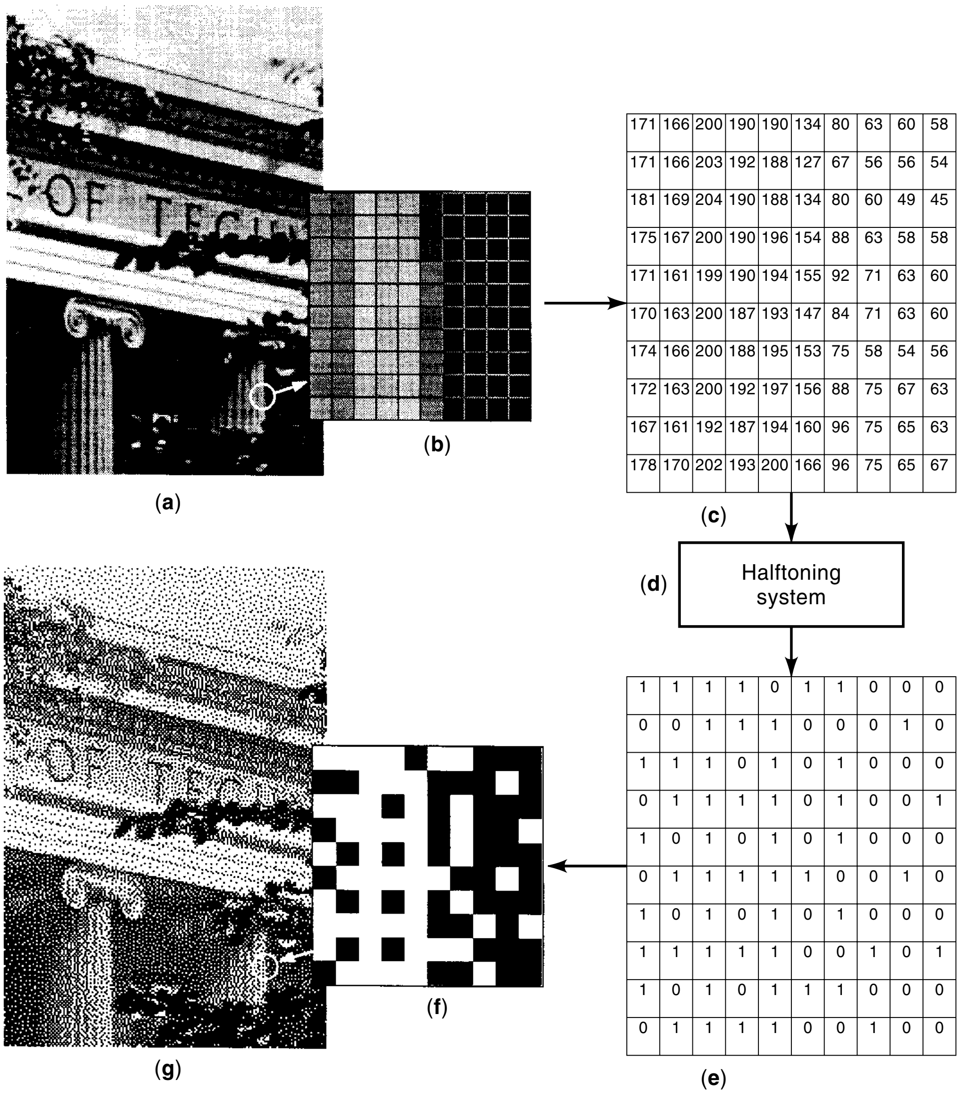
HALFTONING

Halftoning is the process of transforming an image with greater amplitude resolution to one with lesser amplitude resolution. This has been practiced for over a hundred years in the printing industry: the solution for displaying continuous-tone images with only black or white dots. In digital printing and display systems today the problem is essentially the same: how to produce the illusion of the original tonal quality of a image by judicious placement of dots. Digital implementations of halftone processes are also called "dithering," and the terms are often used interchangeably.

Figure 1 illustrates the role of a halftoning system. A continuous-tone image (a) is digitally represented by discrete sample (b) called picture elements or "pixels." The amplitude of each pixel is represented by an integer (c); in this example the amplitude values are represented with 8 bits, ranging from 0 (black) to 255 (white). These integer amplitude values are the input to a halftoning system (d) that performs some processing to produce output pixels (f) with amplitudes values

J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*. Copyright © 1999 John Wiley & Sons, Inc.



171	166	200	190	190	134	80	63	60	58
171	166	203	192	188	127	67	56	56	54
181	169	204	190	188	134	80	60	49	45
175	167	200	190	196	154	88	63	58	58
171	161	199	190	194	155	92	71	63	60
170	163	200	187	193	147	84	71	63	60
174	166	200	188	195	153	75	58	54	56
172	163	200	192	197	156	88	75	67	63
167	161	192	187	194	160	96	75	65	63
178	170	202	193	200	166	96	75	65	67

1	1	1	1	0	1	1	0	0	0
0	0	1	1	1	0	0	0	1	0
1	1	1	0	1	0	1	0	0	0
0	1	1	1	1	0	1	0	0	1
1	0	1	0	1	0	1	0	0	0
0	1	1	1	1	1	0	0	1	0
1	0	1	0	1	0	1	0	0	0
1	1	1	1	1	0	0	1	0	1
1	0	1	0	1	1	1	0	0	0
0	1	1	1	1	0	0	1	0	0

Figure 1. The halftone process: (a) Continuous-tone input image; (b) input pixels; (c) input amplitude values; (d) halftoning system; (e) output amplitude values; (f) output pixels; (g) output halftone image.

(e) of, in this case, 1 bit per pixel. The goal of the halftoning system is to generate an image (g) that is perceptually similar to the original input image (a).

The "perceptually similar" criterion is a very important one, and is linked to the characteristics of human vision. If we take a more traditional engineering criterion of minimizing mean-square error, for example, our halftoning system would generate the output shown in Fig. 2. Since human vision is the mechanism for measuring halftone output performance, other approaches must be taken. We exploit the fact that our high spatial frequency response drops quickly above 6 or so cycles per degree. So high-frequency patterns are perceived as their macroscopic average.

Dithering systems can be used to process color images on display devices with the capability to handle more than two levels of amplitude, as will be address at the end of this article. It is easier, however, to demonstrate and conceptualize dithering methods when used for generating bitonal output, and so examples will be shown for this case. We will explore various key classes of techniques for achieving the halftoning "illusion" and their trade-offs.

Figure 2. Minimum MSE output: Result of a fixed threshold.

Figure 3. Result of dithering with a white-noise threshold.

WHITE NOISE

Perhaps the first solution that comes to mind when considering the problem of how to distribute pixels in order to form macroscopic averages is to threshold with white noise. Historically this was the first approach taken (1) for electronic displays with independently addressable dots. An example of this is shown in Fig. 3. While the technique will break up the spurious contours seen in Fig. 2, it suffers from apparent graininess. This is due to the presence of long wavelengths in the dither patterns.

An image digitally dithered with white noise is often called a mezzotint after a printmaking technique invented in the seventeenth century. The practice involved roughening or grinding the dark regions on a copper plate by a skilled craftsman in a somewhat random fashion by hand. The resulting scratches acted as tiny wells which held ink. A photographic enlargement detailing an actual seventeenth-century mezzotint is shown in Fig. 4. The patterns do not suffer the very long wavelengths seen in white noise dither, but also are not as structured as those due to periodic screens. The ancient mezzotint engravers would probably be outraged at the association. A true mezzotint beautifully renders delicate shades of gray without the graininess seen in white noise. However, the term is used in modern software applications to describe white-noise dithering.

