Reprinted from
Journal of Imaging Science and Technology® 60(6): 060403-1–060403-9, 2016.
https://doi.org/10.2352/ISSN.2470-1173.2017.18.COLOR-043
2599908

On Large Local Error Accumulation in Multilevel Error Diffusion

Reiner Eschbach▲
The Norwegian Colour and Visual Computing Laboratory, Faculty of Computer Science and Media Technology,
NTNU—Norwegian University of Science and Technology, Gjøvik, Norway
Monroe Community College, Rochester, NY
E-mail: Reiner.Eschbach@yandex.com

Marius Pedersen▲
The Norwegian Colour and Visual Computing Laboratory, Faculty of Computer Science and Media Technology,
NTNU—Norwegian University of Science and Technology, Gjøvik, Norway

Abstract. *Error diffusion is an often used method that transforms a continuous tone (multibit) image into an image of lower bit depth, most commonly into a binary output of black and white. The simplicity of the processing and the quality of the output have made error diffusion a frequently used tool. Part of the image quality is attributed to the minimization of quantization errors in the error-diffusion process. This article describes local instabilities in a color multilevel error-diffusion system that—in contrast—can lead to large local errors in the output, far exceeding the normally expected quantization errors. This can have serious negative effects specifically in connection with the design and incorporation of color calibration sheets. © 2016 Society for Imaging Science and Technology.*

INTRODUCTION

Error diffusion is a feedback-based halftoning technique that first was described for image data by Hale[1]—for a one-dimensional signal—and Floyd and Steinberg[2]—for a two-dimensional image in 1976. In a simple description, the input value at location (m, n) is quantized to its "nearest" possible output level. The error between the input at (m, n) and output at (m, n) is then compared and defined as an error. This error is subtracted from future pixels and this combination of actual input value and accumulated error value is considered the actual input for quantization. This can be written as:

$$o(m, n) = Q\left[i(m, n) - \sum_{i,j} a_{j,k} e(m-j, n-k)\right] \tag{1}$$

and

$$e(m, n) = o(m, n) - \left[i(m, n) + \sum_{i,j} a_{j,k} e(m-j, n-k)\right] \tag{2}$$

where $o(m, n)$ refers to the output, $Q(\ldots)$ is the quantization operator, $i(m, n)$ is the original input and a_j, k are the weights with which the error is distributed to yet unprocessed pixels. Note that the error might be defined as input minus output, or output minus input; however, this only changes signs in (Eqs. (1) and (2)).

Note that at this point we are not assuming anything about $i(m, n)$, in terms of vector or scalar quantity, $o(m, n)$ in terms of number N and distribution of available output levels, and $a_{j,k}$ in terms of the actual error distribution.

In the past, a lot of work has been done on the selection of the weighting coefficients $a_{j,k}$, for example using the larger matrix of Jarvis et al.,[3] or by simplifying the weights as for example done by Fan et al.,[4] or by optimizing the weights according to some criteria as in Kim and Gill[5] and Kolpatzik and Bouman[6] switching between different weight matrixes as in Eschbach[7] and adapting the weights to other requirements, e.g., complex-valued for holography[8] or based on hardware constraints (found mainly in the patent literature.[9,10])

An additional area of work was the quantizer Q, which in the binary case is a simple step function with an optionally spatially varying threshold or distance metric[11] or an input dependent threshold.[12] Generalizing the quantizer to an N-level system,[13] including an 8-bit to 8-bit conversion,[14] or to a vector quantizer,[15] e.g., for color, was also done early.

In (virtually) all cases, the modifications attempted to optimize the output with respect to some subjective or objective criteria, such as image perception, quantization noise spectrum, etc. In most cases, the optimization of error-diffusion weights $a_{j,k}$ was done under the additional constraint

$$\sum_{j,k} a_{j,k} = 1 \tag{3}$$

essentially guaranteeing that 100% of the error is compensated. The criterion of Eq. (3), is not a sufficient criterion and might lead to numerical instabilities[16] that grow exponentially. In the stable situations, Error Diffusion

▲ IS&T Members.

Received June 23, 2016; accepted for publication Oct. 3, 2016; published online Nov. 14, 2016. Associate Editor: Chunghui Kuo.

