items in the $j^{\text{th}}$ bucket, then the chi-squared value $D^2$ can be computed as follows:

$$D^2 = \sum_{j=1}^{K} \frac{(m_j - Z_j)^2}{Z_j}. \tag{1}$$

If the distribution of the observed data $X$ is similar to its expected distribution $Z_j$, then the value of $D^2$ will tend to zero.

If a sequence of random numbers has a stable distribution, the data are equally spread over the whole domain [LK91]. So, if we use the uniform distribution as the expected distribution in the chi-squared test, a small value of $D^2$ represents a stable random distribution. Using the boxcar function to count the number of data items in each bucket [YLC04], we can express the uniformity of the random distribution as a chi-squared value.

Let $N$ be the number of data items and let $N/K$ be the uniform expected value for all buckets. Then we can calculate $CS(X)$ as follows:

$$CS(X) = \sum_{j=1}^{K} \left( \sum_{i=1}^{N} B_j(x_i) - \frac{K}{N} \right)^2, \tag{2}$$

$$B_j(x) = \begin{cases} 1, & \text{if } b_j \le x \le b_{j+1}, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where $x_i$ denotes each random number in table $X$, and $B_j(x)$ is a boxcar function that measures whether data value $x$ is located between $b_j$ and $b_{j+1}$, which is the range covered by the $k^{\text{th}}$ bucket (see Figure 2(a)). Therefore $\sum_{i=1}^{N} B_j(x_i)$ is the number of data items in the $j^{\text{th}}$ bucket. To determine the number of buckets, $K$, we follow the rule of thumb [Rey84] that there should be at least five data items in any bucket.

In Equation (2), a low value of $CS(X)$ means that the distribution of $X$ is similar to the uniform random distribution. However because the boxcar function is an integer block that is appropriate for an interger problem, we approximated it by the piecewise function shown in Figure 2(b), which is a simple quadratic curve.

### 3.2. Autocorrelation Test Term

Autocorrelation is a widely used mathematical tool to analyze the periodic offset of signals [PM96] in one or many dimensions. Generally, autocorrelation is the cross-correlation of a signal with a lagging version of itself. Let $\vec{k}$ be the offset vector of the lagged signal. A higher autocorrelation value is likely to indicate a periodicity in the signal with the period $\vec{k}$. Assuming that $x_i$ is a data sequence of length $N$, we can calculate an autocorrelation value $autoc(\vec{k})$ for an offset $\vec{k}$ as follows:

$$autoc(\vec{k}) = \frac{\sum_{i=1}^{N}(x_i-\mu)(x_{i+\vec{k}}-\mu)}{\sum_{i=1}^{N}(x_i-\mu)^2}, \tag{4}$$

where $\mu$ is the mean value of $x_i$ and $autoc(\vec{k})$ has the range $[-1,1]$. Because the ideal noise function should be non-periodic, a stable random number table, from which the noise function will be generated, should also be non-periodic. Therefore, a stable random number table must have a low autocorrelation for any offset $\vec{k}$. We have designed $AC(X)$ to be a periodic measure of a random number table: it is the sum of the absolute autocorrelation value at all offset vectors. $AC(X)$ is formulated as follows:

$$AC(X) = \sum_{\vec{k}\in\Theta} ||autoc(\vec{k})||, \tag{5}$$

where $\Theta$ is whole domain of $X$.

### 3.3. Band-limited Test Term

The Perlin noise function constructs a pattern within an assigned band, and this means that the noise used must have a limited range of frequencies. To generate a natural pattern which contains a range of frequencies, a noise function can be constructed by summing a multi-band noise such as fractal Browning motion (FBM) or turbulence [EMP*02]. To generate a stable multi-band noise function requires a noise function that is completely band-limited. However, the original version of Perlin noise contains unnecessary low frequencies [CD05], which can cause an aliasing problem in details of the resulting pattern. The low frequencies can also impair amplitude control of multi-band noise. Although Perlin improved his noise function [Per02] by using a fixed surface normal to improve the band-limited property, this turns out to produce only a weak form of band-limitation. To fix the problem of band-limitation, Cook et al. [CD05] introduced Wavelet noise, which eliminates the low frequencies using a procedural band-pass pyramid. We use the same procedural band-pass pyramid to measure band-limitation in random number table.