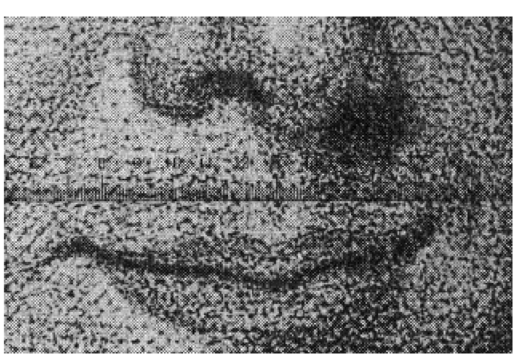


Figure 4. Detail of a seventeenth century mezzotint. Numbered reticle marks are in millimeters.

Frequency-Domain Metric

Representing signals in the frequency domain can often simplify complexity seen in the spatial domain. This is indeed the case with dither patterns. It allows us a means to examine the distribution of energy and its consequences on the quality of the patterns. As it is the flat regions of an image where the nature of dither is most important, the focus will be on the power spectrum of patterns that result from the dithering of a single fixed gray level.

Two-dimensional spectral plots can also afford to be made more succinct. Most well-formed patterns share the characteristic of being isotropic. This leads to a metric that summarizes the spectrum radially. Figure 5 shows the segmenting of a spectral period into concentric annuli. Averaging the power spectrum in each annulus results in a "radially averaged power spectrum" where these averages are plotted against the "radial frequency," the radial distance from the dc center point to the annulus. As with the horizontal or vertical spatial frequency, this radial frequency is in units of inverse spatial sample periods.

The dc or zero frequency center point corresponds to the macroscopic average, or gray level, of the dither pattern. Since this datum is already known, it contributes nothing to aiding our interpretation of the nature of the distribution of pixels that make up the dither pattern, and so is omitted in resulting plots. Radial frequency can go as high as $1/\sqrt{2}$ at the high-frequency corners; these high-frequency corners correspond to a checkerboard pattern, the highest 2-D pattern possible.

The power spectrum also needs to be normalized for the gray level, g, which we will specify as ranging from $g = 0$ (black) to $g = 1.0$ (white); normalizing in this way separates gray level from the specific number of bits used in the input image. Spectral energy increases as the number of minority pixels in a bitonal pattern increases, peaking at $g = 0.5$, and

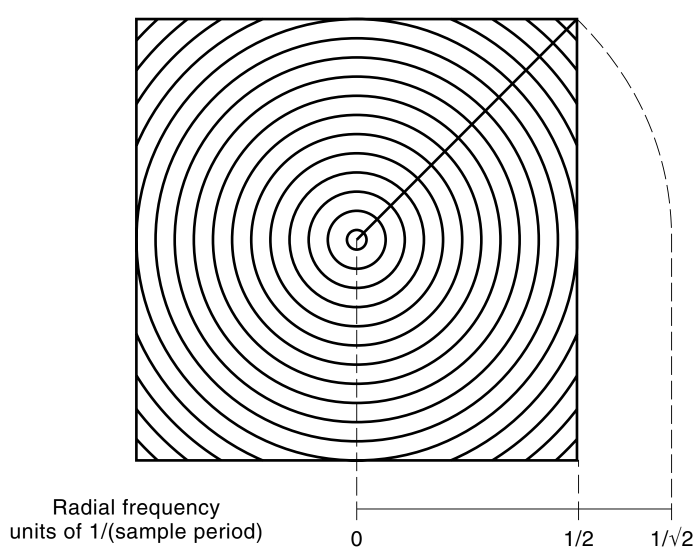


Figure 5. Segmenting the Fourier transform into concentric annuli.

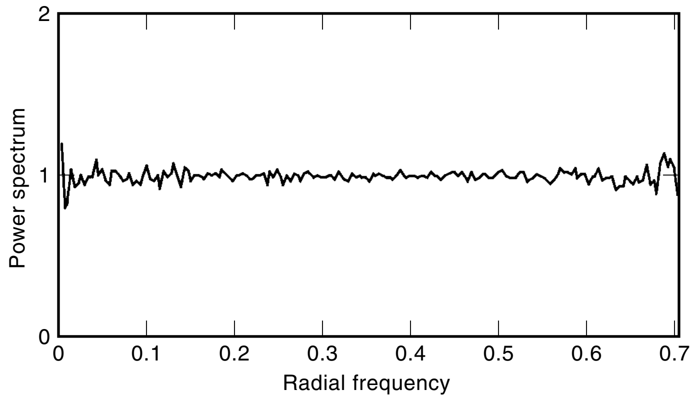


Figure 6. Radially averaged power spectrum for white noise dither patterns for all gray levels.

so the spectral values are divided by the gray level variance (3)

$$\sigma_g^2 = g(1-g)$$

Figure 6 shows the measured radially averaged power spectrum for white-noise patterns. Using the normalization described above, the same plot results for all gray levels, differing only in the small perturbations around 1.0. As expected, dithering with a white-noise threshold produces patterns with a flat spectrum. The term "white noise" falls from the fact that equal energy is present at all wavelengths. This includes energy at low frequencies, which is the cause of the grainy artifacts in Fig. 3. While the higher frequencies tend to be invisible, the arbitrarily long wavelength of the low frequencies can be very noticeable and detract from the image content.

CLUSTERED DOT

Clustered-dot halftones are those that we commonly see in mass hard copy publications produced by offset printing, such as in newspapers, magazines, and books. As opposed to dispersed-dot patterns, such as white noise, where the addressability of each individual pixel is used, the pixels in clustered-dot patterns are nucleated in groups in regular intervals. Figure 7 illustrates an example of this type of halftone. It is important to note that the pixel size of this (and other) examples is much larger than practical printing systems, and is shown as such to allow detailed examination.

Around 1850 the feasibility of a process for printing continuous-tone images was demonstrated by photographing an image through a loosely woven fabric or "screen" placed some distance from the focal plane. It is this process that gave us the word "halftone." It came into practical use in the 1890s when the halftone screen became commercially available, consisting of two ruled glass plates cemented together. In the 1940s the contact screen, a film bearing a properly exposed light distribution of a conventional screen, was introduced.

This screen is sometimes called the graphic arts screen, printer's screen, or classical screen. Even in view of the popularity of dispersed-dot screens, this type of screen is still very important for many forms of printing. In the case of offset printing, there is a minimum size dot that will hold ink, so high-resolution dispersed-dot screens will not work. In electrophotographic (laser) printing a clustered-dot screen is more robust against fluctuations in density.

Figure 7. Result of halftoning with a 4 × 4 macro-cell classical screen.

In Fig. 7 it appears that the screen is oriented at 45 degrees. There is a property of human vision that justifies this. In Fig. 8 a solid 50% gray rectangle is halftoned with (a) 0-degree and (b) 45° clustered-dot screens of the same frequency. Horizontal and vertical lines can be seen in (a), but 45° lines are not as apparent in (b). This is a feature that was recognized in practice since halftone screens were first produced over a hundred years ago. This orientation sensitivity in the frequency response of vision is now well known; the visual system is most acute for orientations at 0 and 90° and least acute at 45° (2).