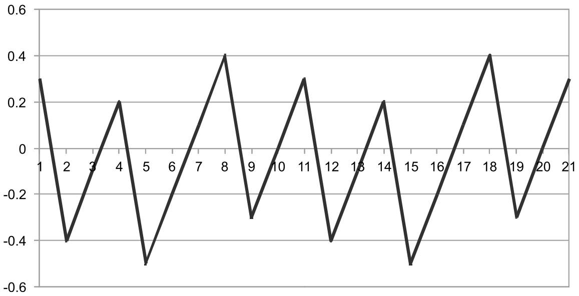


Figure 1. First 20 error values for a constant input of 0.3 and the two output levels [0, 1].

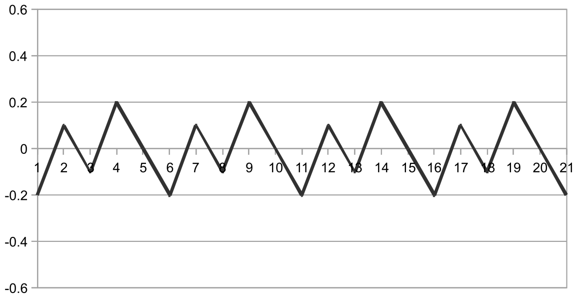


Figure 2. First 20 error values for a constant input of 0.3 and three output levels [0, 0.5, 1].

as described in Eqs. (1)–(3) normally results in local errors that are less than the dynamic range for an individual pixel.

In this article, we will describe a situation that creates a large accumulation of process errors that does result in a large localized drift between the desired and the created value that goes beyond the bounds that are commonly associated with the quantization. However, despite being large with respect to the difference between quantization states, it is not growing exponentially to lead to numerical instabilities. Theoretically, the described instability can lead to an infinite error; however, this value will only be approached in an infinite number of steps, as will be described in subsequent sections.

MULTILEVEL ERROR DIFFUSION

As a general observation one can say that—for scalar error diffusion—an increase in the number of output levels $Q\{\hat{\imath}\}$ will directly improve the quality of the output image $o(m, n)$. Transiting from the original binary method $N = 2$ to other small number of states, e.g., $N = 3$, $N = 4$, causes some texture problems, commonly for input values close to an intermediate output level,[13] but reduces the locally accumulated error. For vector systems a similar effect is also often observed.[17] The improvement in output quality can be understood by considering that the quantization error is the difference between the desired value and the closest (to a specific criterion) possible output value and thus, with more possible output values the quantization error in general decreases. This is illustrated for a simple case of 1D error diffusion[1] with an input value of 0.3 and the two output levels $Q\{i\} = [0, 1]$ in Figure 1. Here, the first 20 errors are shown.

Adding a third output level at 0.5 to $Q\{i\} = [0, 0.5, 1]$ changes the errors to the ones shown in Figure 2, which is a clear reduction compared to the errors shown in Fig. 1.

Note that for Figs. 1 and 2 only one weight $a_{1,0} = 1$ was used, which is identical to the method described in by Hale.[1]

For scalar signals, the reduction in error through the addition of additional output states is intuitive. If the quantization error one has to make at a certain location is smaller than in the case of fewer output levels, then the accumulated error should also be smaller. This situation, however, can change if we are transitioning from a scalar to a vector system.

VECTOR ERROR DIFFUSION

Error Diffusion has early on been generalized to Vector Error Diffusion (VED).[15] One of the promises of VED is the joint control of the output dot placement, eliminating the cross-separation issue from the scalar approach. Another promise is the ability to better utilize a system where the different available output states are note separable. Consequently, there is considerable amount of work done in VED.[18–21] In this article, we will concentrate purely on the error accumulation for some of the choices of output states. At its core, any vector error diffusion is a multilevel system, since it takes at least three output states to define a possible area or volume. Let us consider a 2-dimensional output space first, generalizing to the 3-dimensional RGB space in later sections.

It can be seen that in the 2D case we can partition the plane into triangles defined by the three closet output states. In the actual 3D case, this would result in a partition into tetrahedra, but we will restrict ourselves to the 2D case and triangles. The 2D color space is tiled by triangles, where the shape of the triangle is defined by the available output states. This physical tiling, rather than an idealized mathematical tiling, result in triangles that are a direct function of the physical output states. For our case, we will consider the effect of different triangle properties on the error accumulation.

