Dithering with Pascal Cellular Automaton

Yvan Richard[0000−0003−4497−3843]

govyvan@hotmail.com
http://www.linkedin.com/in/yvanrichard

Abstract. In this article, I present two versions of a cellular automaton (CA) that evolve according to a set of rules derived from a well-known combinatorial structure: Pascal's Triangle. These CAs produce point sampling commonly observed in high-quality digital dithering and halftoning techniques. The first version is a probabilistic cellular automaton (PCA), which samples discrete values from Pascal's Triangle as a probability distribution. The second version is deterministic, where tone-dependent lattice paths are selected from Pascal's Triangle by analyzing the local points distribution in the halftone pattern. This deterministic version also offers significant computational gains over its probabilistic counterpart.

Keywords: Halftoning and Dithering · Cellular Automata · Error Diffusion · Image Rendering and Quality · Pascal Triangle · Lattice Paths · Quantization · Discrete Sampling

1 Introduction and Overview

I found that a simple Cellular Automata (CA) coupled with — Pascal's Triangle — as probabilistic rules, was able to produce high-quality halftone images. These images presents a point sampling signature that effectively simulates the pleasing texture and detail typically achieved with Frequency Modulation (FM) screening and specifically, Error Diffusion (ED) techniques. While cellular automata and error diffusion operate on different principles, they both share similarities in their spatial and temporal dependencies but also in their predictability, to some extent.

In cellular automata, each cell's state depends on its neighbours, showing spatial dependencies, and the system evolves in discrete steps according to predefined rules, indicating temporal dependencies. Similarly, in error diffusion, pixel quantization error spreads to neighbour pixels, showing spatial dependencies, and the process iterates over the image in a stepwise manner, with each pixel's quantization affecting subsequent ones, demonstrating temporal dependencies.

In general, cellular automata are governed by a set of predefined rules leading to a deterministic evolution. However, by introducing element of randomness or probability into their ruleset, probabilistic cellular automata (PCA) exhibit

more complex dynamics and simulate stochastic processes with greater fidelity. Similarly, error diffusion algorithms are usually deterministic, but the addition of random noise to the quantization error or diffusion weights reduce the visibility of quantization artefacts, leading to more visually pleasing outputs.

However, beyond these spatial and temporal similarities, cellular automata and error diffusion differs in a fundamental aspect, the way data are quantified. Typically, cellular automata computations are performed using binary or integer values, reflecting the discrete nature of the states. Whereas error diffusion utilizes fractional weights to spread the quantization error of pixel values across neighbour pixels, resulting in continuous floating-point data calculations.

By recognising these characteristics, we can effectively assess how this novel approach utilising Cellular Automata and Pascal's Triangle compares to the established effectiveness of Error-Diffusion technics in generating high-quality halftone images. The combination of these elements in the context of digital dithering and halftoning not only enhances our current understanding but also opens up further research opportunities in both digital imaging techniques and computational theory.

1.1 Error Diffusion

Error diffusion is a digital image processing technique commonly employed in halftoning — a process that converts continuous-tone images into binary images while approximating the original appearance. It is classified as a frequency modulated (FM) dithering technique that produces halftone images with higher spatial resolution and superior image quality by varying the pixel density (frequency) based on the pixel values of the underlying image. Widely embraced across industries such as printing, publishing, and display technologies, error diffusion methods are pivotal in generating high-quality halftone images and find application in devices like inkjet printers and e-ink displays.

Error-Diffusion can be described as follows: the algorithm scans the image, row by row, and pixel by pixel, the current pixel value plus a feedback error is compared to a threshold, the result of the comparison assigns an output value, e.g. 0 or 1 to the halftone image and the quantization error is distributed to neighbour pixels using predefined coefficients. By distributing the error across the image, the error diffusion process tends to smooth out abrupt transitions between different intensity levels, maintaining the overall tonal balance and detail in the final output. Below are the Floyd and Steinberg coefficients, the ($\square$) indicates the currently processed pixel and ($-$) indicates the previous pixel.

$$\frac{1}{16}\begin{bmatrix} - & \square & 7 \\ 3 & 5 & 1 \end{bmatrix}$$

Over the past decades, researchers and specialists have explored numerous variations and extensions of error diffusion algorithms to address specific chal-

lenges, such as reducing artefacts, improving performance, and adapting to different types of input images and outputs, encompassing multilevel, color and tone dependencies. These variations include modifications to the diffusion coefficients, error weighting schemes, scan path, threshold modulation, noise shaping and incorporation of perceptual models, leading to a substantial body of literature, articles and patent filings, [1–12].

Blue-Noise — Poisson Disc Sampling To analyze the local points distribution in the halftone pattern, Ulichney [3] introduces a statistical tool in the frequency domain called the Radially Averaged Power Spectrum density or RAPSD [3]. This tool helps to identify blue-noise point sampling, which represents an "ideal" power spectrum for dither patterns (Fig. 1). Unlike other dithering patterns, such as ordered or clustered-dot dithering, a blue-noise dither pattern is a spatial arrangement of pixel values that minimizes low-frequency structures, meaning that any regular or repetitive patterns are avoided. This results in a more natural and visually pleasing pattern.