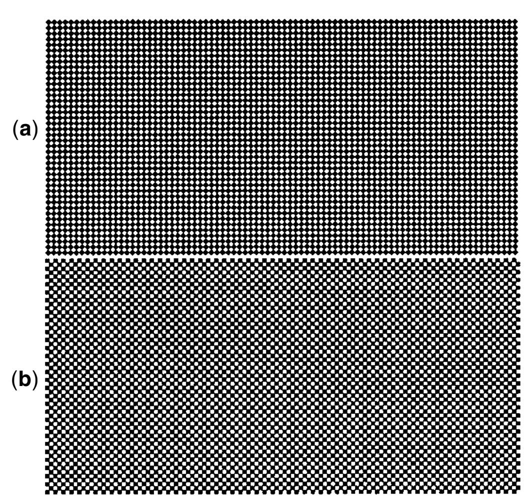


Figure 8. Orientation perception of (a) a 0° screen and (b) a 45° screen.

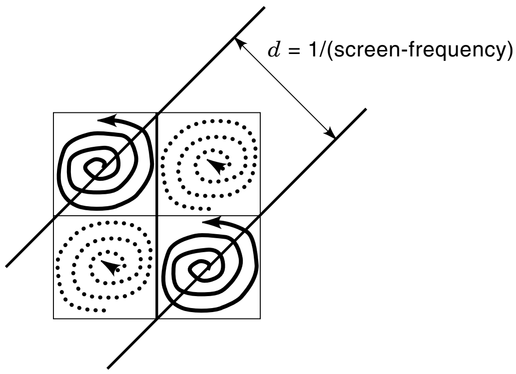


Figure 9. Ordering strategy for digitally generating a classical screen.

To build a dither template that will produce a classical screen digitally, the ordering suggested by Fig. 9 is used. The dither template is a matrix of integers specifying the order in which pixels will be turned "on" as gray level increases. It is periodic and is thus replicated throughout all of two-space. In this figure the array is segmented into four regions. Threshold value ordering begins with at the centers of the dark spirals until half of the array elements are assigned, then continues from the outsides of the dotted spirals until the remain half is assigned.

A characteristic of the classical screen is the screen frequency, usually stated in terms of 45-degree screen-lines per inch (lpi). Newspapers typically use 85 lpi screens, while glossy-paper magazines will use 150 lpi screens. Figure 9 indicates how the screen frequency is related to the dither template. The screen period d depends on the number of pixels in the macro-cells and on the resolution of the final output. By way of example, consider a classical screen with 8 × 8 macro-cells printed on a 1200 dots-per-inch printer. In this case $d = 8\sqrt{2}/1200$, and the screen frequency would be $1/d$ or about 106 lpi.

The dither template for an 8 × 8 macro-cell classical screen is shown in Fig. 10(a), and a 4 × 4 macro-cell screen in (b). These are called "templates" because the actual dither arrays must be normalized to the number of input levels, as detailed in the last section of this article. Figure 7 was generated with the dither template shown in Fig. 10(b).

If we take an 8 × 8 macro cell and use it to dither several fixed gray levels, we can use the radially averaged power spectrum to examine its frequency domain characteristics. A plot of spectra for this case is shown in Fig. 11. As stated earlier, the dc term is omitted because it simply reflects the average gray level and does not contribute to nature of the dither pattern under observation. In this case we see a preponderance of low-frequency energy for all gray levels. This is consistent with what we would expect for a clustered dot.

Recall that in the case of white noise, it was the low frequencies that contributed to graininess. So in this case we might assume that clustered-dot patterns will always suffer from low-frequency textures. It is important to point out that the unit of radial frequency in these plots is inverse sample period. Not counting the dc component, the lowest-frequency component of a periodic dither pattern is a function of the dither template size and the spatial sample period (resolution) of the display device. If the dither template size is small

63	58	50	40	41	51	59	60	64	69	77	87	86	76	68	67
57	33	27	18	19	28	34	52	70	94	100	109	108	99	93	75
49	26	13	11	12	15	29	44	78	101	114	116	115	112	98	83
39	17	4	3	2	9	20	42	88	110	123	124	125	118	107	85
38	16	5	0	1	10	21	43	89	111	122	127	126	117	106	84
48	25	8	6	7	14	30	45	79	102	119	121	120	113	97	82
56	32	24	23	22	31	35	53	71	95	103	104	105	96	92	74
62	55	47	37	36	46	54	61	65	72	80	90	91	81	73	66
64	69	77	87	86	76	68	67	63	58	50	40	41	51	59	60
70	94	100	109	108	99	93	75	57	33	27	18	19	28	34	52
78	101	114	116	115	112	98	83	49	26	13	11	12	15	29	44
88	110	123	124	125	118	107	85	39	17	4	3	2	9	20	42
89	111	122	127	126	117	106	84	38	16	5	0	1	10	21	43
79	102	119	121	120	113	97	82	48	25	8	6	7	14	30	45
71	95	103	104	105	96	92	74	56	32	24	23	22	31	35	53
65	72	80	90	91	81	73	66	62	55	47	37	36	46	54	61

(a)

13	11	12	15	18	20	19	16
4	3	2	9	27	28	29	22
5	0	1	10	26	31	30	21
8	6	7	14	23	25	24	17
18	20	19	16	13	11	12	15
27	28	29	22	4	3	2	9
26	31	30	21	5	0	1	10
23	25	24	17	8	6	7	14

(b)

Figure 10. Dither templates for an (a) 8 × 8 macro-cell and (b) 4 × 4 super-cell classical screen.

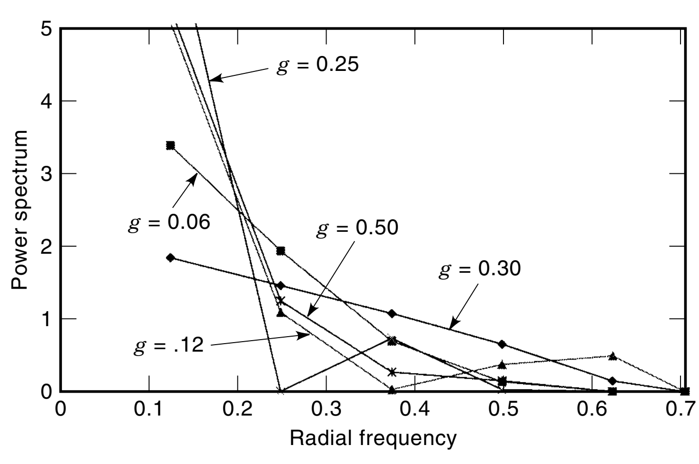


Figure 11. Radial spectra of gray levels dithered with an 8 × 8 macro-cell classical screen.

enough and the spatial sampler period is high enough, then this lowest frequency will still be a fairly high frequency in absolute terms. It is under these conditions that clustered-dot halftoning work well.

In offset color printing, images are produced by overlaying component images in each of four printing inks, CMYK: cyan, magenta, yellow, and black. To avoid moiré patterns, the clustered-dot screen for each of the component images are designed at different angles. The black screen, the most apparent color, is set at 45, the least apparent color, yellow, is set at 0, and cyan and magenta are set at ± 15.

RECURSIVE TESSELLATION

For displays or printers that are not of the highest spatial resolution, it is desirable to use dispersed-dot dither patterns. A historically popular choice for dispersed-dot dither templates have been those that can be generated by the method of recursive tessellation. The goal is to create a template ordering so dither patterns that result from periodically replicating the fundamental period is as homogeneous as possible.