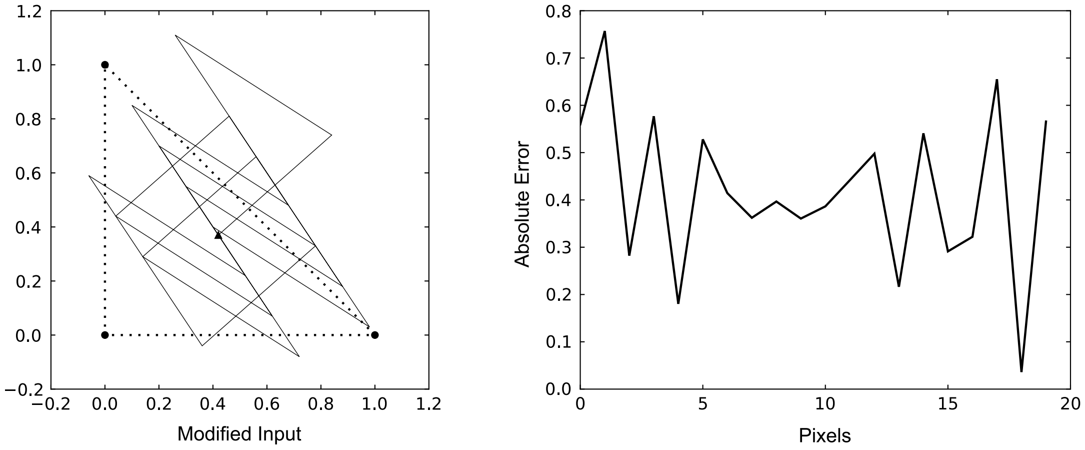
In the correspondence to the scalar case, we can define a 2D space as having the output states [(0, 0), (1, 0), (0, 1)] as a simple idealized form and examine the effect changes have on this structure. In order to visualize the error, we can now use two different descriptions. The first is the Modified Input and the second is the absolute error, meaning the length of the error vector in that space.

The modified input is a commonly used metric, since it shows the value that is actually given to the quantizer. In rewriting Eq. (2) we can define the Modified Input i_{mod} as

$$i_{mod}(m, n) = i(m, n) - \sum_{i,j} a_{j,k} e(m - j, n - k) \qquad (4)$$

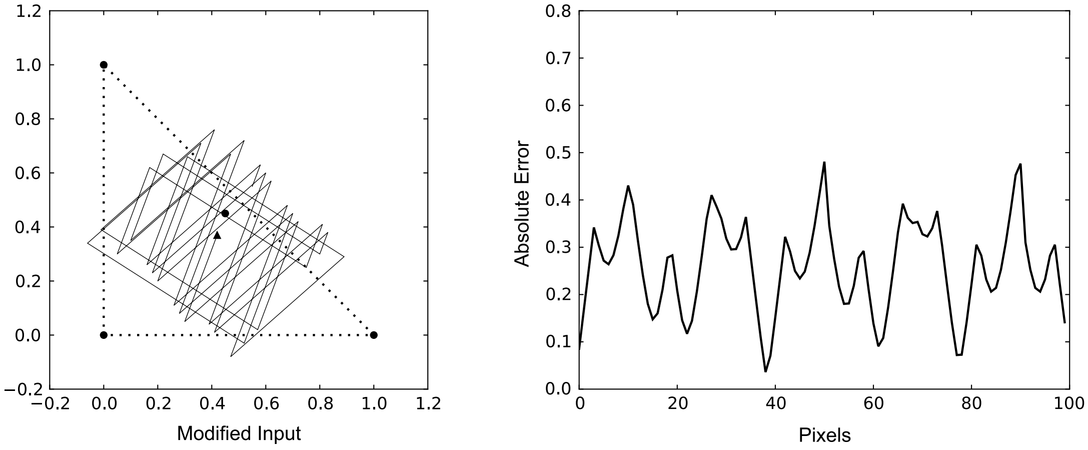
where the Modified Input shows the relation to the original data and the dynamic range.

Both error metrics are shown for an input value of (0.42, 0.37) as example in Figure 3. Fig. 3(a) shows the progression of the Modified Input for the first 20 pixels. The three output stats that define the gamut are indicted by circles and their



(a) (b)

Figure 3. (a) Progression of the Modified Input for a given constant input value (triangle) of (0.42, 0.42) and three output states (circles) [(0, 0), (0, 1), (1, 0)]. (b) Progression of the absolute error in the same scenario.