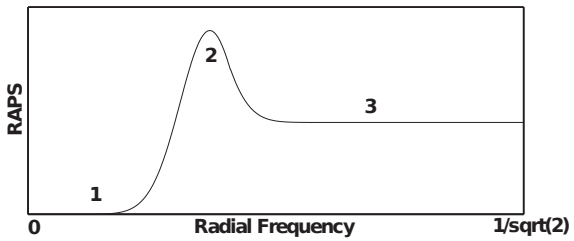
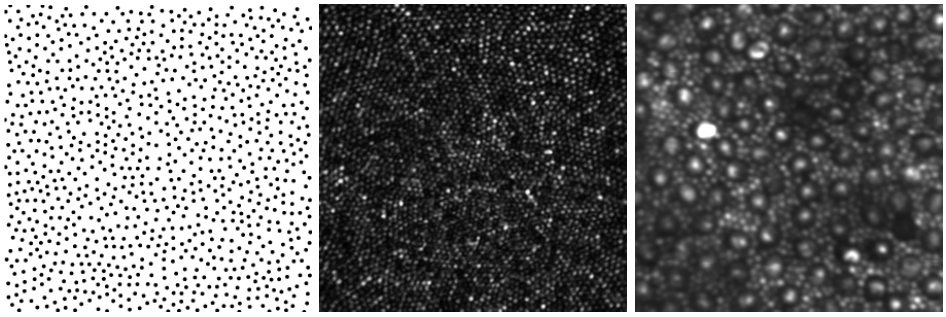


Fig. 1: Ideal Radially Averaged Power Spectrum of a blue-noise halftone pattern: (1) Pulse at origin (DC component = 0), (2) surrounded by a peak at the principal frequency and (3) surrounded by white noise.

The concept of blue-noise dithering is closely related to Poisson Disc Sampling in the field of statistical sampling, which also aims to distribute points (or samples) with a minimum distance between them. It is generally used as an antialiasing technique, where unwanted artefacts introduced by regular sampling can be attenuated by non-uniform sampling [13] and [14].

Indeed, Yellott [15] discovered in 1984 that the cone photoreceptors in the mammalian rhesus retina are arranged following a Poisson disc distribution. Recently, the first photographs of rod receptors from a living human eye (Fig. 2.b and 2.c) have been taken using a noninvasive imaging technique [16].

A typical Poisson disc sampling algorithm adds points randomly, at each new added point it checks that the distance to neighbours points is beyond a fixed threshold, otherwise the point is rejected. The algorithm iterates until it can no longer add points (Fig 2.a). A Comparison of methods for generating Poisson Disk Distributions is presented in [17].



(a) (b) (c)

Fig. 2: Poisson Disc Sampling Generated generated by Dart throwing (a), Human Cones Photoreceptors mosaic at the foveal center (b), Human Cones and Rods Photoreceptors mosaic at the foveal center - cones are the bigger, brighter points, while rods are the little dots that surround them (c) “Courtesy of the Dubra lab at the Medical College of Wisconsin."

1.2 Cellular Automata

Cellular automata were first introduced and developed by mathematicians John von Neumann [18] but it was the work of John Conway [19] that brought this computational model into the spotlight with his creation of the Game of Life. Since then, numerous researchers have contributed to the advancement of cellular automata theory and applications over the years. Stephen Wolfram extensively studied and popularized them in his book "A New Kind of Science" [20].

Typically, they consist of a grid of cells, each of which in a finite number of states. The state of each cell evolves over discrete time steps based on a set of rules that determine how the state of a cell changes based on its current state and the states of its neighbour cells.

Cellular automata can be classified into two main types:

- In deterministic cellular automata, the rules for state transitions are strictly defined and do not involve any randomness. This means that for a given initial configuration of cell states, the evolution of the system is completely predictable and reproducible.
- In probabilistic cellular automata (PCA), the rules for state transitions include elements of randomness. This means that the evolution of the system can vary even if the initial configuration is the same.

And the evolution of cells can occur in two different ways:

- In synchronous updating, all cells evolve simultaneously at each time step.
- in asynchronous updating, cells evolve sequentially rather than simultaneously.

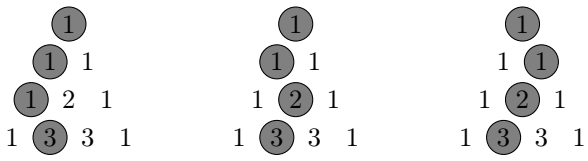
1.3 Pascal's Triangle

The Pascal's Triangle is a fascinating mathematical structure named after the French mathematician Blaise Pascal [21], although it was known to many mathematicians before him. It finds application across various fields due to its remarkable properties and connections to different areas of mathematics, ranging from combinatorics to algebra, from probability theory to geometry and extending into computer science and physics [**?**, 22].