To measure the unnecessary low frequencies in a given random number table $X$ (see Figure 3(a)), we apply down-sampling to the upper-level pyramid. Let $X^{\downarrow}$ be the result of down-sampling. The term $X^{\downarrow}$, which is half the size of $X$, contains the low frequencies. Then we perform up-sampling to expand the pyramid again. Finally we obtain $X^{\downarrow\uparrow}$ (Figure 3(b)). Cook et al. [CD05] generate Wavelet noise from $(X - X^{\downarrow\uparrow})$ (Figure 3(c)). Since we have to design an objective function which measures the band-limited term, we need to formulate $X^{\downarrow\uparrow}$ numerically. If $X$ has a perfectly band-limited form, all values in $X^{\downarrow\uparrow}$ will be zero. We therefore measure the band-limited property of a random number table as the magnitude of $X^{\downarrow\uparrow}$. The value of $BL(X)$, which tests the band-limited property, is formulated as follows:

$$BL(X) = ||X^{\downarrow\uparrow}||, \tag{6}$$

where $||.||$ represents the sum of the squared values of each element in the matrix $X^{\downarrow\uparrow}$.

| | $CS(X)$ | $AC(X)$ | BL(X) |
|---|---|---|---|
| Before | 102.5153 | 12.2456 | 5.7672 |
| After | 23.0342 | 3.7643 | 0 |

(a) Seperate optimization

| | $CS(X)$ | $AC(X)$ | BL(X) |
|---|---|---|---|
| Before | 102.5153 | 12.2456 | 5.7672 |
| After | 25.0466 | 4.6672 | 0 |

(b) Composite optimization

**Table 1:** *Optimization convergence: (a) changed values resulting from optimizing three terms separately; (b) changed values resulting from optimizing three terms with a weighted summation. - average values for 100 random number table generations*

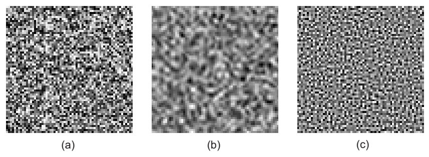

**Figure 3:** *Elimination of low frequencies using the procedural band-pass pyramid: (a) PRN table; (b) $X^{\downarrow\uparrow}$ is a down-up sampling of (a); (c) a new random number table desired from (a) without the low frequencies in (b).*

### 3.4. Random Table Generation by Optimization

Using the three tools which have been introduced in the foregoing sections, we can now design an objective function that corresponds to a stable random number table. A lower value of each test term, $CS(X)$, $AC(X)$ and $BL(X)$, represents a more stable condition. Therefore, representing each random number in $X$ as a unknown variable, we can design the objective function as follows:

$$\text{minimize } \; w_1\, CS(X) + w_2\, AC(X) + w_3\, BL(X), \tag{7}$$

where $w_i$ represents the weight of each term. (We used 0.1, 1 and 2 as the weights for normalization of each term of the function value: see Table 1.) We used the interior-reflective Newton method [CL94] to solve the optimization problem.

| Dimension | Resolution | time |
|---|---|---|
| 1 | 128 | 12 sec. |
| 2 | $128 \times 128$ | 243 sec. |
| 3 | $128 \times 128 \times 128$ | 183 min. |

**Table 2:** *Optimization costs for noise generation. - average values for 20 random number table generations*

## 4. Result

We experimented with the optimization, observing the rate of convergence when the three terms were treated separately and together. Because Equation (7) consists of a weighted sum of three different terms, If the independence between each term is not guaranteed, a conflict can occur which interrupts the search for an optimal solution. We found that each of the three terms is almost independent of the other two (see Table 1). Figure 4 shows the change in autocorrelation values and energy in the frequency domain during stabilization of a two-dimensional random number table. Before the optimization, the random number table contained unstable periodic points (see the red points in Figure 4(a)) and unnecessary low frequencies (see Figure 4(c)). The optimization eliminated these undesirable properties (see Figure 4(b) and (d)).