(a) (b)

Figure 4. Adding an additional output state at (0.45, 0.45), additional circle in (a) reduces the Modified Input and the absolute error in (b).

convex hull is indicated by the dotted line. The solid line is the progression of the Modified Input. One can see that the Modified Input stays close to the gamut. It extends beyond the boundaries to some degree, since the Modified Input adds an error term to the actual input value. Since the actual input value is limited by the gamut, we would expect to add—at most—a value of $O\{\text{dynamic range}\}$.

Fig. 3(b) shows the progression of the absolute error for the first 20 pixels. The absolute error stays within one dynamic range—as was expected.

If one now adds a fourth possible output state, as is done in Figure 4, we effectively should reduce the error and thus create a Modified Input that is closer to our available dynamic range and an absolute error that decreased from before. In our case, we are introducing a level (0.45, 0.45) [indicated as the fourth circle] in Fig. 4(a) and as predicted, the effect is similar to the effect observed in the scalar black and white case, in that the error decreases. Note that we increased the number of pixels to 100, since some effects are only visible at larger pixel counts.

In the case shown in Fig. 4, the maximum error was reduced from approximately 0.8 to 0.5 as one might expect when comparing it to the scalar case. From the data above, one might assume that the increase in number of output levels will always create a reduction in the error. However, we will show that contrary to this assumption, slight changes in the scenario can actually increase the error considerably.

Please be aware that all input values that we will be using here and in the subsequent sections are within the dynamic range, meaning within the convex hull of the defined output states. Thus every value can be well represented by the output states and no issue exists caused by a violation of the available dynamic range. We emphasize this point, since violations of the dynamic range have an effect that can be confused when only a few pixels are regarded.

To illustrate the case of increased errors, let us consider the same distribution of states as before, but using a slightly different input value of (0.51, 0.485) which is still within the convex hull of [(0, 0), (1, 0), (0, 1)], but moves the input value to the triangle spanned by the output states [(0.45, 0.45),

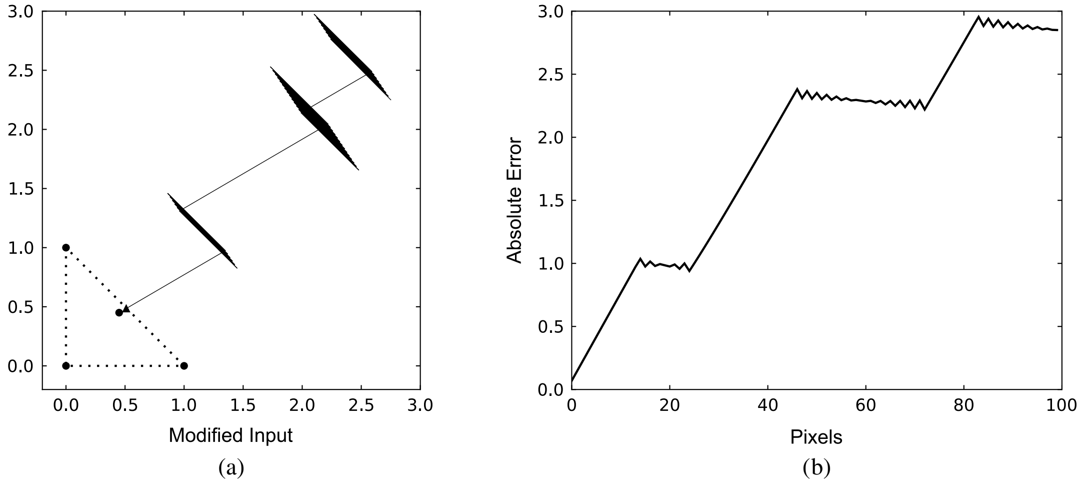


Figure 5. Changing the input value slightly [black triangle in (a)] leads to a large Modified Input in (a) and the corresponding large absolute error in (b).

(1, 0), (0, 1)]. This is shown in Figure 5, where the scales of the x- and y-axes have changed with respect to Fig. 5.