It's a triangular array of numbers where each number is the sum of the two numbers directly above it. The triangle starts with a single 1 at the top, and each row thereafter is constructed by adding adjacent numbers in the previous row. Here's an example of the first few rows:

$$\begin{array}{c} 1 \\ 1\quad 1 \\ 1\quad 2\quad 1 \\ 1\quad 3\quad 3\quad 1 \\ 1\quad 4\quad 6\quad 4\quad 1 \end{array} \qquad \begin{array}{c} \binom{0}{0} \\ \binom{1}{0}\ \binom{1}{1} \\ \binom{2}{0}\ \binom{2}{1}\ \binom{2}{2} \\ \binom{3}{0}\ \binom{3}{1}\ \binom{3}{2}\ \binom{3}{3} \\ \binom{4}{0}\ \binom{4}{1}\ \binom{4}{2}\ \binom{4}{3}\ \binom{4}{4} \end{array}$$

Binomial Coefficients and Lattice Paths Counting Each element of Pascal's Triangle, denoted as $\binom{n}{k}$[1], represents the number of ways to choose k elements from a set of n elements. This is directly related to counting paths in a lattice grid. If you interpret the rows and columns of Pascal's Triangle as coordinates, you can see that moving from the top of Pascal's Triangle (0, 0) to any position (n, k) involves a series of steps right (increasing k) and down (increasing n). This mirrors moving in a lattice grid where you can only move right or up. Here's an example of the three different paths for $n = 3$ and $k = 1$, and respective lattice path.



n \ p	1	2	3
0	0	0	0
1	0	0	1
2	0	1	1
3	1	1	1

Finally, the sum of the elements in a row of Pascal's Triangle corresponds to the total number of lattice paths from the top of the triangle to that row. This sum is equal to 2^n for the n-th row, which encompasses all possible combinations of steps in the lattice grid.

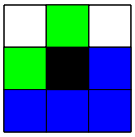
[1] $\frac{(n)!}{k!\times(n-k)!}$

2 Proposed Methods

In its simplest form, I showcase two new dithering methods leveraging cellular automaton principles and Pascal's triangle, which are:

- Pascal's rows as Probabilistic rules
- Pascal's paths as Deterministic rules

In contrast to error diffusion, which typically reads one pixel ■ and updates (writes) the values of the next neighbour pixels ■, these two new methods read the values of the preceding neighbour pixels ■ and update (write) the value of the current pixel ■ accordingly.



Both methods involve generating a discrete value based on the preceding neighbour pixels' values, which is then added to the current pixel value before thresholding. The resulting quantization error is updated for the current pixel, and the process repeats iteratively and asynchronously, following a serpentine or zigzag scan pattern on the raster image. However, they diverge in the way the discrete value is generated.

- In the *probabilistic* version, we sample a discrete value from the rows of the Pascal cumulative distribution function (CDF), where the preceding pixels' values index the corresponding row to be sampled.
- In the *deterministic* version, for each contone value, we attribute a specific Pascal lattice path where the discrete value is indexed in the corresponding path by the preceding pixels' values.

Whereas the probabilistic version does not provide direct control over the points distribution pattern, the deterministic version allows selecting a specific lattice path for each contone value to minimize visual artefacts and enhance the perceived quality of the dithered image. This approach is similar to tone-dependent error diffusion techniques [9], where the diffusion weights are adapted based on the local tone of the image to produce the *blue-noise* dither pattern.

For each method, a visual assessment, the processing time, and the 2D power spectral density (PSD) estimation, along with its radially averaged power spectral density (RAPSD), are provided and compared with:

- Ulichney [3] - [FS with 50% random weights (4), fixed, non-deterministic]
- Ostromoukhov [9] - [3 diffusion weights, tone-dependent, deterministic]

2.1 Pascal's rows as Probabilistic rules

In this section, I present the probabilistic version for two preceding neighbour pixels' values ■ or Left/Right - Top

Pascal's cumulative distribution function Given an 8-bit continuous-tone grayscale image, where each pixel's value ranges from 0 (representing black) to 255 (representing white), we generate Pascal's Triangle up to 256 rows, corresponding to the available levels of grayscale intensity.

Then, the binomial coefficients $\binom{n}{k}$ for each row of Pascal's Triangle are summed and normalized [2] to create a cumulative distribution function (CDF). This function allows for the sampling of new discrete values between 0 and 255 by leveraging the fact that the n-th row of Pascal's Triangle (starting from 0) has n+1 entries. Below, the Pascal's triangle cumulative distribution function generation (left-aligned form), up to the 5-th row.

Pascal's Triangle Coefficients	Cumulative sum of Coefficients	Normalized Cumulative sum (CDF)
Row 0 : 1	1	1
Row 1 : 1 1	1 2	$\frac{1}{2}$ 1
Row 2 : 1 2 1	1 3 4	$\frac{1}{4}$ $\frac{3}{4}$ 1
Row 3 : 1 3 3 1	1 4 7 8	$\frac{1}{8}$ $\frac{4}{8}$ $\frac{7}{8}$ 1
Row 4 : 1 4 6 4 1	1 5 11 15 16	$\frac{1}{16}$ $\frac{5}{16}$ $\frac{11}{16}$ $\frac{15}{16}$ 1

Sampling Pascal's CDF Given the values of the two preceding neighbour pixels ■, A and B, and the value of the current pixel ■, C, the discrete sampling process follows these steps:

1. Calculate the absolute difference, Diff = $|A - B|$
2. Retrieve the corresponding row from Pascal's CDF, PCDF(Diff + 1)
3. Generate a random value from the uniform distribution, Rand = $U(0, 1)$
4. Find the first PCDF(Diff + 1) value that is greater than or equal to Rand.
5. Retrieve the discrete Index of this PCDF value
6. Calculate the new value for C, New ■ = Index + min(A,B)

$$\text{In summary: New} = \begin{cases} A & \text{if } A = B \\ [A, B] & \text{otherwise} \end{cases}$$

Dithering with Pascal's probabilistic rules Finally, the current pixel value ■ plus the New value ■ is compared to a threshold (e.g. 255), the result of the comparison assigns an output value, e.g. 0 or 1 to the halftone image and the current pixel value, C, is updated with the resulting quantization error.