Figure 12 shows the stages in recursively tessellating or tiling the plane to construct the ordering of a 4×4 dither template. In stage $i = 1$, shown in part (a), the fundamental 4×4 period is identified and replicated throughout all of two-space. Note that the top and bottom edges of the gray period shown are copies of each other, as are the left and right edges. We begin at the center position and assign a rank of 0. This selection is periodically replicated. With the goal of homogeneity, the candidate for the next position should be in the center of the voids between the replicated position already assigned. This void center can be found by constructing perpendicular bisectors between nearest neighbors; the corners of the resulting tile are the next candidate, and they are thus labeled with the next rank, 1.

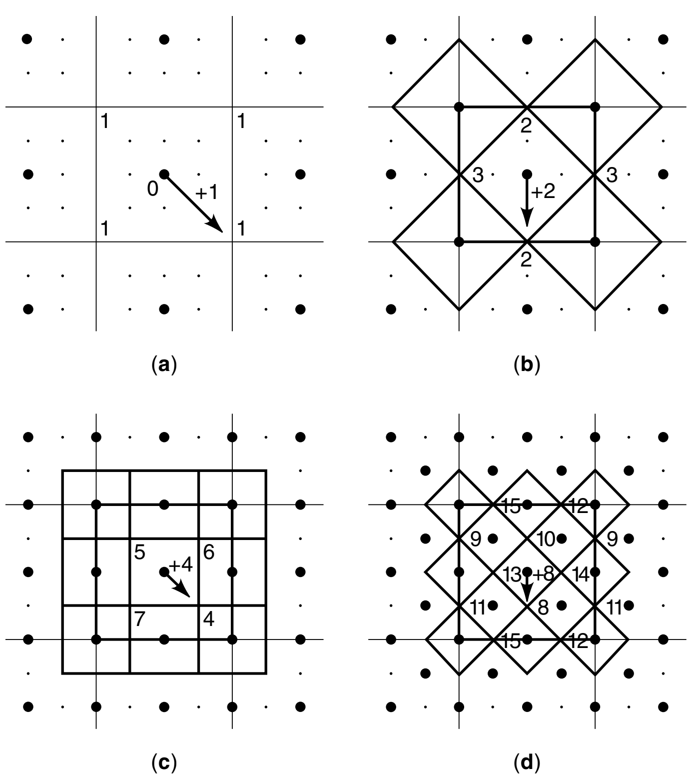


Figure 12. Stages in generating a recursive-tessellation dither template.

In stage $i = 2$, in part (b), void centers are again found at the corners of sub-tiles formed by perpendicularly bisecting nearest neighbors of already assigned points. The ranks of these new corner points are assigned by summing the rank of the point in the center of the sub-tile and 2^{i-1}, where i is the current stage. This summing is depicted in the vector as shown. The direction of the vector from the sub-tile center to a corner can be to any of the four possibilities but must remain fixed from tile to tile.

The process continues for stages $i = 3$, in part (c), and $i = 4$, in part (d), completing the assignment of all 16 elements. The completed 4×4 dither template is shown is Fig. 13(a), and an image dithered with this array is shown in Fig. 14. A recursive tessellation array of this size is said to be fourth order (3). An array of order η will have 2^{η} unique elements.

An eighth-order dither template is shown in Fig. 13(b). A shortcut to forming arrays of a lesser order η is to simply right-shift the binary representations of the elements in the eighth-order array by $(8 - \eta)$ bits. Arrays with more unique elements can render more gray levels but will also introduce larger periods, and thus longer wavelength patterns.

Dithering with this type of dither template is referred to as Bayer's dither, in reference to his famous 1973 proof of

0	14	3	13
8	4	11	7
2	12	1	15
10	6	9	5

(**a**)

0	234	58	218	14	230	54	214	3	233	57	217	13	229	53	213
128	64	186	122	142	78	182	118	131	67	185	121	141	77	181	117
32	192	16	250	46	206	30	246	35	192	19	249	45	205	29	245
160	96	144	80	174	110	158	94	163	99	147	83	173	109	157	93
8	224	48	208	4	238	62	222	11	227	51	211	7	237	61	221
136	72	176	112	132	68	190	126	139	75	179	115	135	71	189	125
40	200	24	240	36	196	20	254	43	203	27	243	39	199	23	253
168	104	152	88	164	100	148	84	171	107	155	91	167	103	151	87
2	232	56	216	12	228	52	212	1	235	59	219	15	231	55	215
130	66	184	120	140	76	180	116	129	65	187	123	143	79	183	119
34	194	18	248	44	204	28	244	33	193	17	251	47	207	31	247
162	98	146	82	172	108	156	92	161	97	145	81	175	111	159	95
10	226	50	210	6	236	60	220	9	225	49	209	5	239	63	223
138	74	178	144	134	70	188	124	137	73	177	113	133	69	191	127
42	202	26	242	38	198	22	252	41	201	25	241	37	197	21	255
170	106	154	90	166	102	150	86	169	105	153	89	165	101	149	85

(**b**)

Figure 13. Values of (a) a fourth-order recursive tessellation template and (b) an eighth-order template.

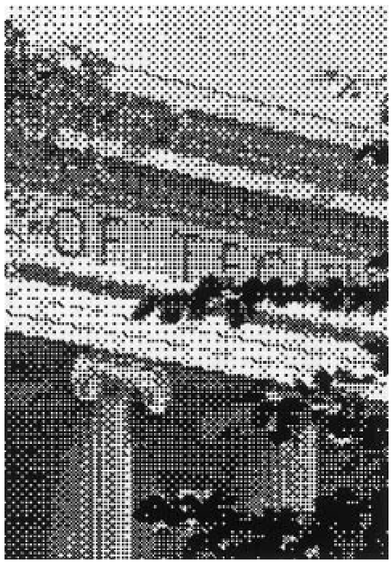


Figure 14. Dithering with a fourth-order recursive tessellation array.

optimality (4). Also, dithering with these arrays, as well as those used for clustered-dot halftoning, are part of a larger genre referred to as ordered dither. Ordered dither is the name given to any dither process that uses a period deterministic dither array. It is the "ordered" nature of the elements in the array that contrast it with the random nature of white-noise dithering.

BLUE NOISE

Engineers like to describe various types of noise with color names. The most well-known example is "white noise," so named because its power spectrum is flat across all frequencies, much like the visible frequencies of light. "Pink noise" is used to describe low-frequency white noise, the power spectrum of which is flat out to some finite high-frequency limit. The spectrum associated with Brownian motion is (perhaps whimsically) referred to as "brown noise."

Blue noise is the high-frequency complement of pink noise. As we have seen, when it comes to producing good-quality halftone patterns, low-frequency energy is the enemy. In this section the concept of the blue-noise metric is described, along with neighborhood and point processes that generate blue-noise dither patterns. Since its introduction (5) the blue-noise concept has become an important part of halftoning research.

Ideally a well-formed dither pattern should have the unstructured nature of white noise without the low-frequency textures. Consider the problem of rendering a fixed gray level, g, with binary pixels whose vertical and horizontal pixel period, or separation, is S. The goal is to distribute the binary pixels as homogeneously as possible. These pixels would be separated by an average distance in two dimensions. This distance is called the principal wavelength and for this square pixel case would have the value

$$\lambda_g = \begin{cases} \dfrac{S}{\sqrt{g}}, & g \le \dfrac{1}{2} \\ \dfrac{S}{\sqrt{1-g}}, & g > \dfrac{1}{2} \end{cases}$$

Since the distribution is assumed to be homogeneous, the corresponding power spectrum would be radially symmetric. The principal wavelength would be manifested as a principal frequency, $f_g = 1/\lambda_g$.

