Perception, 2004, volume 33, pages 837 – 854

DOI:10.1068/p5141

# Creating noisy stimuli

Flip Phillips
Department of Psychology, Skidmore College, Saratoga Springs, NY 12866-1632, USA;
e-mail: flip@skidmore.edu
Received 15 October 2001, in revised form 25 March 2003

**Abstract.** A method for creating a variety of pseudo-random 'noisy' stimuli that possess several useful statistical and phenomenal features for psychophysical experimentation is outlined. These stimuli are derived from a pseudo-periodic function known as *multidimensional noise*. This class of function has the desirable property that it is periodic, defined on a fixed domain, is roughly symmetric, and is stochastic, yet consistent and repeatable. The stimuli that can be created from these functions have a controllable amount of complexity and self-similarity properties that are further useful when generating naturalistic looking objects and surfaces for investigation. The paper addresses the creation and manipulation of stimuli with the use of noise, including an overview of this particular implementation. Stimuli derived from these procedures have been used successfully in several shape and surface perception experiments and are presented here for use by others and further discussion as to their utility.

## 1 Introduction

As vision research has evolved from 'brass-instrument' to 'silicon-instrument' science, computer-graphics techniques have made entirely new classes of experimentation possible. For example, in experiments on 3-D shape perception, depictions of arbitrary objects can be created 'on-the-fly' in real time for use in methods where direct measurement may provide the best possible insight into some perceptual process. Thus, the method of adjustment can be used directly on almost any 2-D or 3-D geometric object. Parameters controlling an object's shape can be manipulated directly rather than indirectly, allowing psychophysical exploration of a much wider variety of candidate stimuli and parameterizations than ever before.

Furthermore, as computational power has increased, so too has our ability to generate more geometrically 'rich' or complex depictions of objects. In many early experiments, computational speed constraints restricted these stimuli to simple, primitive objects like spheres, cylinders, planes, etc. These experiments were able to make significant contributions to our fundamental understanding of 3-D shape and object perception, but experiments that require more ecologically valid stimuli were sometimes left wanting for more geometrically complex stimuli.(1) This applies especially to experiments meant to explore the nature of geometric representation *itself*.

Interest in the perceptual representation of 3-D shape and form has led us through several experiments whose stimuli were significantly more complex geometrically than those we had previously used. In our earliest experiments our stimuli objects were defined by deforming primitive objects (eg spheres) with the use of a Fourier-like composition of sine waves (Norman et al 1995; Koenderink et al 1996). This resulted in well-defined, analytically tractable objects whose visual and geometric complexity was far greater than that of objects created by traditional geometric primitives.

(1) By 'geometric complexity' I mean, in an information-theoretic manner, the minimum number of primitive elements (eg vertexes and edges) needed to accurately depict a particular 3-D object. For example, the 3-D geometry of a billiard ball has a relatively simple description—a sphere $2\frac{1}{4}$ inches (5.7 cm) in diameter. On the other hand, the geometry of a crumpled chewing-gum foil requires significantly more information to recreate it exactly.

Several colleagues[(2)] have pointed out that these stimuli appeared, phenomenally at least, to be geometrically more 'organic' than traditional geometric-primitive-based stimuli. Following up on this line of thinking, we have actually used organic material *as* stimuli (Norman et al 2001).

There is certainly an appeal in treating objects as Fourier-composable (and thus decomposable) entities. However, we noticed that, at least for the somewhat simple method we were using, our stimuli possessed a rather homogeneous appearance. Furthermore, the final objects were selected in a somewhat subjective manner—after generation the objects were visually probed for certain relevant characteristics (number of ridge structures, areas of spatial and curvature extrema, etc). This sort of 'visual random search' method was acceptable since it was used to find stimuli that had natural appearing characteristics but were not necessarily representative of a natural object per se. In later experiments we have begun to investigate more direct methods for generating ecologically complex stimuli with a desirable set of characteristics.

Thus, in some of our more recent experiments (Phillips and Todd 1996; Phillips et al 1997, 2003; Phillips and Thompson 2000; Phillips and Voshell 2001) we have investigated a different class of analytically defined stimuli. These stimuli possess geometric qualities frequently found in natural objects, specifically levels of quasi-fractal self-similarity (Thompson 1942; Mandelbrot 1977) and organized randomness derived from simple origins, also referred to as *complexity* in various philosophical and mathematical circles. In this paper, I describe these stimuli, their ecological relevance, and, most importantly, their generation.

It should be emphasized that, in this article, I concentrate on computer-graphic *depictions* of objects. However, there is no reason that these techniques could not be used to create *actual* physical realizations of objects through stereo-lithography or some similar process. Visual and haptic studies with truly 3-D objects similar in spirit to the ones discussed here have been performed by Norman and others (Norman et al 2003) and the techniques outlined here could be applied to the creation of suitable objects for further parametric exploration of their findings. Still, the unfortunate realities of journal publication force me to remain constrained to flatland depictions in this article.

## 2 Noise

Traditional signal-detection experiments measure an observer's or system's sensitivity to a signal embedded into some non-signal or 'noise'. Different experimental methods and sensory modalities will often dictate the dimensionality of this signal and associated noise. For example, a simple auditory detection experiment might use a 1-D sine-wave signal plus 1-D noise. A reading experiment might embed a 2-D signal of text into a background of 2-D noise. In our previously mentioned experiments on 3-D shape perception, the stimuli require an extension of noise into an additional dimension. As a result, I decided to further generalize this noise to an arbitrary number of dimensions so that it might be applicable to a greater variety of stimuli, visual and otherwise.

On first blush it might seem like a simple combination of random-number generators would suit our purposes—one generator for each needed dimension/parameter. While this intuition is correct at the core, our noise should possess other properties that further constrain and shape the randomness in a variety of ways to make it most useful.

Our random function is, at its essence, a scale-invariant, multidimensional distribution of smoothly varying random numbers. Not coincidentally, the classic computer-graphics literature refers to this class of function as *noise* (Peachey 1985; Perlin 1985). In our

(2) As well as assorted cleaning personnel, office managers, and random unsolicited observers.

particular case I use low-pass filtered white noise (sometimes called pink noise) that possesses useful properties for stimulus generation, namely
(i) it is random, yet repeatable;
(ii) its range if $[-1, 1]$ on each dimension;
(iii) it is band-limited;
(iv) it is statistically stationary and isotropic;
(v) it is pseudo-periodic.
(Modified, after Ebert et al 1998.)

By our definition, I would like a function that is random (actually pseudo-random in this computationally limiting case), yet repeatable—ie the same output can be generated for the same input. In this spirit, it is somewhat instructive to consider our noise as *itself* a potential signal. For even more flexibility I would like our noise to have some characteristics similar to traditional wave functions. Items (ii)–(iv) in the above list are consistent with traditional trigonometric wave functions: a fixed range and band limit make Fourier composition, decomposition, and general analysis possible.(3) The underlying process is statistically stationary and isotropic, meaning that it is stable over time(4) and is the same regardless of direction. Finally, the process is pseudo-periodic, that is it possesses a fixed wavelength ($\lambda$) but the function's value varies from cycle to cycle. For example, for the trigonometric wave functions,

$$f(\theta) = f(\theta + \lambda) . \tag{1}$$

Our noise, on the other hand, does not obey this relationship strictly. Yet, on average, the noise function spends half of its time above zero and half of its time below zero, in a way similar to traditional wave functions. Therefore, since *noise* is essentially a wave function, in that it is pseudo-periodic, it can be more broadly defined as

$$f(\theta) = a \times noise(F\theta + \phi) , \tag{2}$$

where $a$ represents amplitude, $F$ frequency ($1/\lambda$), and $\phi$ phase. For subsequent references, this parameterization is implied when the form $noise(\theta)$ is used.