Table 2 shows the computation times for these experiments. The computational environment was an Intel Core2 CPU running at 2.13 GHz, with 2 Gb of memory.

## 5. Conclusion

We have introduced a noise generation method with the ideal property of the resulting noise function. Using statistical tools which are traditionally applied to the testing of random numbers, we generated a stable random number table as a resource for Perlin's and other widely used noise functions.

One limitation of our work is the length of the optimization time. Because the number of unknown variables involved in constructing a random number table is huge, the optimization is time-consuming. The optimization time is very much affected by the statistical tools used to achieve the ideal properties of the noise function. Other statistical methods might be more effective, and we will investigate new ways of generating noise patterns in many aspects.

## Acknowledgements

This research was supported by University IT Research Center Project(ITRC).

## References

[CD05] COOK R. L., DEROSE T.: Wavelet noise. In *Proceedings of ACM SIGGRAPH '05* (2005), pp. 803–811.

[CL94] COLEMAN T. F., LI Y.: On the convergence of interior-reflective newton methods for nonlinear minimization subject to bounds. *Mathematical Programming 67*, 1-3 (1994), 189–224.

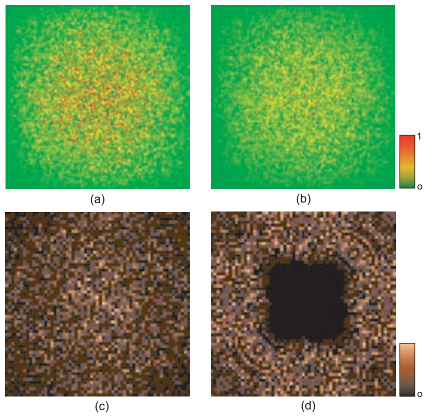

**Figure 4:** *Result of the optimization to stabilize the random number table. The autocorrelation value is stabilized from (a) to (b). The transformation from (c) to (d) shows the elimination of unwanted low frequencies.*

[EMP*02] EBERT D., MUSGRAVE F. K., PEACHEY D., PERLIN. K., WORLEY S., MARK B., HART J.: *Texture & Modeling: A Procedural Approach*, 3rd ed. Morgan Kaufmann, 2002.

[Har01] HART J. C.: Perlin noise pixel shaders. In *Proceedings of Graphics Hardware 2001: Eurographics/SIGGRAPH Workshop* (2001), pp. 87–94.

[Knu98] KNUTH D. E.: *The Art of Computer Programming*. Addison-Wesley, 1998.

[Lew89] LEWIS J. P.: Algorithms for solid noise synthesis. In *Proceedings of ACM SIGGRAPH '89* (1989), pp. 263–270.

[LK91] LAW A. M., KELTON W. D.: *Simulation Modeling & Analysis*, 3rd ed. McGraw-Hill, 1991.

[Per84] PERLIN K.: A unified texture/reflectance model. In *SIGGRAPH '84 Advanced Image Synthesis Course Notes* (1984).

[Per85] PERLIN K.: An image synthesizer. In *SIGGRAPH '85 Proceedings* (1985), pp. 287–296.

[Per95] PERLIN K.: Realtime responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics 1*, 1 (1995), 5–15.

[Per02] PERLIN K.: Improving noise. In *SIGGRAPH '02 Proceedings* (2002), pp. 681–682.

[PM96] PROAKIS J. G., MANOLAKIS. D. G.: *Digital Signal Processing: Principles, Algorithms, and Applications*, 3rd ed. Cambridge University Press, 1996.

[PN01] PERLIN K., NEYRET F.: Flow noise. In *SIGGRAPH '01 Technical Sketches and Applications* (2001), p. 187.

[Rey84] REYNOLDS. H. T.: *Analysis of Nominal Data*. Sage Publications, 1984.

[YLC04] YOON J. C., LEE I. K., CHOI J. J.: Editing noise. *Computer Animation and Virtual Worlds 15*, 3 (2004), 277–287.