Figure 15 exemplifies the radially averaged power spectrum of a well-formed blue-noise dither pattern for a fixed gray level. There are three important features. The pattern should consist of an isotropic field of binary pixels with an average separation of λ_g. This corresponds to a peak of energy at the principal frequency (a). The average separation should vary in an uncorrelated white-noise-like manner, but unlike white noise the wavelengths of this variation must not be significantly longer than λ_g. So other key features of a blue noise spectrum are (b) the sharp cutoff below the principal frequency, and (c) a flat white-noise-like spectrum above the principle frequency.

Neighborhood Processes

In image processing, a point process is an operation that uses as its only input the value of the current pixel; a neighborhood process uses as input the value of the current pixel along with values of pixels surrounding it. One neighborhood process that attempts to generate blue noise is the error diffusion algorithm. A graphical illustration of the algorithm is shown in Fig. 16(a). Assuming that the input signal varies from $g = 0$ (black) to $g = 1.0$ (white), the threshold block simple sets the output to 0 for values less than $\frac{1}{2}$ and to 1 for values greater than or equal to $\frac{1}{2}$. The binary output signal is subtracted from the prethreshold signal to form an error. This error is "diffused" into yet to be considered input values as governed by the error filter. The signal consisting of past error values is passed through this filter to produce a correction factor to be added to future input values.

This algorithm was first introduced by Floyd and Steinberg (6), who also proposed the error filter shown in Fig. 16(b). The algorithm processes pixels in a raster order, so the only nonzero filter elements are those in front of and below the current pixel. As with all error filters, the elements must sum to one. An image dithered by error diffusion with this filter is shown in Fig. 17.

Several larger error filters have been proposed that appear to create better looking images. However, upon inspection,

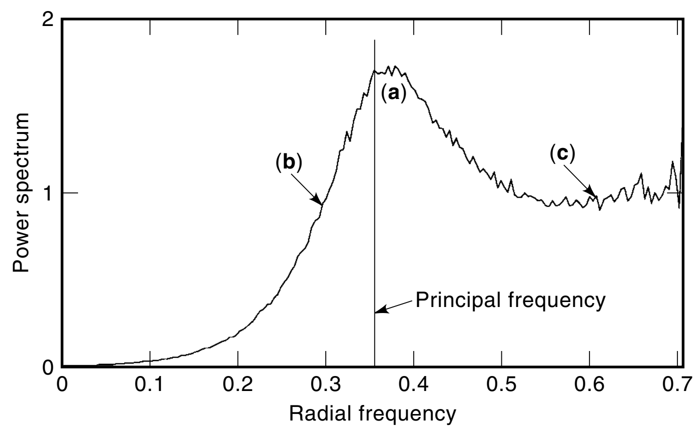


Figure 15. Spectra Characteristics of a blue-noise dither pattern: (a) Energy peak at principal frequency; (b) sharp low-frequency cutoff; and (c) high frequency white noise.

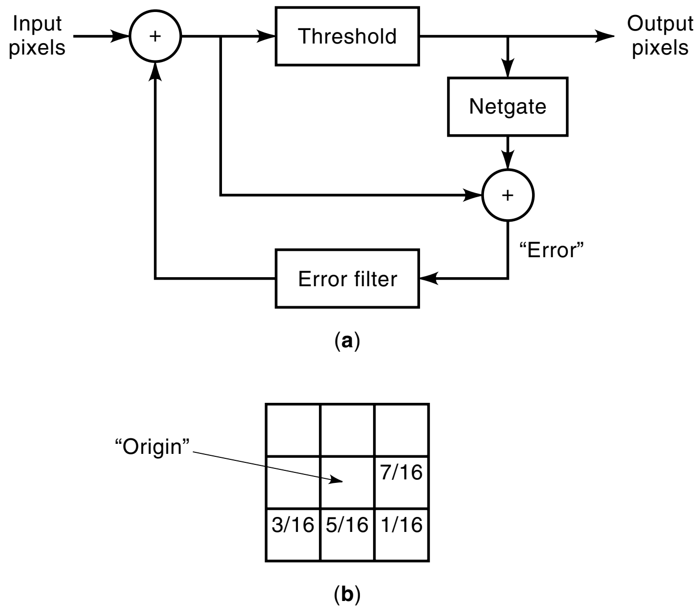


Figure 16. (a) The error diffusion algorithm; (b) error filter identified by Floyd and Steinberg.

they look better because the filters tend to sharpen more; areas of flat gray are in fact less homogeneous than the original four-element filter. Using error diffusion as a sharpening means is less desirable than employing an independent presharpening step because there is no control over the degree of sharpening. The effective sharpening filter intrinsic to the error diffusion algorithm has been measured (7); besides being hard to control, it is also found to be very asymmetric.

Many of the gray level patterns that result from the error diffusion algorithm with the Floyd and Steinberg error filter suffer from directional "worm" artifacts. Also disturbing distortions occur for other patterns, particularly those close to the perfect checkerboard at 50% gray. These anisotropic features are reflected in the radially averaged power spectrum. These shortcomings can be overcome by incorporating two changes in the traditional error diffusion algorithm. First, processing the input on a serpentine raster can break up most of the directional artifacts, and second, adding some random perturbations to the weights in the error filter can further break up stable structures that would otherwise be manifested as unwanted texture.

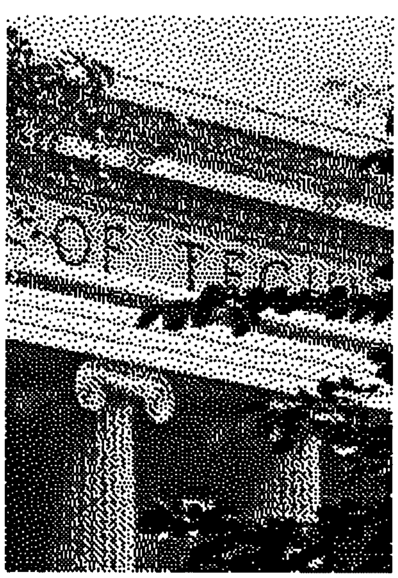


Figure 17. Error diffusion dithering using the error filter identified by Floyd and Steinberg.

Figure 18. Improved error diffusion using a serpentine raster and a "noisy" weight filter.

The result of this modified error diffusion is shown in Fig. 18. Gray levels for this process exhibit well-behaved blue-noise properties, as are plotted in Fig. 19. Each of the plots have a peak at the principal frequency for the respective gray level. As gray level increases from 0.03 to 0.50; the distribution of spectral energy moves like a wave crashing into the high-frequency wall, then bounces back in the same way above 0.50. The plots for gray levels g above 0.50 are very similar to those for $(1 - g)$ but not explicitly drawn to avoid graphic confusion.

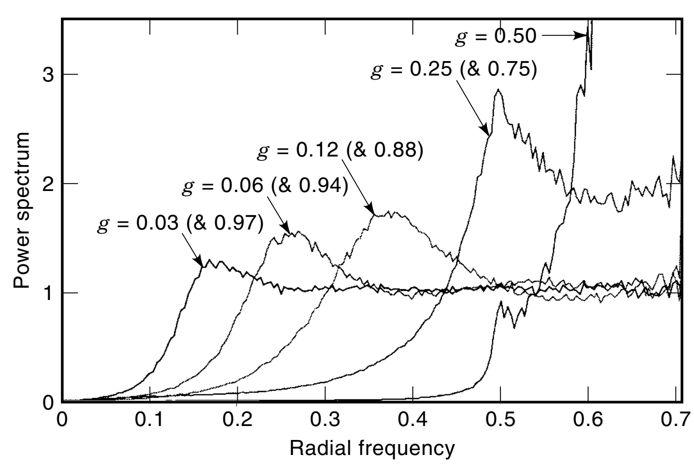


Figure 19. Radial spectra for various gray levels for the blue-noise process used to generate Fig. 18.