### 2.1 *Noise as smooth randomness*

We first need a raw source or basis from which to create our noise. Our source consists of the output of a uniformly distributed random-number generator on the range $[-1, 1]$ (ie white noise) whose values have been further constrained to alternate above and below 0. This noise is then box-filtered at an appropriate frequency to yield a smoothly differentiable signal that interpolates between deterministic, pseudo-random values. As an example, figure 1 shows the results of a uniformly random function, filtered at different frequencies resulting in various smooth random signals.

2.1.1 *Implementation details.* As stated above, the basis for our smooth noise function is the output from a uniform distribution of continuous values on the interval $[-1, 1]$. For computational efficiency, we pre-calculate a large number of random values which are then stored in two lookup tables, one for numbers on the interval $[-1, 0)$ and another from $(0, 1]$. To generate a particular random basis function a third table is consulted containing indexes into the random-number tables. This third table, sometimes called a *permutation* or *address hashing table*, is a shuffled list of locations in the other two tables. For example, if there are 256 random numbers in each of the positive/negative lists, the permutation table contains the numbers 1 through 256 in

(3) Previously, I mentioned that some of the stimuli used in our studies are pseudo-fractal. Of course, a truly fractal signal would not be band-limited. The basis function (ie our noise) used to recursively generate the fractal signal is band-limited.

(4) Actually, it is stable in any dimension(s) with respect to any other dimension(s).

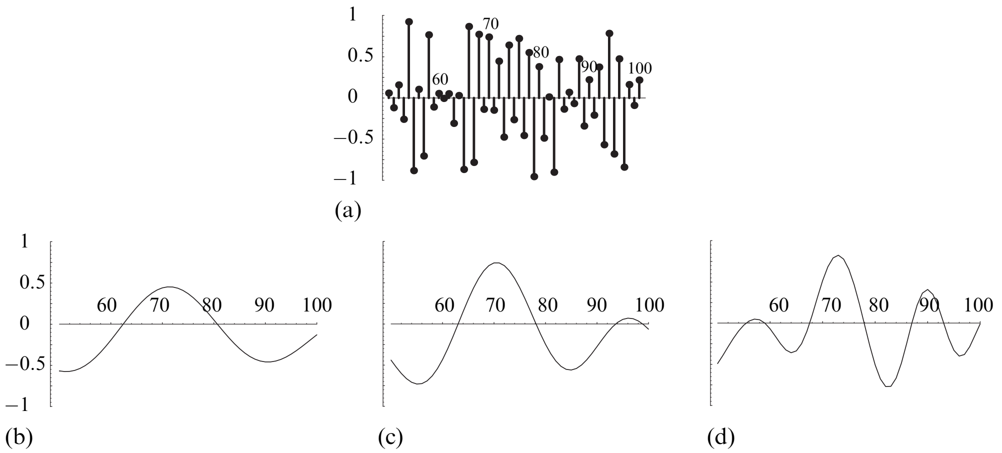
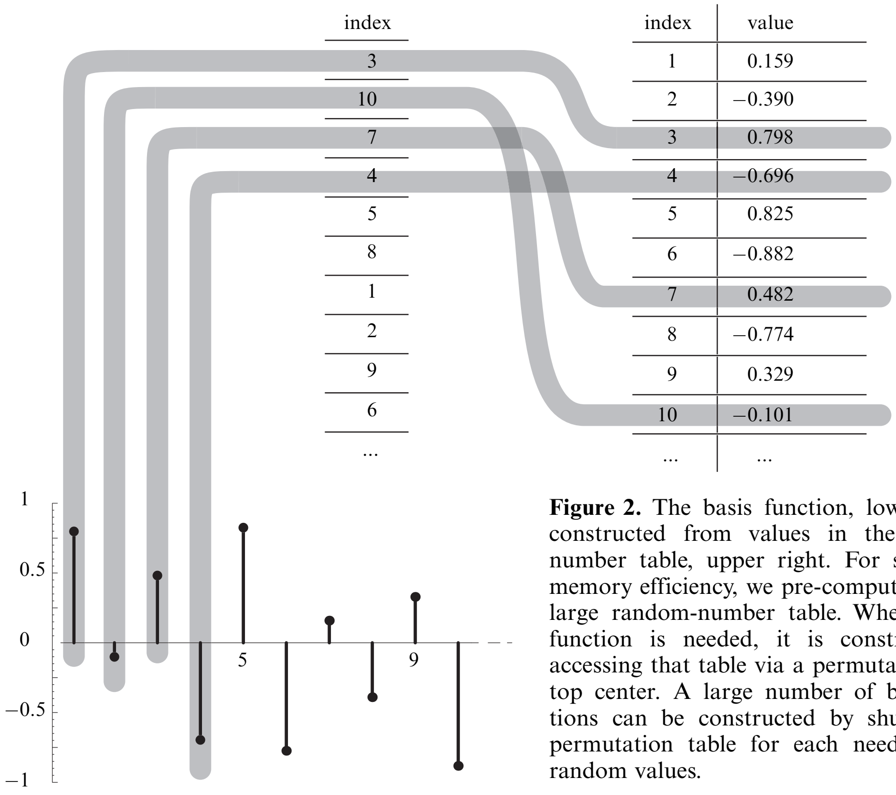


**Figure 1.** (a) A uniform, random function where each integer-valued location in $x$ has an associated random value that alternates above and below zero (all units are arbitrary). Below, a series of functions derived by box-filtering and renormalizing (a). (b) Shows (a) filtered at 1/20 cycle/unit, (c) shows (a) filtered at 1/10 cycle/unit, and (d) shows (a) filtered at 1/5 cycle/unit.

random order. Consulting these indexes in order, alternating between the positive and negative tables, yields the set of random numbers constituting our basis function. Figure 2 illustrates this process for one particular permutation.

Finally, the random basis function is filtered to yield the resulting smoothed signal. Generally, the filtering is achieved through simple discrete convolution. In cases where greater smoothness is required the signal can be 'supersampled' by interpolating additional discrete values between the basis function locations by using a cubic or other



| index |
|---|
| 3 |
| 10 |
| 7 |
| 4 |
| 5 |
| 8 |
| 1 |
| 2 |
| 9 |
| 6 |
| ... |

| index | value |
|---|---|
| 1 | 0.159 |
| 2 | −0.390 |
| 3 | 0.798 |
| 4 | −0.696 |
| 5 | 0.825 |
| 6 | −0.882 |
| 7 | 0.482 |
| 8 | −0.774 |
| 9 | 0.329 |
| 10 | −0.101 |
| ... | ... |

**Figure 2.** The basis function, lower left, is constructed from values in the random-number table, upper right. For speed and memory efficiency, we pre-compute a single, large random-number table. When a basis function is needed, it is constructed by accessing that table via a permutation table, top center. A large number of basis functions can be constructed by shuffling the permutation table for each needed set of random values.

interpolating spline (Bartels et al 1987). This resampled signal can then be filtered with a convolution kernel enlarged by a factor equal to the number of additional locations interpolated. For situations where speed is not essential (ie generation of stimuli not in real-time) I have used Mathematica (Wolfram 1999) to calculate more precise quasi-analytical filterings of the basis functions.

As an added feature, the statistical characteristics of noise can be broadly manipulated by utilizing different types of filtering (bandpass filters for example) and/or by using different distributions from which the basis function is created. While these variations are too numerous to cover here, Peachey (in Ebert et al 1998) describes several other methods for generating noise. These include using gradient calculations, non-regular basis functions, and various convolution techniques.

2.1.2 *Stimulus example.* Several investigators have sought to measure the perceptual thresholds of the curvedness of a line segment (Graham 1965; Tyler 1973; Watt and Andrews 1982). In most of these experiments there is a tacit assumption that the segment under consideration is smooth and sharply contrasted with its background. What if the extent of the line was not constrained in any way, that is, what if the edges could take on a 'fuzzy' or noisy character? One way to achieve this distortion is to simply distort the radius of curvature as a function of angle. The single-parameter equation

$$f(\theta) = noise(\theta) + r\{\sin(\theta),\ \cos(\theta)\} \tag{3}$$

shows the addition of noise to the arc of radius $r$, as a function of $\theta$. Figure 3 illustrates an example of this manipulation at various amplitudes.