[2] The sum of the coefficients in the n-th row of Pascal's Triangle is 2^n.

Algorithm 1 Dithering with Pascal Probabilistic CA

```
procedure DitherImage(input_image)
    Initialize an empty dithered image
    Initialize PascalCDF
    for each pixel (x, y) in the input image do                    ▷ Serpentine Scan
        pixel_A ← input_image(x, y − 1)
        pixel_B ← input_image(x − dir, y)
        pixel_C ← input_image(x, y)
        discrete_Value ← SamplePascalCDF(pixel_A, pixel_B, PascalCDF)
        new_pixel_C ← pixel_C + discrete_value
        dithered_image(x, y) ← new_pixel_C > Threshold             ▷ Thresholding
        quant_error ← new_pixel_C − 255
        input_image(x, y) = quant_error                            ▷ Update current pixel value
    end for
    return dithered image
end procedure
procedure SamplePascalCDF(A, B, PCDF)
    diff_AB ← abs(A - B)
    Rand ← Random number between 0 and 1                           ▷ Uniform distribution
    Index ← PCDF(diff_AB, Rand)
    return Index + min(A,B)
end procedure
```

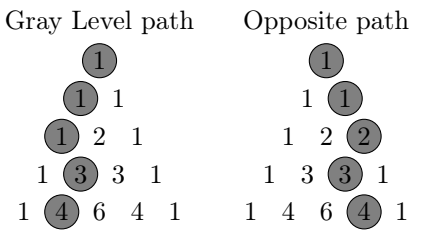
2.2 Pascal's paths as Deterministic rules

In this section, I present the deterministic version to halftone an 8-bit continuous-tone grayscale image based on two preceding neighbour pixels' values ■ or Left/Right - Top. To select, for each contone value, the Pascal path that best mimics a *blue-noise* dither pattern, we explore and analyze all the paths given by Pascal's triangle up to N rows, where N represents the maximum contone value allowed. From the radially averaged power spectrum density (RAPSD), we identify the path for which the spectral energy is concentrated above the principal frequency B_f, which is defined as a function of the input contone value C as:

$$B_f(C) = \begin{cases} \sqrt{C} & \text{for } 0 < C \leq \frac{1}{2} \\ \sqrt{1-C} & \text{for } \frac{1}{2} < C \leq 1 \end{cases}$$

However, an 8-bit continuous-tone grayscale image represent $2^{255} \simeq 5.8 \times 10^{76}$ paths to explore, which is infeasible due to the immense computational complexity and time required, even with GPU acceleration. To overcome this difficulty, from the pair factors of 255 [1,3,5,15,17,51,85,255], we select [15, 17] and generate all the possible paths for Pascal's triangle up to row 15 and 17. This results in $2^{15} = 32,768$ and $2^{17} = 131,072$ paths, respectively.

We select the paths that present a *blue-noise* dither pattern and combine them to generate the lattice paths of up to 256 entries for each gray level from 0 to 255. It is not necessary to find the respective lattice path for each gray level from 0 to N among the 2^N possible lattice paths in Pascal's triangle up to row N. Indeed, from the lattice paths of gray levels from 0 to $\lfloor \frac{N}{2} \rfloor$, we can generate the corresponding lattice paths for the gray levels from $\lfloor \frac{N}{2} \rfloor + 1$ to N by taking advantage of Pascal's lattice path symmetry.



n \ p	C	C'
0	0	0
1	0	1
2	0	2
3	1	2
4	1	3

Algorithm 2 Generate Pascal Triangle Paths

```
function GENERATE_PASCAL_PATHS(N)
    Input: N (number of rows)
    Output: paths (array of paths with indices)
    num_paths ← 2^N                                   ▷ Total number of paths
    paths ← cell array of size num_paths
    for i ← 0 to num_paths − 1 do
        binary_str ← dec2bin(i, N)                    ▷ Convert number to binary string
        row ← 0
        col ← 0
        path_indices ← [(row, col)]                   ▷ Initialize with the top node indices
        for j ← 1 to N do
            row ← row + 1
            if binary_str[j] =′ 0′ then
                col ← col                             ▷ Column stays the same for left child
            else
                col ← col + 1                         ▷ Column increments for right child
            end if
            path_indices ← append (row, col)
        end for
        paths[i + 1] ← path_indices                   ▷ Store the path with indices
    end for
end function
```

Algorithm 3 Dithering with Pascal Deterministic CA

```
procedure DITHERIMAGE(input_image, generated_paths)
    Initialize an empty dithered image
    for each pixel (x, y) in the input image do                    ▷ Serpentine Scan
        pixel_A ← input_image(x, y − 1)
        pixel_B ← input_image(x − dir, y)
        pixel_C ← input_image(x, y)
        discrete_value ← DiscreteValueFromPaths(pixel_A, pixel_B, pixel_C, generated_paths)
        new_pixel_C ← pixel_C + discrete_value
        dithered_image(x, y) ← new_pixel_C > Threshold             ▷ Thresholding
        quant_error ← new_pixel_C − 255
        input_image(x, y) = quant_error                            ▷ Update current pixel value
    end for
    return dithered image
end procedure
procedure DISCRETEVALUEPATH(A, B, C, PATHS)
    diff_AB ← abs(A - B)
    Value ← PATHS(diff_AB, C)
    return Value + min(A,B)
end procedure
```