The Modified Input can now be on the order of (2.5, 2.5) despite the dynamic range of the input being bound by the triangle [(0, 0), (1, 0), (0, 1)]. Considering the four possible output states (circles in the Fig. 5) spanning the dynamic range, one clearly sees that the Modified Input is now deviating considerably.[22] This deviation can also been seen when looking at the absolute error as done in Fig. 5(b), where we see an increasing in absolute error by essentially a factor of 6.

Since the addition of an output state causes, in this case, a problem, one might say that it could be circumvented by a more intelligent definition of the possible output states. However, output states are often defined by actual physical means, e.g., colorants, separations, in which case problem illustrated above exist.[20,21] Omitting that output state in the computation is also not a desired option since it eliminates the advantages this output state creates for other input levels than the one used in this example.

CONVEX HULL AND CIRCUMCENTERS

In order to get a better understanding of the process causing the described problem, one can visualize the output states as the set of points P that form the corners of triangles as done in Figure 6. For the 2-dimensional case, we now have a set of triangles spanning the entire possible in-gamut space and the definition of the three closest output states to any input state. Choosing the triangles one can also easily generalize the description to a 3D color space with more than the saturated output states and to spectral data for a physical rather than visual match between input and output.

In the example of Fig. 4, the input value was contained inside the lower triangle (labeled A in Fig. 6) and for the case of Fig. 5 was contained in the slim upper triangle (labeled C in Fig. 6).

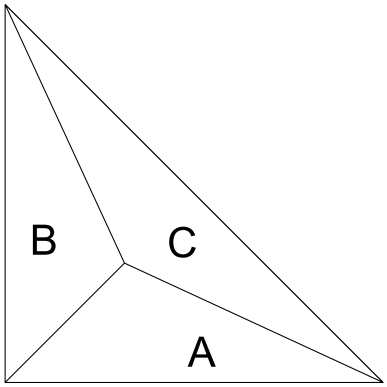


Figure 6. Triangles A–C defining the areas for the given output states.

The shape difference of the triangles (A, B) versus (C) can be used to explain and predict the behavior of the Modified Input and absolute error.

In calculating the "closest" possible output states, we will always pick one of the three states that forms the triangle the current value is in. The boundary for the selection, or decision surface, can be shown as the orthogonal bisectors of the different sides, meeting in the circumcenter of the triangle. This circumcenter is the center of the circle going through the corresponding states, i.e. the circumcircle. This is shown in Figure 7 where the circumcenters of the three triangles are marked using the labels of their respective triangles. Note that the outside triangle spanned by [(0, 0), (1, 0), (0, 1)] is not indicated since it never forms the smallest triangle a value can be in.

As one can see that the circumcenters for all obtuse triangles fall outside the actual triangle and in the case of triangle "C" the furthest outside. The relative circumcenter position is an actual explanation for the error behavior we observed in Fig. 5. When thinking of the bisectors as the "threshold line" between two states, it becomes clear that the modified input value has to approach—and cross—the bisector in order to select another output state. If three states are needed—as in our examples—the modified input should

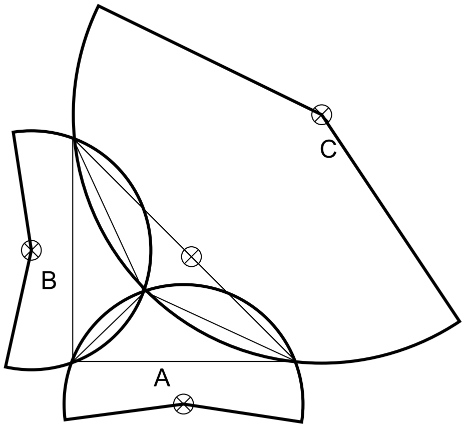


Figure 7. Circumcenter locations for the three triangles spanning the dynamic range of the examples.