From roughly an arm's-length viewing position, the top segments appear curved but as the amplitude of the noise is increased it is not as clear that there is a global curvature underlying the local deformations. Notice that, for each amplitude, the pattern of peaks and troughs remains the same, and that their relative height and depth changes from cycle to cycle. This clearly illustrates the characteristics and underlying structure of our 1-D noise.

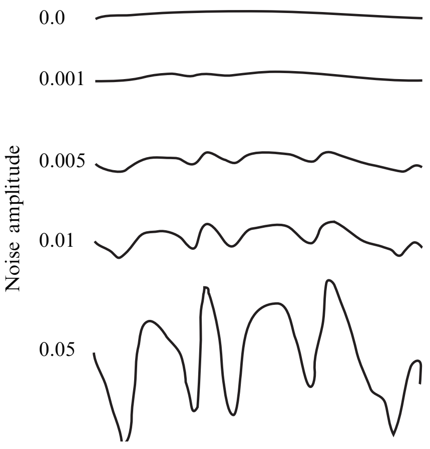


**Figure 3.** 1-D noise from our example experiment. A 5° arc segment is shown with varying amounts of added noise. In each case the amplitude of the noise added is a function of the angular position on the arc. The overall amplitude of the added noise increases in each subsequent figure, in a proportion indicated to its left. The frequency and phase offset remain constant at 2 cycles per degree and 0, respectively.

## 2.2 *Multidimensional noise*

Figure 1 above illustrates a simple scalar-input/scalar-output noise function, which is the foundation for all other variations used here. This traditional 1-D noise is useful for a large class of stimulus generation situations where a wave-like function is needed. But one important requirement of our noise function is that it be multidimensional in range *and* domain; that is, an arbitrary-sized noise output (scalar or vector) is obtained for any size input (scalar or vector).[5]

$$y = noise(x)\,, \tag{4}$$

$$y = noise(\boldsymbol{x})\,, \tag{5}$$

$$\boldsymbol{y} = noise(x)\,, \tag{6}$$

$$\boldsymbol{y} = noise(\boldsymbol{x})\,. \tag{7}$$

The multidimensionality of the *noise* function provides flexibility to manipulate any parameterized object characteristic regardless of its inherent dimensionality, for example if one would like to assign a particular luminance value (a scalar output) to a location on an object based on its 3-D position in space (a vector input), or if one would like to specify a color (a vector) based on depth from the observer (a scalar).

### 2.3 *Multidimensional output*

Creating a function with a multidimensional output is a somewhat straightforward extension of the 1-D case. Equations (6) and (7) illustrate this situation. Essentially, a different random basis function is constructed for each required output dimension. Each filtered smooth function is accessed at the same input parameter location, the results are collected into a vector of the requested dimensionality and returned.

2.3.1 *Implementation details.* The implementation here simply uses a different 1-D noise generating function for each dimension of the desired output. For computational efficiency, the permutation method shown in figure 2 is again used, only with a different permutation for each requested function. For example, for the first dimension we might access the permutation sequentially, for the second dimension we could skip every other value. More complex access methods involve permuting access into the permutation itself via a nested function. Thus, for the scalar-input – vector-output case [see equation (6)] any size output vector can be created from a single input value by indexing the respective functions for each requested dimension. As with the 1-D case, the resulting bases are filtered to yield smooth random values at a set of, possibly different, frequencies.

2.3.2 *Stimulus examples.* This class of noise is especially useful in temporal scenarios. As an example, consider the case of time-varying 2-D noise. We would like a 2-D output (a point's location for example) that changes as a function of a scalar input (time). Figure 4 shows an example of this: in this case, the $\{x, y\}$ location of a hypothetical point is varied as a function of single scalar parameter, time.

Figure 5 shows a series of parametrically defined circles, defined as in equation (3), but, instead of being distorted along the radius, each point is distorted in both $x$ and $y$ via 2-D output from noise. These stimuli and their generation are similar in spirit to those of Alter and Schwartz (1988) and Hess et al (1999). Our method provides an alternate parameterization of the deformation that might be useful in further exploration of their findings.

The output of noise is not limited to only two dimensions. For example, the noise function can be used to distort surface locations on objects over time, creating anything from blurry 'soft' surface/air interfaces to undulating, boiling objects by appropriately adjusting the frequency and amplitude of the noise.

Last, we are not limited to distortions in position. A 7-D output might be used to change position, transparency, and RGB color of a location on a 3-D surface, all of which would then vary smoothly over the function's parameter, time as an example.

[5] I use the traditional notation $\boldsymbol{x}$ to denote vectors and $x$ to denote scalars.

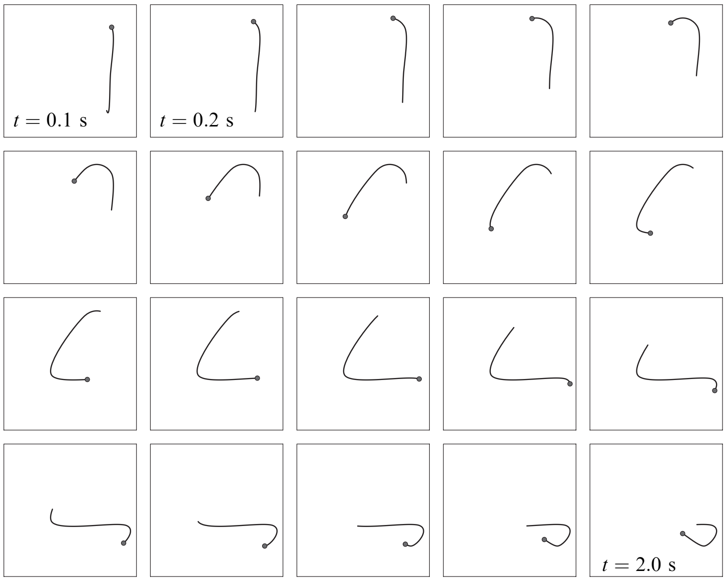


**Figure 4.** An example of multidimensional noise output from a scalar input. Here, a sequence of frames is shown, illustrating a randomly generated trajectory over 2 s. The $\{x, y\}$ location of the point is determined as a function of a single parameter, $t$ (time). To clearly illustrate the point's trajectory a 1 s 'tail' is added.

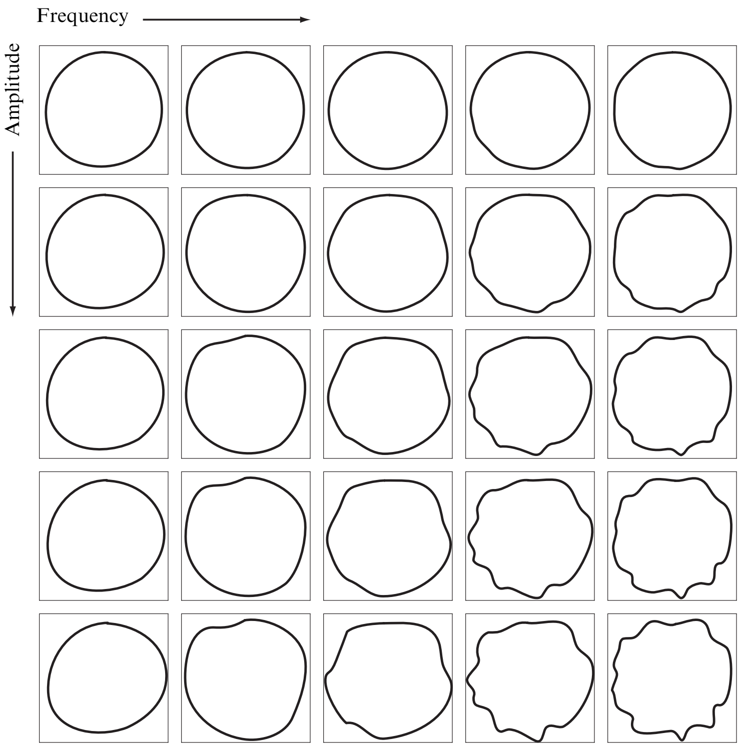


**Figure 5.** An example of multidimensional noise output from a scalar input. Here, a series of parametrically defined circles, with $\{x, y\}$ location defined as a function of $\theta$ as in equation (3), but distorted in rectangular space instead of polar space. Amplitude of noise increases from top to bottom, frequency from left to right.