3 Results

3.1 Pascal's rows as Probabilistic rules

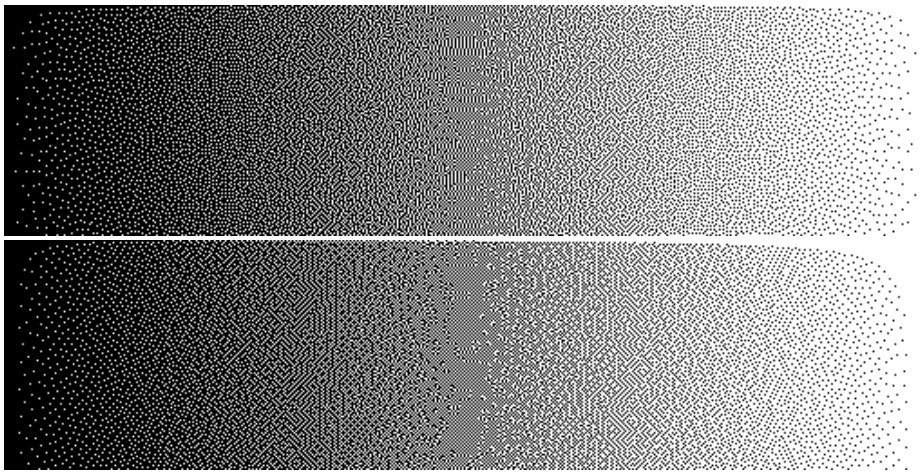


Fig. 3: Dithered gray ramp: (Top) Ulichney - FS with 50% random weights (4 neighbour pixels - non-deterministic); (Bottom) Our method (2 neighbour pixels - non-deterministic)

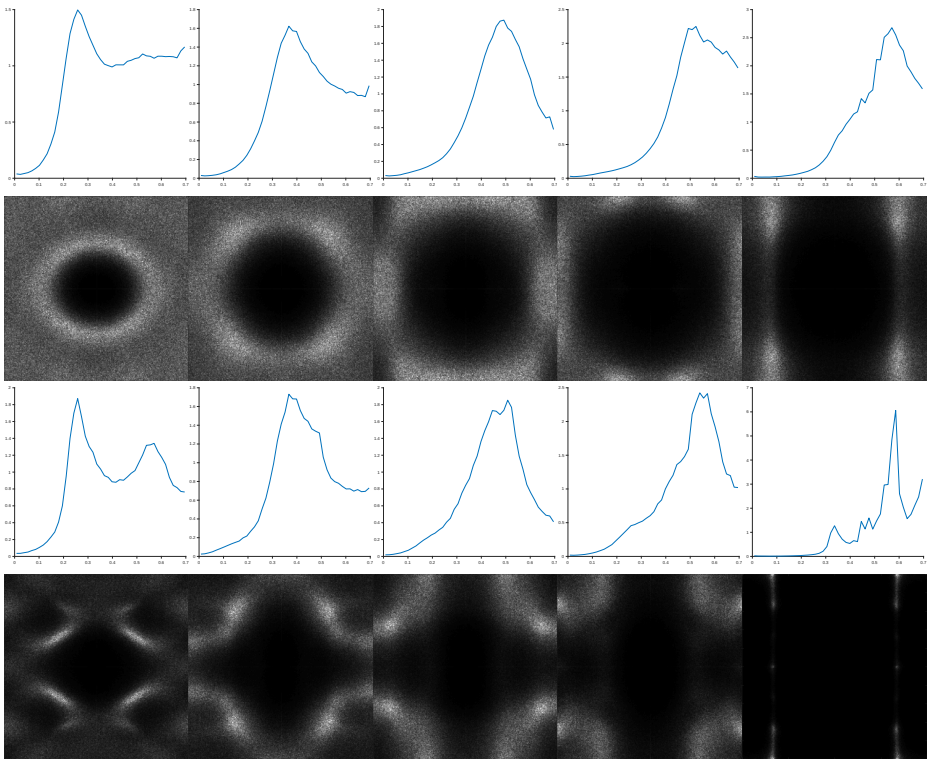


Fig. 4: RAPSD and PSD for gray levels [17, 34, 51, 68, 85]: (Top) Ulichney - FS with 50% random weights (4 neighbour pixels - non-deterministic); (Bottom) Our method (2 neighbour pixels - non-deterministic)