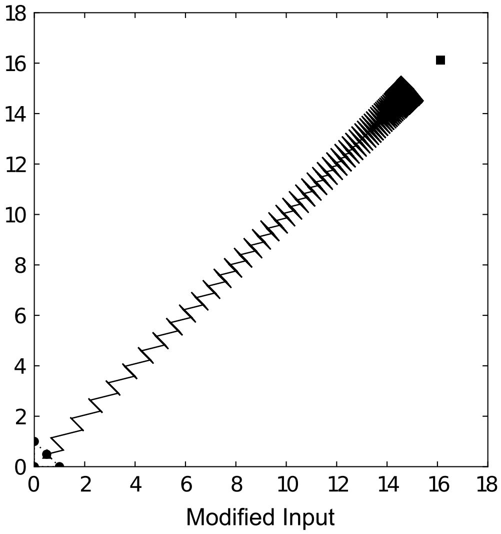


Figure 8. An extreme example of a shallow obtuse triangle having a circumcenter at (17, 17).

approach the circumcenter point before the algorithm can effectively switch between the different states. For the case of Fig. 5 this circumcenter would be at location (2.975, 2.975) which is the value the Modified Input approaches. It should be noted that the Modified Input will fluctuate around the circumcenter in an area that makes all three output states accessible.

Considering the circumcenter, one can also understand that theoretically the error can approach infinity, but that it needs an infinite number of steps to get there. An extreme example is shown in Figure 8 where the circumcenter point is marked by a black square at location (17, 17). Note that the input dynamic range is defined by the same triangle as before (visible four circles) and that the actual error of approximately 15 is thus well outside the actual data range. Fig. 8 also indicates the need to show large pixel counts, since drifts and quasi-stability alternate often, with the overall Modified Error still approaching the circumcenter marked by the square, as already visible in Fig. 6.

At this point one might—correctly—state that there are only few colors inside an extremely obtuse triangle and that in real cases no image will have a large number of constant color values inside this small triangle. This is true for natural scenes but changes if one considers the creation of multilevel or multispectral calibration sheets. Inside each calibration sheet, there might be more than 20,000 identical values (assuming a 600 dpi printer and a patch of ca 5×5 mm^2). In that case, there is a large accumulated error. This error will actually influence subsequent calibration patches since the error diffusion will compensate the entire error and a local accumulation in one direction will have to be compensated by subsequent pixels. If this happens on a calibration sheet, the actual calibration data will contain large errors that subsequently will be embedded into all images run through the system. Thus a strongly obtuse output state configuration in vector error diffusion can lead to actual problems in print even if the images do not contain considerable image content in that area of color space.

ERROR LIMITATION

There are two obvious approaches to reduce the problem in error diffusion. One of the approaches is to reduce the error feedback, either by using weighting coefficients that do not sum to "1" or by limiting the total accumulated error through, e.g., thresholding. A second approach is to eliminate output states from the set of used states as a function on the centroid point for that set of states.

Both methods are practical implementations, but they do not offer new insights into possible ways to modify the error-diffusion method to get a more stable system and to avoid potential other problems that are inherent in a vectorial/multilevel system. Therefore, we are trying to explore alternate methods that still fulfill the full error feedback through $\sum_{j,k} a_{j,k} = 1$ and still use all available output states, albeit with a potentially different selection mechanism. For the purpose of this article, we will concentrate on a purely mathematical determination of "error" either by considering absolute or vector errors, and we will not directly evaluate visual quality. The reason for this is that the visual quality is potentially influenced by other attributes as texture, "worms," etc. and that those artifacts can occur almost independent on the actual local error.

L2 Norm and Homogeneous Distributions

In error diffusion, scalar and vectorial, the error calculation can be seen as an L2 norm. When using this standard norm in vector error diffusion, the selection of the output state will tend to move the modified input toward the circumcenter of the available states. For a vectorial scenario, however, one can envision different approaches. Of these approaches one can argue a preference for approaches that are indistinguishable from an L2 norm for the scalar case, thus including, e.g., L1 norm. However, one can take a different view of thresholding in error diffusion and regard it as a two step process, with the first state being a decision process with a definable decision surface and the second step

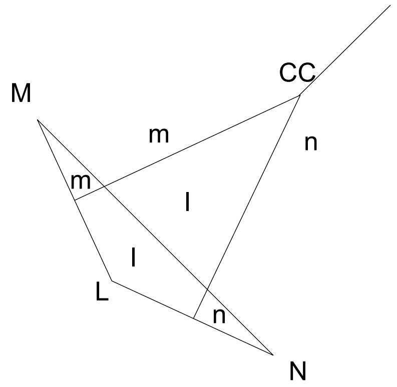


Figure 9. Indicating the likelihood that an output state is selected for an obtuse triangle.

is the calculation and forwarding of the corresponding error. This is not actually a deviation from the original concept of error diffusion. Consider quantization to be a process by which one image is transferred to another image under a specific norm. The standard (e.g., L1, L2) norm would give a direct binarization, and error diffusion actually modifies this decision by incorporating past behavior. There is no principle argument against doing a similar step for vector quantization.