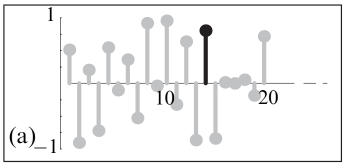
## 2.4 *Multidimensional input*

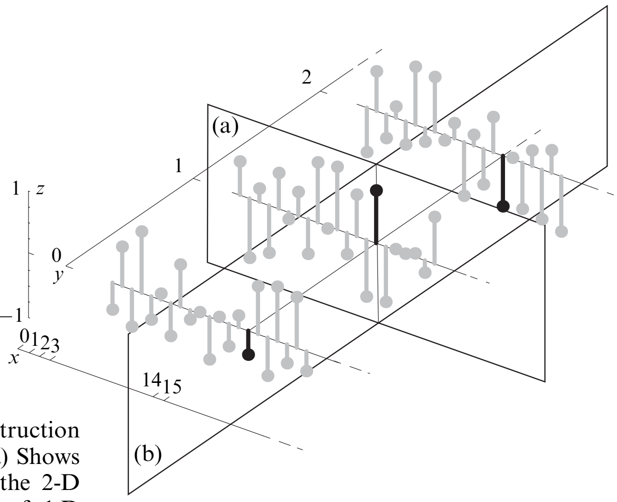
Some scenarios demand a unique output for an arbitrary-dimensioned input. For example, if we would like to jitter the intensity of an RGB-specified color as a function of all three of its components we might simply add the $\{r, g, b\}$ elements to create a scalar input for noise but there would be significant overlap when the elements of the color are the same only shuffled (cf the individual components of full on red, green, and blue—$\{1, 0, 0\}$, $\{0, 1, 0\}$, $\{0, 0, 1\}$—will all sum to 1).

Thus we need a way to map the multiple components of the input vector into a predictable and consistent location in the noise function. Since the goal of our noise is smoothly varying, systematic randomness, we would like a different version of the noise function for any given dimension of input vector, yet we would like a smooth, continuous output from the function for sufficiently close values of the input vector. Our solution is similar to that for multidimensional output—additional random functions are created by the permutation method above, one function for each integer-valued location on each dimension.

2.4.1 *Implementation details.* Figure 6 illustrates the process used for multidimensional input. In this 2-D example, a plurality of 1-D noise basis functions are generated by the permutation method shown in figure 2, one function for each integer-valued location in the second dimension. Figure 6a shows one such function at $y = 1$.[(6)] For a particular $\{x, y\}$ location the appropriate set of values are chosen from each noise function for the $x$-location (figure 6b). This signal is appropriately filtered and sampled at the appropriate $y$-location, resulting in the scalar value returned by the function.

This process can be repeated over any number of dimensions to yield the desired single *scalar* value for an arbitrary-length vector input. For example, a 3-D input would collect the various (b) planes for the first two dimensions, as generated by the method





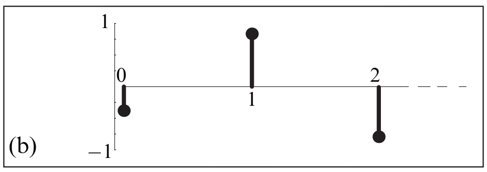


**Figure 6.** An example of the construction of scalar noise for a 2-D input. (a) Shows a given basis function in 1-D. For the 2-D case shown here, multiple planes of 1-D noise are extended into another dimension. (b) Shows a function that derives its values across the multiple instances of 1-D noise. Thus, for a given 2-D input (here, $\{14, 1\}$), the appropriate noise basis function is derived from the values at the $x$-location in each $y$ function.

(6) In the traditional computer-graphics literature this set of functions is referred to as the 'integer lattice'.

shown in figure 6 and use them as (a) planes for the final interpolation. As with the multidimensional output case, each dimension can be filtered separately, resulting in anisotropic or isotropic values if necessary.

2.4.2 *Stimulus examples.* Figure 7 shows an example of noise generated as a function of $\{x, y\}$ location and time. Each image shows a 2-D smoothly varying spatiotemporally random signal. The output of the noise function is a scalar representing the gray-level of each pixel. Note that the time dimension could have just as easily been another spatial dimension, creating a solid block of noisy values. This type of noise (so-called solid noise) is frequently used in computer-graphics simulations of materials that possess smoothly variable but solid structure in 3-D, such as marble, granite, wood, clouds, etc. In the case of time-varying 3-D objects, such as clouds, the addition of another dimension to the input vector results in temporally evolving 3-D structure, similar to the evolving structures shown in figure 7.

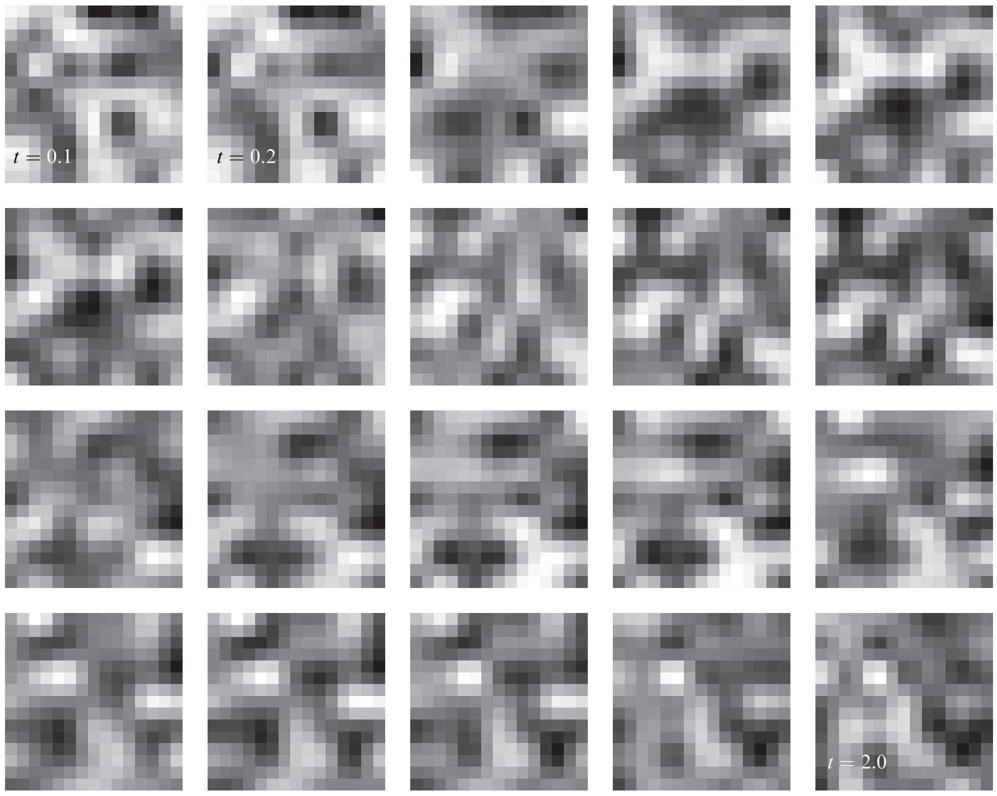


**Figure 7.** An illustration of multidimensional-input/scalar-output noise. In this figure a series of 2-D plots of noise are shown further varying as a function of time. The intensity at a given $\{x, y\}$ location is determined as a function of its location and $t$ (time). Note that this third parameter could just as easily have been another geometric parameter, creating a block of so-called 'solid noise'.