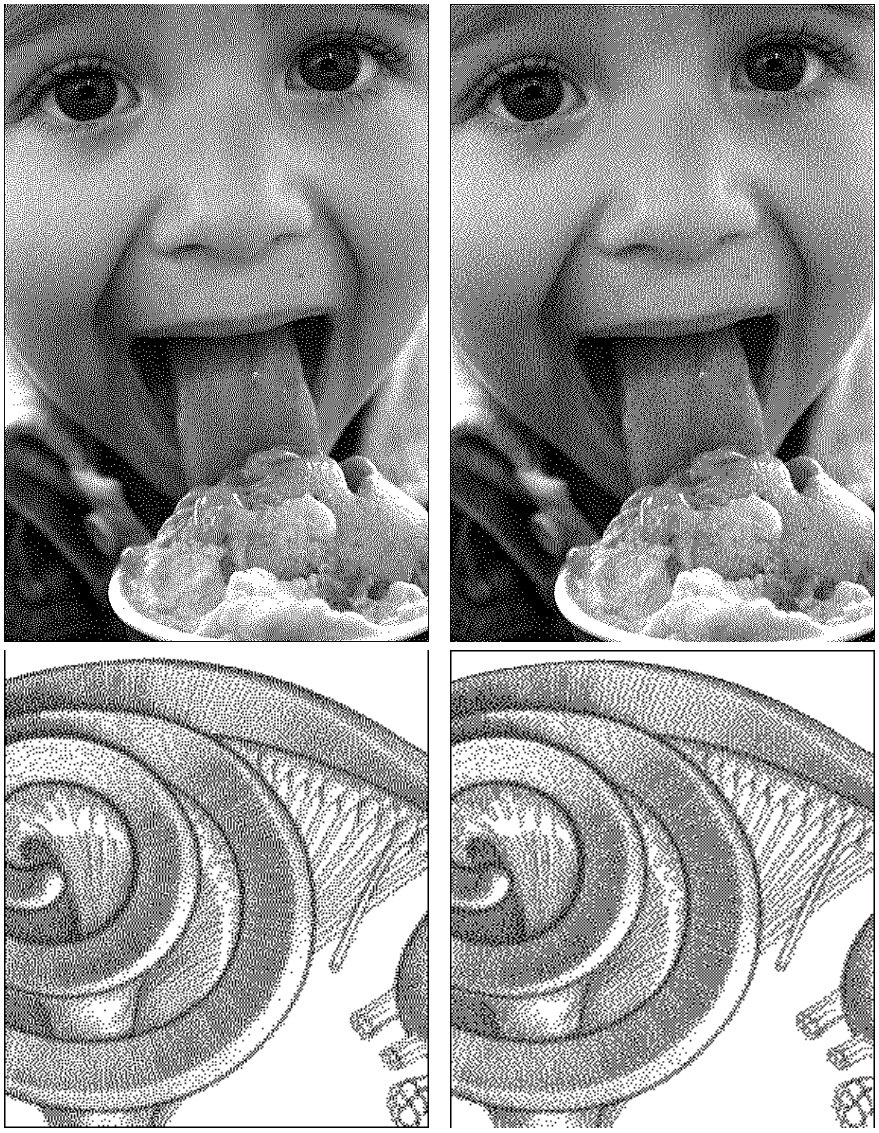


Fig. 5: (Left) Ulichney - FS with 50% random weights (4 neighbour pixels - non-deterministic); (Right) Our method (2 neighbour pixels - non-deterministic)

Remarks : With two preceding neighbour pixels, our probabilistic method presents higher intensities along the diagonals, indicating directional frequency distribution. Dithered images produced by this method have more structured and anisotropic textures. In contrast, Ulichney method presents a circular symmetry with a bright ring around the center, pointing to an isotropic frequency distribution where frequencies are uniformly spread in all directions.

3.2 Pascal's paths as Deterministic rules

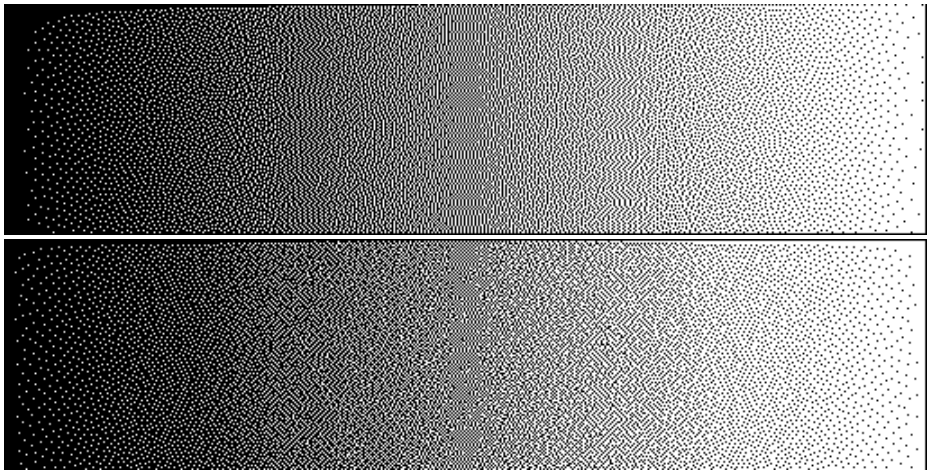


Fig. 6: Dithered gray ramp: (Top) Ostromoukhov - Tone-dependant weights (3 neighbour pixels - deterministic); (Bottom) Our Tone-dependent method (2 neighbour pixels - deterministic)