Point Processes

Blue-noise dithering can also be achieved with the point process of ordered dither. The trick of course is using an appropriate dither array. Because of the implementation advantages of ordered dither over neighborhood processes, this has become an active area of research. In the printing industry, ordered dither arrays used for this purpose are often referred to as "stochastic screens."

An overview of approaches to generating blue-noise dither templates is presented in (8). One approach would be to build a template by directly shaping the spectrum of binary patterns by an iterative process so as to force blue-noise characteristics (9).

A very straightforward and effective approach to generating relatively small dither templates of this type is the Void-and-Cluster algorithm (10), and it will be outlined here. As with all ordered dither, the array and resulting binary patterns are periodic. Figure 20 illustrates this periodicity. This algorithm looks for voids and clusters in prototype binary patterns by applying a void- or cluster-finding filter at the area under consideration. Because of this implied periodicity, a filter extent will effectively wrap around as shown.

Binary image patterns are made up of pixels with one of two states that can be thought of as either "1" or "0", or "black" or "white." Except for the special case where there is exactly the same number of pixels of each state, there will always be more pixels of one state ("majority pixels") than the other ("minority pixels") in any given binary pattern. A void-finding filter considers the neighborhood around every majority pixel in a prototype binary pattern, and a cluster-finding filter considers the neighborhood around every minority pixel. The algorithm uses these filters to identify the biggest void or tightest cluster in the pattern.

We start by relaxing an arbitrary initial pattern to form one that is homogeneously distributed. In Fig. 21(a) a 16 × 16 binary pattern is shown with 26 minority pixels randomly positioned. The purpose of the algorithm is to move minority pixels from tight clusters into large voids. With each iteration the voids should be smaller and the clusters looser. This is done one pixel move at a time until both the voids stop getting smaller, and the clusters stop getting looser. It turns out that the condition of convergence is quite simple; processing is complete when removing the pixel from the tightest cluster creates the largest void.

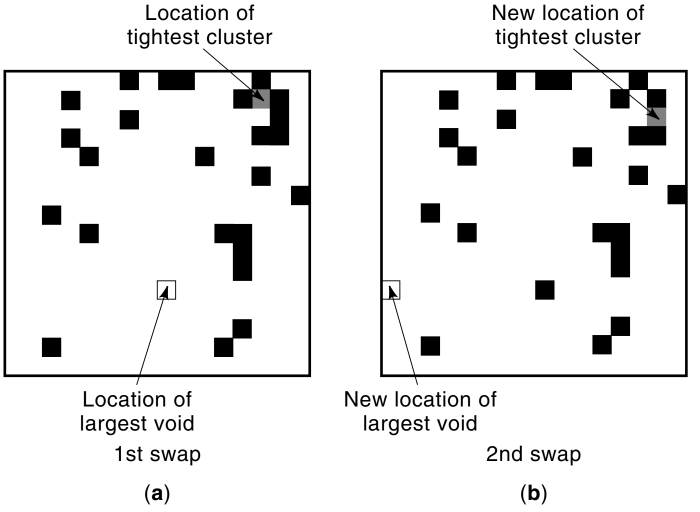


Figure 21. Example of the first two iterations of the initial binary pattern generator. The 16 × 16 input pattern has 26 minority pixels.

The minority pixel in the tightest cluster and the majority pixel in the largest void are identified in Fig. 21(a). After the first iteration the minority pixel in the tightest cluster is moved to the largest void, resulting in the pattern shown in Fig. 21(b). Once again, the locations of the new tightest cluster and new largest void are identified. It should be noted that it is entirely possible for minority pixels to be moved more than once; the search for voids and clusters at each iteration is independent of past moves.

The results of this example are summarized in Fig. 22 where 12 iterations were needed before convergence. Four periods are shown of both the (a) input pattern, and (b) the relaxed or rearranged pattern to illustrate the wraparound or edge-abutting consequences of tiling two-space with such patterns. Note how homogeneously distributed the resulting pattern is.

Next, starting with this relaxed pattern as a starting point, a dither template is ordered in parallel. Elements of increas-

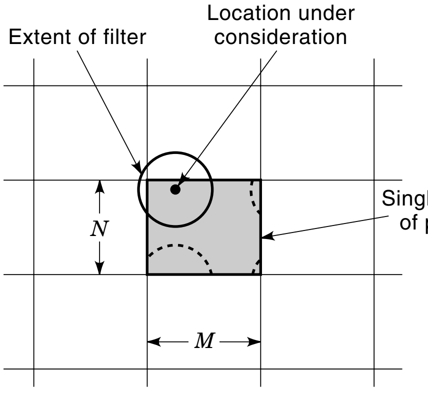


Figure 20. Two-dimensional periodicity of ordered dither pattern, and wraparound property of void-and-cluster-finding filters.

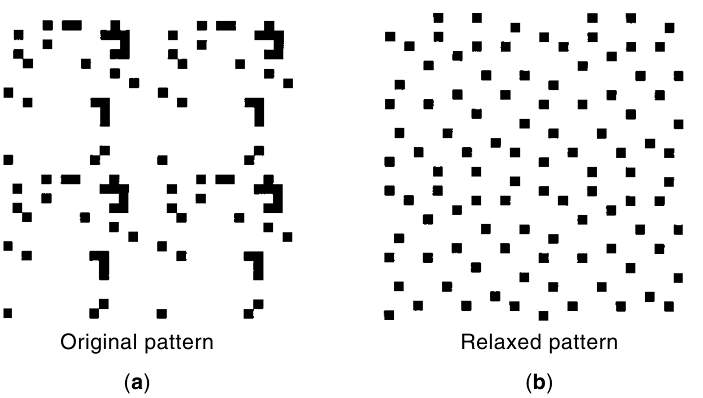


Figure 22. Result of the initial binary pattern generator. Four periods of the 16 × 16 input pattern (a) and the rearranged, or relaxed, pattern (b) are shown to illustrate the wraparound properties. The input pattern is the same as that shown in Fig. 21.

Figure 23. Dithering with a 32 × 32 void-and-cluster array.

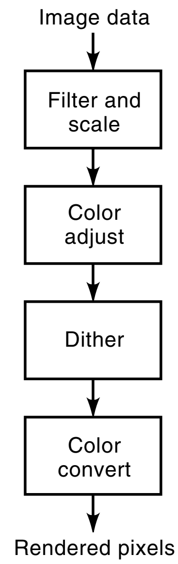


Figure 24. Stages of an image-rendering system.

ing value in the dither template are entered as minority pixels are inserted into the voids. Then returning to this starting pattern, elements of decreasing value are entered as minority pixels are removed from the tightest clusters.

Figure 23 shows the result of dithering an image with a 32 × 32 void-and-cluster generated dither array. It should be noted that the image does not appear as sharp as those produced by error diffusion. As mentioned earlier, the added runtime complexity of error diffusion does afford the side benefit of serving as a sharpening filter, even if uncontrollable. As will be shown in the next section, a prefilter as part of a rendering system can make up for this.

It is interesting to note that if a completely empty (all white) pattern is used as a starting point, this algorithm will generate recursive tessellation dither templates. This will in fact also result if the starting point is any of the recursive tessellation patterns.