Of course, we are not limited to planar stimuli and it is straightforward to create noisy 3-D surfaces as well. One simple surface description is the Monge form (Porteous 1994), specified in 3-D by

$$z = f(x, y) \,. \tag{8}$$

Figure 8 shows a Monge surface with noise as a function of $\{x, y\}$ position to determine the height (a scalar) of the surface at that location.

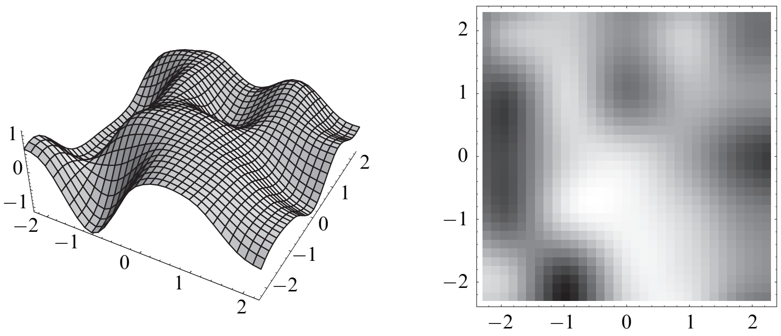


**Figure 8.** A simple Monge-form surface with noise used to generate the height at a given $x, y$ surface location. On the left, a perspective view of the surface and on the right a depth map with lighter shading indicating larger values for the function.

Stimuli of the class shown in figures 7 and 8 have been used in several of our experiments, sometimes used as 'height fields' (see next section) for surface perception or used in their 2-D form as textures or as background noise for other stimuli in signal-detection experiments (Phillips and Todd 1996; Phillips et al 1997, 2003; Phillips and Thompson 2000; Phillips and Voshell 2001).

## 2.5 *Multidimensional input and output*

Last, the case of multidimensional input and output, as shown in equation (7) can easily be achieved via multiple instantiations of multidimensional input noise described in the previous section. This class of noise rounds out the taxonomy of input/output scenarios that one might encounter when using this function.

2.5.1 *Implementation details.* The implementation of multidimensional input/output noise is simply a direct combination of the multidimensional input and multidimensional output methods described above. As with these methods, a permutation system is used to create the various basis functions from a random-number table in order to facilitate computational efficiency. Similarly, filtering is done via convolution or resampling then convolution, depending on smoothness and speed constraints.

2.5.2 *Stimulus examples.* Figures 9 and 10 show 2-D and 3-D examples of multi-dimensional-input/output noise, respectively. Both examples illustrate smoothly spatially varying vector fields, fields that can be used to create optic flow or other stimuli locally varying in time and/or space. Furthermore, as with other varieties of our noise, it can be scaled and added to an existing signal for signal-detection experiments.

As mentioned previously, the multidimensional capability of the function can be used in a variety of ways other than distorting geometry. For example, figure 11 illustrates the use of a vector output to map both height and color to the surface. In one case, a hue is selected by using a scalar value and in the other an RGB value is selected by using a $\{r, g, b\}$ triplet vector, requiring a 2-D or 4-D output vector, respectively.

As a whole, the dimensional flexibility of our noise makes this function ideal for the creation of signals or objects whose description is best done with a vector whose size is greater or smaller than the input vector. For example, if, for a given experimental manipulation, two different signals need to be created for a single independent variable, it can easily be done by requesting a 2-D output for the single variable. Similarly, if only one value is needed for an experiment with multiple independent variables, a multidimensional input vector can be input and only one value requested.

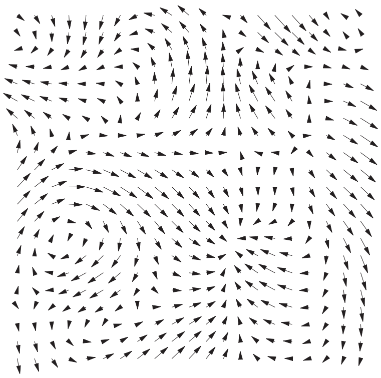


**Figure 9.** A 2-D flow field illustrating multidimensional input and output noise. Here, at each $\{x, y\}$ location a 2-D vector is computed by using noise. As an example, the resulting vector field can be used to generate an optic flow field as either stimuli or as noise, scaled and added to an existing signal.

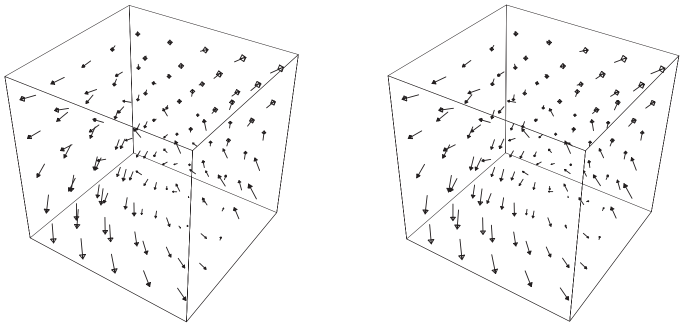


**Figure 10.** A 3-D vector field generated in the same manner as the 2-D version above. The stereo pair is cross-fusible. As with the 2-D case it can be easily seen that the vectors change direction in a globally random, but locally smooth, manner.

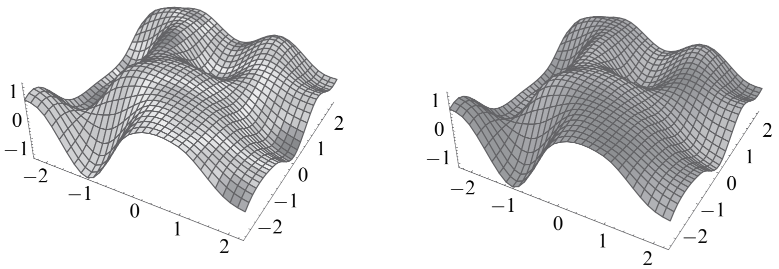


**Figure 11.** Surfaces where both the geometry and color are determined by operating on a multi-dimensional output of noise. On the left a 2-D vector is returned with one component used to determine the height and the other used to select a hue. On the right, a 4-D vector is requested, the first element used as a height and the remaining three used as a vector to determine an RGB color value.

## 3 Fourier synthesis

While a single noise signal can be used to create nicely varying smooth surfaces, a more interesting, visually complex, and potentially ecologically relevant class of surfaces can be created by a summation of multiple signals.

In the computer-graphics literature (Peachey 1985; Perlin 1985; Upstill 1989; Ebert et al 1998; Apodaca and Gritz 1999), *turbulence* has been used to describe a class of self-similar (or *fractal*) (Mandelbrot 1977) functions created by summing multiple octaves of the noise function, scaling by a $1/F$ factor at each octave. $n$ octaves of noise at location $\boldsymbol{x}$ can be computed as

$$fracsum(\boldsymbol{x}, n) = \sum_{F=0}^{n-1} \frac{noise(2^F \boldsymbol{x})}{2^F} . \tag{9}$$

Figure 12 shows a set of plots illustrating the summation of an incrementally larger number of ($n$ from 1 to 5) octaves of our noise function.

Notice that the overall, low-frequency shape of the function maintains its presence throughout the addition of more octaves. The $1/F$ scaling takes care of placing the appropriate de-emphasis on each additional octave.