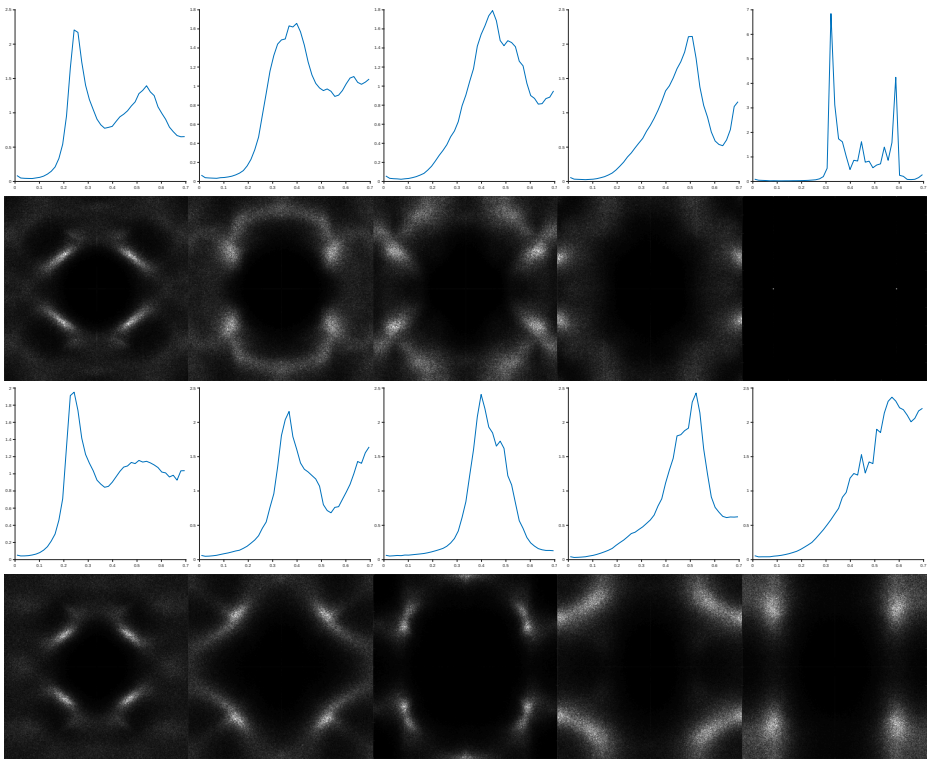


Fig. 7: RAPSD and PSD for gray levels [17, 34, 51, 68, 85]: (Top) Ostromoukhov - Tone-dependant weights (3 neighbour pixels - deterministic); (Bottom) Our Tone-dependent method (2 neighbour pixels - deterministic)

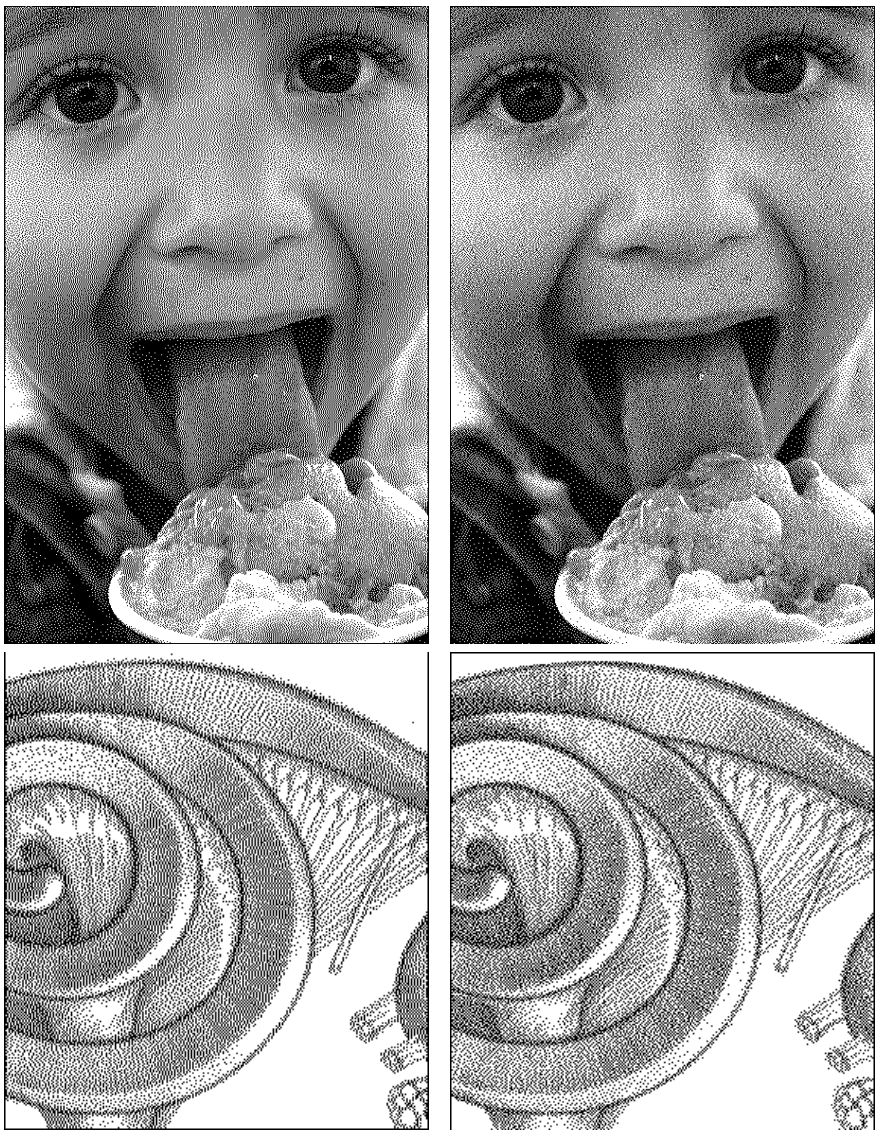


Fig. 8: (Left) Ostromoukhov - Tone-dependant weights (3 neighbour pixels - deterministic); (Right) Our Tone-dependent method (2 neighbour pixels - deterministic)

Remarks : Besides points sampling optimization, our deterministic method still presents diagonal, structured, and anisotropic textures, which likely originate from the use of only two neighboring pixels. However, the points distribution and density in the highlights and shadows have improved, as well as the processing time, since we no longer have to sample discrete values from a probability distribution. It is now comparable with the methods of Ulichney and Ostromoukhov, as well as standard error-diffusion techniques.