RENDERING SYSTEMS

The goal of an image-rendering system, of which halftoning is a part, is to take device-independent image date and tailor it to a target display. Figure 24 illustrates the major phases of a image-rendering system: (1) filter and scale, (2) color adjust, (3) dither, and (4) color space convert.

In the first stage the original image data must be resampled to match the target window or page size. Scaling should be independent in each dimension to allow for asymmetric pixel aspect ratios in either the source data or the target display. A band-limiting filter should be used for reductions, and an interpolating filter should be used for enlargements.

Sharpening can also occur in this stage. A typical sharpening scheme can be expressed by the following equation:

$$I_{\text{sharp}}[x, y] = I[x, y] - \beta\Psi[x, y] * I[x, y]$$

where $I(x, y)$ is the input image, $\Psi(x, y)$ is a digital Laplacian filter, and * is the convolution operator. The nonnegative parameter βcontrols the degree of sharpness, with $\beta = 0$ indicating no change in sharpness. When enlarging, sharpening should occur before scaling , and when reducing, sharpening should take place after scaling. To illustrate the effect of sharpening, Fig. 25 shows an image that was presharpened with a sharpening factor of $\beta = 2.0$, then dithered using the same process as that of Fig. 23.

The second stage of rendering is color adjust, most easily achieved with a lookup table (LUT). In the case of color images, each color component can use a separate adjust LUT. In the case of luminance-chrominance color, an adjust LUT for the luminance component controls contrast and brightness, and LUTs for the chrominance components control saturation.

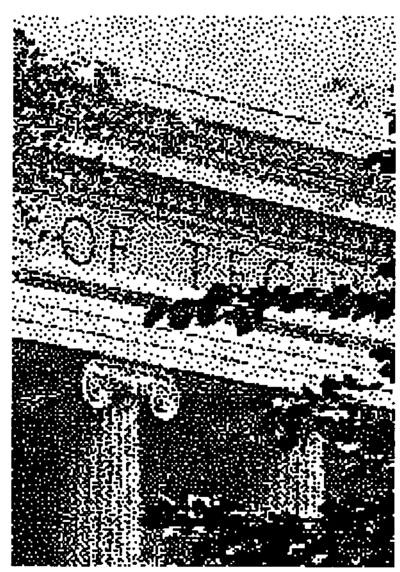


Figure 25. Result of presharpening the input image with a β-2.0 Laplacian, prior to dithering. The dithering process is the same as that used in Fig. 23.

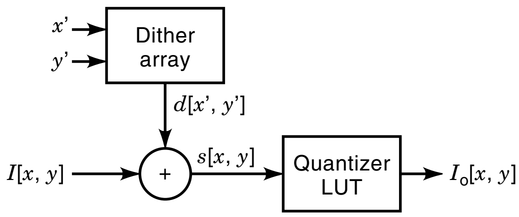


Figure 26. Dithering system with 2 LUTs.

Multilevel Dithering

The third stage of Fig. 24 is dithering, the focus of this article. While hard copy display products are capable of marking the color component of a pixel as either on or off, video products generate many levels. When an ordered dither array is used, such as that generated by the method of recursive tessellation, void-and-cluster, or other methods, multilevel dithering can be implemented very efficiently. This section details a means for doing this with minimum hardware or software, yet guarantees output that preserves the mean of the input. It allows dithering from any number of input levels N_{i} to any number of output levels N_{o}, provided $N_{\mathrm{i}} \geq N_{\mathrm{o}}$. Note that N_{i} and N_{o} are not restricted to be powers of two.

Each color component is treated as an independent image. The input image I_i can have integer values between 0 and $(N_{\mathrm{i}} - 1)$, and the output image I_o can have integer values between 0 and $(N_{\mathrm{o}} - 1)$. A deterministic dither array of size $M \times N$ is used. To simplify addressing of this array, M and N should each be a power of two. A dither template (e.g., that in Fig. 13) defines the order in which dither values are arranged. The elements of the dither Template T have integer values between 0 and $(N_{\mathrm{t}} - 1)$, where N_{t} is the number of template levels, which represent the levels against which image input values are compared to determine their mapping to the output values. The dither template is central to determining the nature of the resulting dither patterns.

Figure 26 shows a dithering system that comprises two memories and an adder. The system takes an input level I_i at image location $[x, y]$ and produces output level I_o at the corresponding location in the dithered output image. The dither array is addressed by x' and y', which represent the low-order bits of the image address. The selected dither value $d[x', y']$ is added to the input level to produce the sum s. This sum is then quantized by addressing the quantizer LUT to produce the output level I_o.

The trick to achieving mean-preserving dithering is to properly generate the LUT values. The dither array is a normalized version of the dither template specified as follows:

$$d[x', y'] = \mathrm{int}\{\Delta_{\mathrm{d}}(T[x', y'] + \tfrac{1}{2}\}$$

where int{ } is integer truncation, Δ_{d}, the step size between normalized dither values, is defined as $\Delta_{\mathrm{d}} = \Delta_{\mathrm{Q}}/N_{\mathrm{t}}$, and Δ_{Q} is the quantizer step size

$$\Delta_{\mathrm{Q}} = \frac{N_{\mathrm{i}} - 1}{N_{\mathrm{o}} - 1}$$

Note that Δ_{Q} also defines the range of dither values. The quantizer LUT is a uniform quantizer with N_{o} equal steps of size Δ_{Q}.

Using the above expressions, it is possible to simplify the system by exchanging one degree of freedom for another. A bit shifter can replace the quantizer LUT at the expense of forcing the number of input levels N_{i}, to be set by the system, at least internally. Here we differentiate between input to the dithering system and "real" or "raw" input levels, I_{r}. If an adjust LUT is used to modify the image data, including a gain makes a "modified adjust LUT." Figure 27 depicts such a system when I_{r} is the raw input level which can have integer values between 0 and $(N_{\mathrm{r}} - 1)$, where N_{r} is the number of raw input levels, typically 256. Therefore the modified adjust LUT must impart a gain of $(N_{\mathrm{i}} - 1)/(N_{\mathrm{r}} - 1)$. The value of N_{i} has yet to be found.

The quantizer LUT can be replaced by a simple R-bit right-shifter if the variable Δ_{Q} can be forced to be an exact power of two: $\Delta_{\mathrm{Q}} = 2^R$. Using the fact that $\Delta_{\mathrm{Q}} = (N_{\mathrm{i}} - 1)/(N_{\mathrm{o}} - 1)$, N_{i} can be set by the expression

$$N_{\mathrm{i}} = (N_{\mathrm{o}} - 1)2^R + 1$$

and the value of R can be shown (11) to be

$$R = \mathrm{int}\left\{\log_2\left(\frac{2^b - 1}{N_{\mathrm{o}} - 1}\right)\right\}$$

where b specifies the number of bits with which the input levels I_i are to be represented.

As an example, consider the case where N_{o} equals 87 (levels), b equals 9 (bits), N_{t} equals 1024 (levels, for a 32×32 template), and N_r equals 256 (levels). Thus R equals 2, meaning that the R-bit shifter drops the least-significant 2 bits. N_{i} equals 345 (levels); the dither array is normalized by $d[x', y'] = \mathrm{int}\{\Delta_{\mathrm{d}}(T[x', y'] + \tfrac{1}{2})\}$ with $\Delta_{\mathrm{d}} = 1/256$; and the gain factor to be included in the modified adjust LUT is 344/255. These data are loaded into the system represented by Fig. 27 and uniformly map input pixels across the 87 true output levels, giving the illusion of 256 levels.