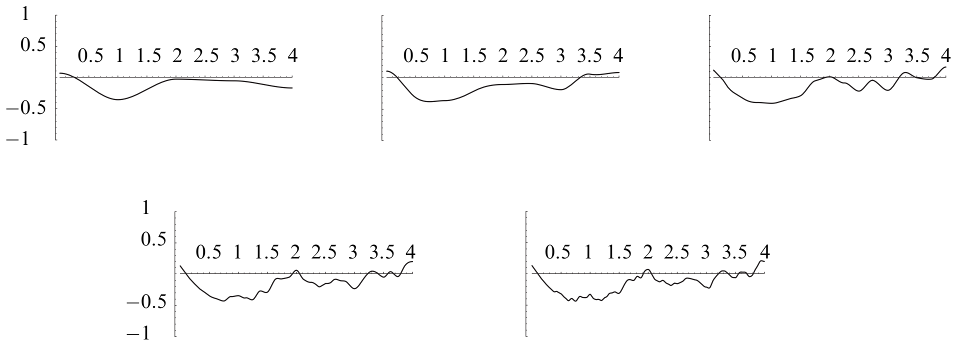


**Figure 12.** A collection of plots showing a range of $n = 1, ..., 5$ octaves of noise, summated. Top left shows the low-frequency carrier function and each subsequent graph includes an additional octave superimposed.

## 3.1 *Fractal versus turbulent summation*

Perlin's implementation of noise (Perlin 1985) makes a slight differentiation between fractal and turbulent summation. In turbulent summation the *magnitude* of the noise function is used to introduce discontinuities in the differential structure as shown in equation (10).

$$turb(\boldsymbol{x}, n) = \sum_{F=0}^{n-1} \frac{|noise(2^F \boldsymbol{x})|}{2^F} . \tag{10}$$

This rectified function creates distinct structures defined by the discontinuities, each at different scales. Objects that require discontinuities, such as landscapes, benefit from this class of function. Figure 13 shows the results of summation of 5 octaves of the magnitude of the noise function. Compare this with the lower right of figure 12 which contains the exact same set of wave functions, frequencies, and phase offsets only with their magnitude preserved during summation.

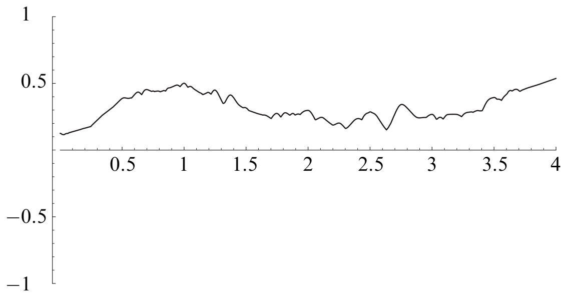


**Figure 13.** Result of summing 5 octaves of the magnitude of the noise function.

## 3.2 *Surfaces*

There is a significant amount of evidence pointing to the fractal self-similarity of objects in nature (Mandelbrot 1977), and a vast amount of physical research suggesting that the formation of surfaces at various scales can be modeled by fractals and other self-similar phenomena (Barabási and Stanley 1995). The results of Knill, Gilden, Cutting, Pentland, and others (Pentland 1983, 1986; Cutting and Garvin 1987; Knill et al 1990; Gilden et al 1993) suggest that, at least for 1-D and 2-D versions of this class of function, human observers can perform discrimination of various sorts on stimuli generated by using fractal Brownian motion (similar to that shown in figure 12). More importantly, Pentland (1983) observed that there are information-processing and linguistic conveniences to describing objects in terms of their self-similarity. Self-similar fractal functions have been used for years in computer graphics to model naturalistic phenomena such as mountains, clouds, fire, foliage, and others. As an example, the surfaces in figure 14 show the effect of creating 3-D surfaces by superimposing multiple octaves of the noise function according to equation (9).

We can do the same with the turbulence-style magnitude summation as well, as seen in figure 15. These surfaces have a phenomenally different shape than the smooth surfaces in figure 14. They are somewhat reminiscent of mountainscapes or other terrestrial structures.

Interestingly, Cutting and Garvin (1987) also make the point that objects in nature tend to have only three to four levels of self-similarity. This nicely suggests that the computational complexity of these functions may be limited in the same way. Indeed, in cases where true octaves (ie frequency in powers of 2) of noise are used, we can

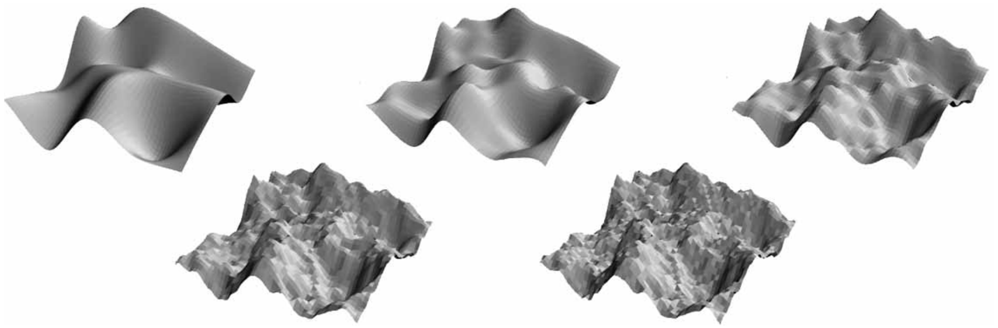


**Figure 14.** A collection of surfaces showing a range of $n = 1, ..., 5$ octaves of noise, summated. Top left shows the low-frequency carrier function and each subsequent graph includes an additional octave superimposed.

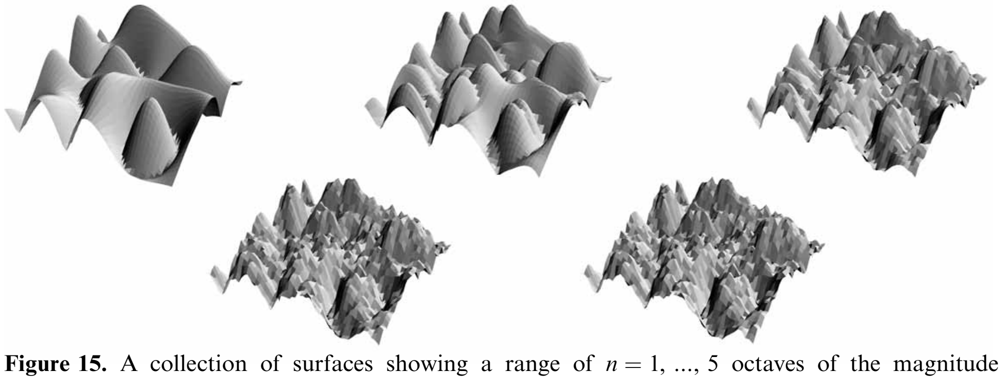


**Figure 15.** A collection of surfaces showing a range of $n = 1, ..., 5$ octaves of the magnitude of the noise function, summated. Top left shows the low-frequency carrier function and each subsequent graph includes an additional octave superimposed.

cover the scale from microscopic to the galactic with a rather humble handful of iterations—a fractal surface with 64 iterations of noise would cover a range of scales out to about $1 : 10^{18}$.

## 4 Objects

In addition to surface displacement we can also use the same functions to displace solid objects. This displacement typically (but not necessarily) moves points on the surface along the surface normal vector $\boldsymbol{n}_p$, a component of the first-order differential structure perpendicular to a theoretical local plane at a given point $p$. So, for a given location $p$ on a surface,

$$p' = p + s\boldsymbol{n}_p \tag{11}$$

displaces it by a scale $s$ along $\boldsymbol{n}_p$. In our stimuli we use the *noise*, *fractalsum*, and *turbulence* functions to scale this displacement. The input to each function is the 3-D location of the point $p$, and the output is a scalar specifying the displacement $s$ along the normal at $p$. In the computer-graphics literature this is sometimes known as solid texture since, unlike the traditional case of flat textures being 'mapped' onto surfaces, these functions are 3-D volumes and the surface is determined from iso-valued locations in that volume. Recall that, since the output of these functions can be multidimensional, these solid textures can be used for more than geometric displacement. Similarly, since the input can be of higher dimensionality as well, the textures can exist in a hyperspace. 3-D objects can be displaced not only by their location in space but also in time or over some other parameterization.

Figure 16 shows a series of unit spheres whose radii are distorted by adding a scalar output from the basic noise function. Both frequency and amplitude are shown as varying.