4 Conclusions and Future Work

In this article, I presented two new methods for generating high-quality dithered images based on the principles of cellular automata and Pascal's Triangle as probabilistic and tone-dependency deterministic rules. Dithering and halftoning are well-established techniques, and these new methods offer valuable insights into potential enhancements in this field, as well as reverse engineering past techniques into new innovations. In their simplest form, these methods address challenges in simulating detailed textures and gradients for image processing. This new framework and its discrete singularity suggest future improvements, such as extending the neighbour scheme from multinomial coefficients for color and multilevel image processing. I've called it the "*The Blue Tip of the Iceberg*" as this article only presents a small and visible portion of a much larger and complex framework. The majority of its mass lies beneath the surface of the water, hidden from view.

The Python implementation can be found at https://github.com/GovYvanR/Dithering-with-Pascal-Triangle, this version already implement the possibility of modulating the threshold.

Acknowledgments. The author would like to express gratitude to numerous individuals who were involved at different degree in the present paper, particularly those who continue to pursue their dreams.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

References

1. Floyd, R.W., Steinberg, L.: An Adaptive Algorithm for Spatial Greyscale. Proceedings of the Society for Information Display **17**(2), 75–77 (1976)
2. Jarvis, J., Judice, C., Ninke, W.: A survey of techniques for the display of continuous tone pictures on bilevel displays. CGIP **5**(1), 13–40 (March 1976)
3. Ulichney, R.: Digital halftoning. MIT Press, Cambridge, MA, USA (1987)
4. Eschbach, R., Knox, K.T.: Error-diffusion algorithm with edge enhancement. J. Opt. Soc. Am. A **8**(12), 1844–1850 (Dec 1991)
5. Velho, L., Gomes, J.d.M.: Digital halftoning with space filling curves. In: Proceedings of the 18th annual conference on Computer graphics and interactive techniques. pp. 81–90. SIGGRAPH '91, ACM, New York, NY, USA (1991)
6. Eschbach, R.: Reduction of artifacts in error diffusion by means of input-dependent weights. Journal of Electronic Imaging **2**(4), 352–358 (1993)
7. Shiau, J.N., Fan, Z.: Set of easily implementable coefficients in error diffusion with reduced worm artifacts. In: Electronic imaging (1996)
8. Knox, K.T.: Evolution of error diffusion pp. 448–458 (1998)
9. Ostromoukhov, V.: A simple and efficient error-diffusion algorithm. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. pp. 567–572. SIGGRAPH '01, ACM, New York, NY, USA (2001)

10. Li, P., Allebach, J.P.: Tone-dependent error diffusion. IEEE Transactions on Image Processing **13**(2), 201–215 (2004)
11. Levien, R.: Output dependant feedback in error diffusion halftoning. IS&T Imaging Science and Technology **1**, 115?118 (1992)
12. Lau, D.L., Arce, G.R., Gallagher, N.C.: Green noise digital halftoning. Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269) **2**, 39–43 vol.2 (1998)
13. Cook, R.L.: Stochastic sampling in computer graphics. ACM Trans. Graph. **5**, 51–72 (January 1986)
14. Mitchell, D.P.: Generating antialiased images at low sampling densities. In: Proceedings of the 14th annual conference on Computer graphics and interactive techniques. pp. 65–72. SIGGRAPH '87, ACM, New York, NY, USA (1987)
15. Yellott, J.I.: Spectral consequences of photoreceptor sampling in the rhesus retina. Science **221**(4608), 382–5 (1983)
16. Dubra, A., Sulai, Y., Norris, J.L., Cooper, R.F., Dubis, A.M., Williams, D.R., Carroll, J.: Noninvasive imaging of the human rod photoreceptor mosaic using a confocal adaptive optics scanning ophthalmoscope. Biomed. Opt. Express **2**(7), 1864–1876 (Jul 2011)
17. Lagae, A., Dutré, P.: A comparison of methods for generating Poisson disk distributions. Computer Graphics Forum **27**(1), 114–129 (March 2008)
18. von Neumann, J.: Theory of Self-Reproducing Automata. University of Illionois Press, Champain, IL (1966)
19. Games, M.: The fantastic combinations of john conway's new solitaire game "life" by martin gardner. Scientific American **223**, 120–123 (1970)
20. Wolfram, S.: A New Kind of Science. Wolfram Media (2002)
21. Pascal, B.: Traité du Triangle arithmétique, avec quelques autres petits traités sur la même matière, Par Monsieur Pascal. Guillaume Desprez (1665)
22. Farina, A., Giompapa, S., Graziano, A., Liburdi, A., Ravanelli, M., Zirilli, F.: Tartaglia-pascal's triangle: a historical perspective with applications. Signal, Image and Video Processing **7**(1), 173–188 (2013)