Bitonal Dithering. Another important simplifying example is the hard copy case when N_{o} is 2. Consider using the system of Fig. 26 with no adjust LUT. Since there would be only one quantization level, the adder and quantizer LUT could be re-

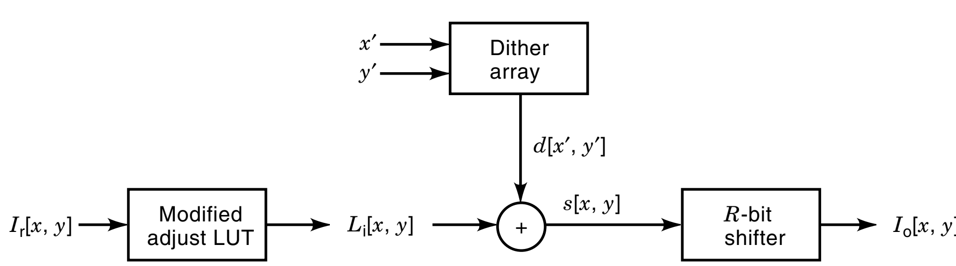


Figure 27. One memory dithering system with adjust LUT and bit shifter.

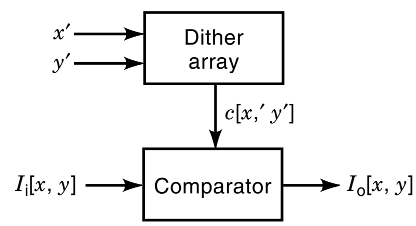


Figure 28. Bitonal dithering system using a comparator.

placed by a comparator as shown in Fig. 28. Here the dither array is further normalized to incorporate the quantizer threshold:

$$c[x', y'] = (N_i - 1) - \mathrm{int}\{\Delta_\mathrm{d}(T[x', y'] + \tfrac{1}{2})\}$$

and the comparator outputs a "1" when the condition $I_\mathrm{i} \geq c[x', y']$ and 0 otherwise. By way of example, suppose that a small dither template with $N_\mathrm{t} = 16$ levels is used, and $N_\mathrm{i} = 256$ input levels. Δ_d would equal 255/16, and the system of Fig. 28 would yield a perfectly uniform, macroscopically mean-preserved representation of the input.

Color Conversion

Referring once again to Fig. 24, consider the final rendering subsystem—color-space convert. In the case of video rendering, a frame buffer that is expecting *RGB* data will not need to convert the color space if the source data are already represented in *RGB,* as is the case of graphics generation systems. However, uncompressed motion video is essentially always transmitted and stored in a luminance-chrominance space. Although the chomaticities of the *RGB* primaries of the major video standards vary slightly, the luminance-chrominance space is always *YUV*. *Y* represents the achromatic component that is loosely called luminance. *U* and *V* are the chrominance components.

Figure 29 shows the parallelepiped of "feasible" *RGB* triplets in the *YUV* coordinate system. Feasible *RGB* points are those that are nonnegative and are not greater than the maximum supported value. *RGB* and *YUV* values are linearly related and can be interconverted by means of a 3 × 3 matrix multiply.

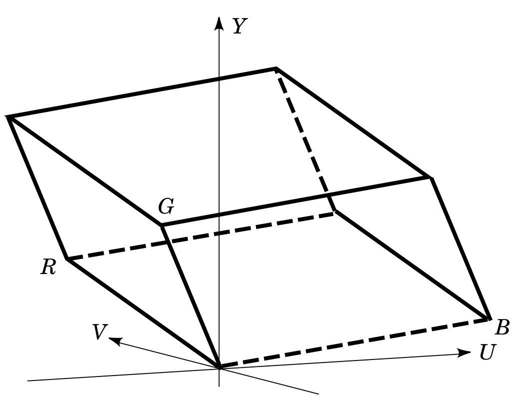


Figure 29. Feasible *RGB* parallelepiped in *YUV* space.

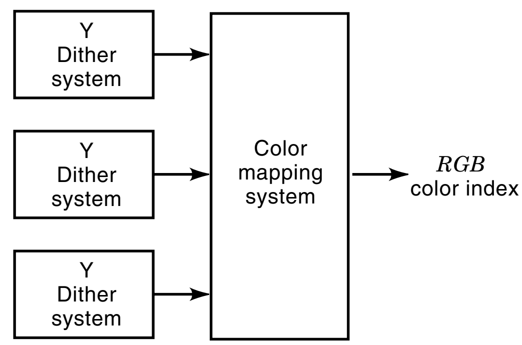


Figure 30. System for dithering three color components and color mapping the collective result.

Figure 30 shows the back end of a rendering system that dithers *Y*, *U*, and *V* color components prior to color space conversion. A serendipitous consequence of dithering is that color space conversion can be achieved by means of table lookup. The collective address formed by the dithered *Y*, *U*, and *V* values is small enough to require a reasonably sized color mapping LUT. There are two advantages to this approach. First, a costly dematrixing operation is not required, and second, infeasible *RGB* values can be intelligently mapped (12) back to feasible space off-line during the generation of the color mapping LUT.

This second advantage is an important one, because 77% of the valid *YUV* coordinates are in invalid *RGB* space, that is, in the space around the *RGB* parallelepiped in Fig. 29. Color adjustments such as increasing the brightness or saturation can push otherwise valid *RGB* values into infeasible space. In traditional systems that perform color conversion by dematrixing, out-of-bounds *RGB* values are simply truncated; this can change the color in an undesirable way.

BIBLIOGRAPHY

1. W. M. Goodall, Television by pulse code modulation, *Bell Syst. Tech. J.,* **30**: 33–49, 1951.
2. F. W. Campbell, J. J. Kulikowski, and J. Levinson, The effect of orientation on the visual resolution of gratings, *J. Physiol. London,* **187**: 427–436, 1966.
3. R. Ulichney, *Digital Halftoning,* Cambridge, MA: MIT Press, 1987.
4. B. E. Bayer, An optimum method for two level rendition of continuous-tone pictures, *Proc. IEEE Int. Conf. Commun., Conf. Rec.,* 1973, pp. 26-11–26-15.
5. R. Ulichney, Dithering with blue noise, *Proc. IEEE,* **76**: 56–79, 1988.
6. R. W. Floyd and L. Steinberg, An adaptive algorithm for spatial grayscale, *Proc. SID,* **17** (2): 75–77, 1976.
7. K. T. Knox, Edge enhancement in error diffusion, *SPSE's 42nd Annu. Conf.,* 1989, pp. 310–313.
8. K. Spaulding, R. Miller, and J. Schildkraut, Methods for generating blue-noise dither matrices for digital halftoning, *J. Electron. Imaging,* **6** (2): 208–230, 1997.
9. T. Mitsa and K. J. Parker, Digital halftoning technique using a blue-noise-mask, *J. Opt. Soc. Amer. A, Opt. Image Sci.,* **9**: 1920–1929, 1992.

10. R. Ulichney, The Void-and-Cluster method for generating dither arrays, *IS&T/SPIE Symp. Electron. Imaging Sci. Technol.,* San Jose, CA, **1913**: 1993, pp. 332–343.
11. R. Ulichney, Video rendering, *Dig. Tech. J.,* **5** (2): 9–18, 1993.
12. V. Bahl, P. Gauthier, and R. Ulichney, Software-only compression, rendering, and playback of digital video, *Dig. Tech. J.,* **7** (4): 52–75, 1996.

ROBERT ULICHNEY
Compaq Computer Corporation