Similarly, we can distort the spheres using the fractal and turbulence equations above. Figure 17 illustrates the effects of superimposition of multiple octaves of the noise and noise-magnitude functions.

Of course, we are not limited to mere spheres—any geometric object can be distorted with this technique as is illustrated in figure 18. This technique might be useful in object categorization experiments, whereas the desired noise masking would take place at the geometry level rather than at the image level (figure 19).

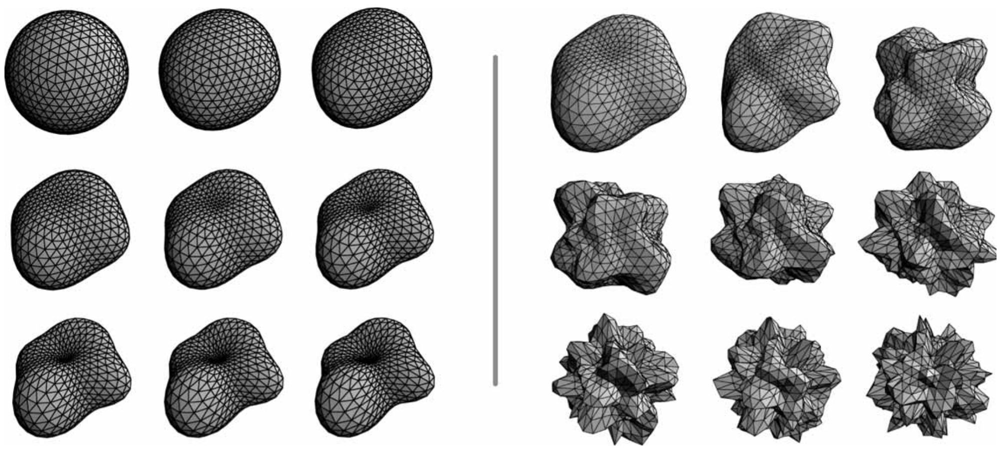


**Figure 16.** A collection of unit spheres displaced according to equation (11) by using noise. On the left, frequency is held constant and the amplitude of the added noise increases left to right, top to bottom. On the right, amplitude is held constant while frequency is increased.

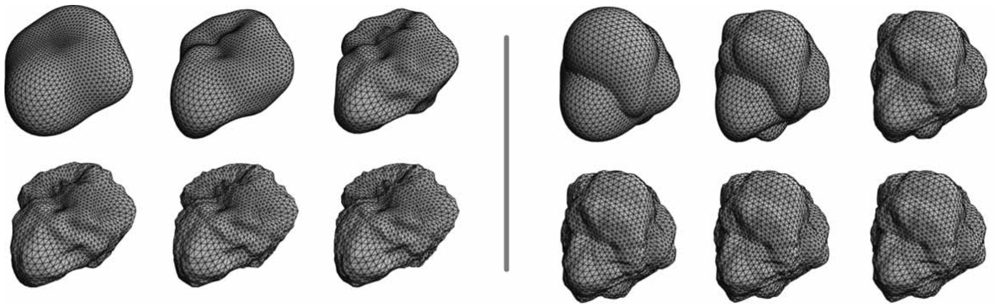


**Figure 17.** A set of unit spheres distorted with summated octaves of a 3-D noise function in a fractal fashion (left) and a turbulent fashion (right). The number of octaves ranges from $n = 1$ to $n = 6$ starting at the top left.

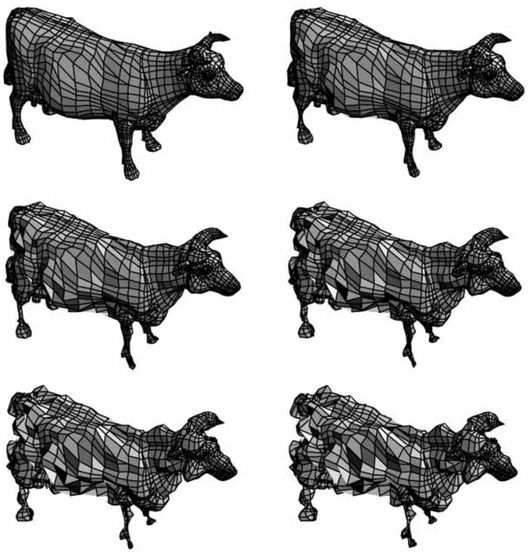


**Figure 18.** An unsuspecting yet apparently content bovine, subjected to noise ranging from amplitude $s = 0$ to $s = 0.5$ in steps of 0.1.

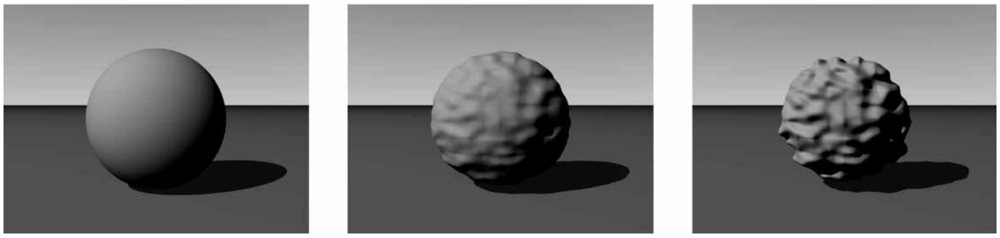


**Figure 19.** Stimuli for a hypothetical 3-D version of the line curvature experiments shown above. Each of the three spheres has been 'squashed' by 5% vertically and the amplitude of the added noise is increased from 0 in the left sphere to 10% of the radius of curvature, on average, in the right sphere.

Finally, in several of our experiments we have chosen to calculate and display the objects using a more sophisticated rendering system (Pixar's RenderMan or the POV-Ray Ray Tracer) which allows us to distort the object while maintaining the true nature of the geometry at the sub-pixel level. In the previous examples, the objects are depicted by means of a wire-frame polygonal mesh, and the vertices of these polygons are manipulated with the various noise functions. In RenderMan and its related rendering environments the base sphere object is represented by a parametric description rather than by a polygonal approximation. Thus, when we distort locations on this parametric sphere, we are performing the distortions on an analytical object rather than an approximation. This allows us to manipulate the features at infinitely many scales. Figure 20 shows some of the stimuli used in our recent experiments that were generated in this fashion.

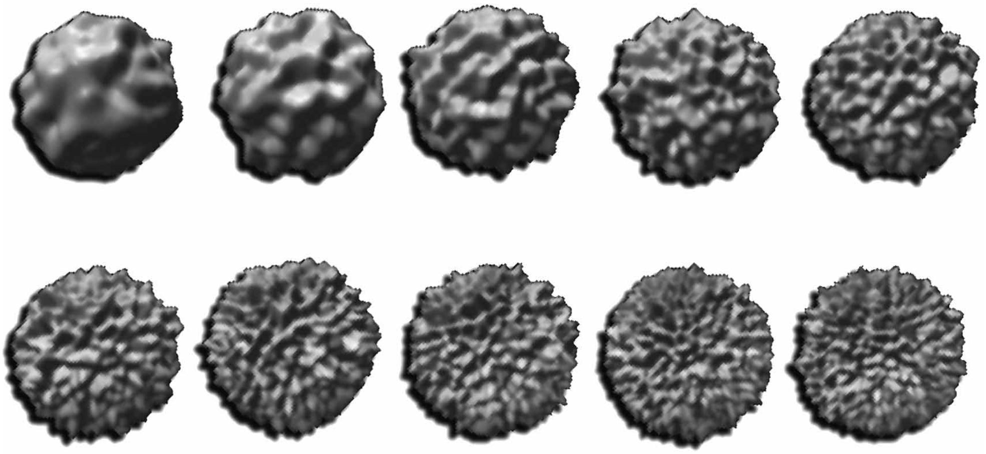


**Figure 20.** A series of stimuli generated using these techniques in RenderMan. Amplitude remains constant and frequency increases from left to right, top to bottom.

## 5 Discussion

As we attempt to investigate the nature of the perceptual representation of objects and surfaces in the world we are naturally required to develop theories which are consistent with the objects and surfaces found there. Initially it suffices to investigate what we might theorize to be fundamental properties, such as the global slant and tilt of a planar object or the localization of points in space, but eventually, if our theories are to stand up to the closest scrutiny, we will have to show that the findings of our artificial, contrived stimuli generalize to the ecological space we inhabit. Indeed, it is this criterion that has led many to investigate objects in situ, largely forgoing the laboratory. Still others choose to strike a balance between the laboratory and the environment on the basis that we may never be able to create artificial stimuli rich enough to mimic some phenomenal aspects of our visual world.

I have described a framework for calculating stimuli that reflect *some* characteristics of objects in the natural world—characteristics that seem to be perceptually relevant and possibly fundamentally involved in the way that we see. Pentland (1983) suggests certain composite, self-similar functions not only ease in the information theoretic descriptions of natural scenes, but also suggests that there are important linguistic parallels to this information. For example, many of the surfaces shown in the previous examples would be well classified as 'mountainous' or 'bumpy'. Similarly, computer-graphics techniques using the same basic self-similarity procedures are employed to create marble, wood, even flames. This is visually compelling testimony to the natural relevance of this class of stimuli. Knill et al (1990), Cutting and Garvin (1987), and Gilden et al (1993) have empirically investigated the perceptual relevance of self-similarity. Also, Barabási and Stanley (1995) have advanced compelling physical evidence that the surface structures of objects in the real world are governed, at least in part, by self-similar mechanisms. The implications for texture and object perception seem rather clear from these ideas—the 'roughness' of a surface at a variety of scales from the microscopic to the mountainous might well be a function of the number of octaves of noise-like randomness that they contain.

Stimuli we have created in this manner have been used primarily to investigate the perceptual representation of 3-D shape, but obviously this does not preclude these techniques from other applications. For example, as pointed out above, computer-graphics artists have used these textures and object successfully for creating phenomenal clouds, marble, wood, flame, and other naturally occurring phenomena. Investigators wishing to study the perception of natural phenomena in a controlled setting could easily use these procedures. Similarly, recognition and detection in the context of noise is of interest to those both at the psychophysical level and in various higher-level cognitive domains [see Gilden et al (1993), and Pelli and Farrell (1999) for a discussion of noise in perception of both images and objects].

By distorting an object with noise instead of its image we may well be able to discover essential 3-D geometric characteristics necessary for successful object recognition. Of course, in this paper we are limited to 2-D depictions of 3-D objects, but this does not preclude the generation of actual 3-D representations of these stimuli. Similarly, the turbulence functions and their kin are almost certainly useful for investigating the scale at which certain geometric or texture phenomena might take place. As mentioned above, texture and surface characteristics could also be easily manipulated with noise as well. More generally, this type of function can be used as noise in any signal-detection situation. The flexibility inherent in its multidimensional range and domain along with its easily controllable statistical properties make it a potentially useful choice for a wide variety of stimulus modalities and experimental paradigms.

**Acknowledgment.** Flip Phillips was supported by a grant from the Keck Foundation.

**References**

Alter I, Schwartz E L, 1988 "Psychophysical studies of shape with Fourier descriptor stimuli" *Perception* **17** 191 – 202

Apodaca A A, Gritz L, 1999 *Advanced RenderMan: Creating CGI for Motion Pictures* (New York: Morgan Kaufmann)

Barabási A-L, Stanley H E, 1995 *Fractal Concepts in Surface Growth* (Cambridge, UK: Cambridge University Press)

Bartels R, Beatty J, Barsky B, 1987 *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling* (Los Altos, CA: Morgan Kaufmann)

Cutting J E, Garvin J J, 1987 "Fractal curves and complexity" *Perception & Psychophysics* **47** 365 – 370

Ebert D S, Musgrave F K, Peachey D, Perlin K, Worley S, 1998 *Texturing and Modeling: A Procedural Approach* 2nd edition (New York: Academic Press)

Gilden D L, Schmuckler M A, Clayton K, 1993 "The perception of natural contour" *Psychological Review* **100** 460 – 478

Graham C H, 1965 "Visual form perception", in *Vision and Visual Perception* Eds C H Graham, N R Bartlett, J L Brown, Y Hisa, C G Mueller, L A Riggs (New York: John Wiley) pp 548 – 574

Hess R F, Wang Y Z, Dakin S C, 1999 "Are judgements of circularity local or global?" *Vision Research* **39** 4354 – 4360

Knill D C, Field D, Kersten D, 1990 "Human discrimination of fractal images" *Journal of the Optical Society of America A* **7** 1113 – 1123

Koenderink J J, Kappers A M L, Todd J T, Norman J F, Phillips F, 1996 "Surface range and attitude probing in stereoscopically presented dynamic scenes" *Journal of Experimental Psychology: Human Perception and Performance* **22** 869 – 878

Mandelbrot B B, 1977 *Fractals: Form, Chance, and Dimension* (San Francisco, CA: W H Freeman)

Norman H F, Norman J F, Clayton A M, Lianekhammy J, Zielke G, 2003 "The visual and haptic perception of natural object shape" *Journal of Vision* **3** 778 (abstract)

Norman J F, Phillips F, Ross H E, 2001 "Information concentration along the boundary contours of naturally shaped solid objects" *Perception* **30** 1285 – 1294

Norman J F, Todd J T, Phillips F, 1995 "The perception of surface orientation from multiple sources of optical information" *Perception & Psychophysics* **57** 629 – 636

Peachey D R, 1985 "Solid texturing of complex surfaces" *Computer Graphics* **19** 279 – 286

Pelli D G, Farrell B, 1999 "Why use noise?" *Journal of the Optical Society of America A* **16** 647 – 653

Pentland A, 1983 "Fractal-based description of natural scenes", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (New York: IEEE) pp 201 – 209

Pentland A, 1986 "Perceptual organization and the representation of natural form" *Artificial Intelligence* **28** 293 – 331

Perlin K, 1985 "An image synthesizer" *Computer Graphics* **19** 287 – 296

Phillips F, Thompson C H, 2000 "Implications of two and three dimensional information in the perception of objects" *Investigative Ophthalmology & Visual Science* **41**(4) 5723, abstract #3849

Phillips F, Todd J T, 1996 "Perception of local three-dimensional shape" *Journal of Experimental Psychology: Human Perception and Performance* **22** 930 – 944

Phillips F, Todd J T, Koenderink J J, Kappers A M L, 1997 "Perceptual localization of surface position" *Journal of Experimental Psychology: Human Perception and Performance* **23** 1481 – 1492

Phillips F, Todd J T, Koenderink J J, Kappers A M L, 2003 "What defines features on smoothly curved surfaces?" *Psychological Review* **65** 747 – 762

Phillips F, Voshell M G, 2001 "Contributions of geometric and image information in the perception of solid objects" *Journal of Vision* **4** 300 (abstract)

Porteous I R, 1994 *Geometric Differentiation: For the Intelligence of Curves and Surfaces* (Cambridge, UK: Cambridge University Press)

Thompson D W, 1942 *On Growth and Form: A New Edition* (Cambridge, UK: Cambridge University Press)

Tyler C W, 1973 "Periodic vernier acuity" *Journal of Physiology* **228** 637 – 647

Upstill S, 1989 *The RenderMan Companion: A Programmer's Guide to Realistic Computer Graphics* (New York: Addison-Wesley)

Watt R J, Andrews D P, 1982 "Contour curvature analysis: hyperacuities in the discrimination of detailed shape" *Vision Research* **22** 449 – 460

Wolfram S, 1999 *The Mathematica Book* 4th edition (Champaign, IL: Wolfram Media; Cambridge, UK: Cambridge University Press)

ISSN 0301-0066 (print) ISSN 1468-4233 (electronic)

PERCEPTION

VOLUME 33 2004

www.perceptionweb.com