Consider Figure 9 where we show the triangle "C" from Figs. 6 and 7 using the labels L, M, N for the three output states defining that triangle and CC for its circumcenter. The areas marked l, m, n are the areas that—under our norm—would select the corresponding states. For values that are inside our triangle, we can see that the output state L is the most likely selected output state, and also the output state that drives the vector error toward CC. Here, "most likely selected" assumes a homogeneous distribution of input values. We chose this language intentionally, since it directly implies a different way at looking at the thresholding in the scalar case. There, a L2 norm would actually create an equally likely output state for a homogeneous input value. If we now assume that it is the uneven likelihood of an output that is in part the driver and that this likelihood in scalar problems is implemented as a simple distance, we can also explore if the likelihood—or something similar—is also a valid approach for the vectorial case.

Under that assumption, we would change the decision boundaries from being the bisectors and circumcenter, to the medians and the centroid. These decision boundaries are shown in Figure 10. Both decision boundaries from Figs. 9 and 10 will degenerate to an identical decision when transitioning to a scalar model. Given this explanation one can argue that a decision surface that more closely resembles the medians and centroid would be preferable in at least some of the vector cases. Note that in acute triangles the circumcenter is also inside the triangle and thus "close" to the centroid.

DECISION BOUNDARIES AND PHANTOM LEVELS

From the previous examples, we have seen that the areas of color space where the output states form strongly obtuse triangles, cause large error accumulations. In the following, we will concentrate on those triangles only. Thus, of all possible output colors for a VED system, only the—likely small—subset of output colors is considered that fall into this class. However, these colors might be contained in the calibration sheet and thus also influence other areas.

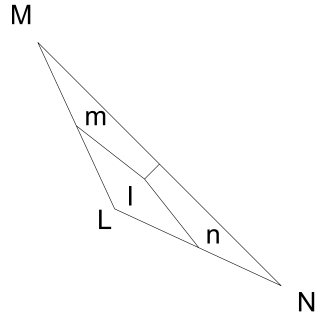
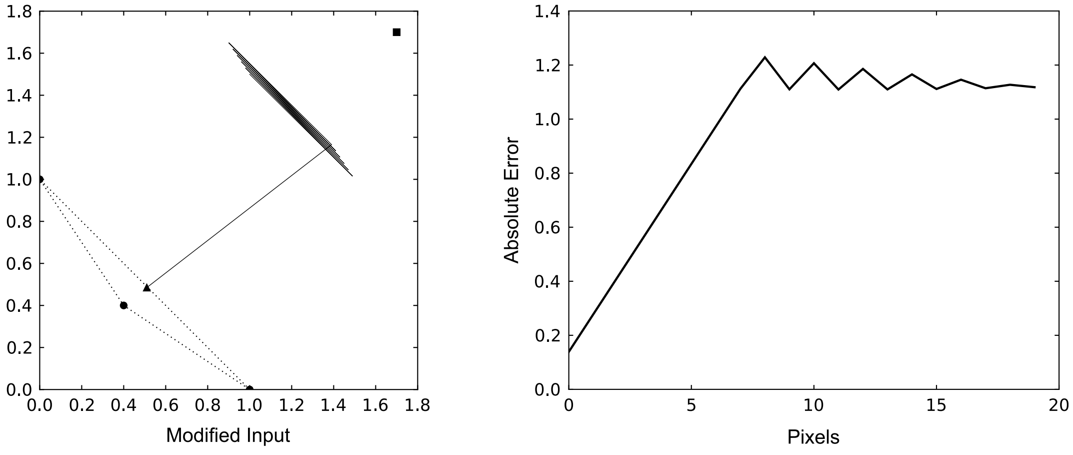


Figure 10. Defining the output state selection through medians and centroid results in an equal likelihood.

In Fig. 10 we argued that decision boundaries are not necessarily an L2 distance. The only requirement that we have for the decision boundaries (or surface in the multidimensional case) is that they maintain full error correction over the image and only locally modify the greedy minimum distance criterion. The "equal likelihood" from the previous paragraph is one such decision boundary; however, it suffers from the fact that the actual processing becomes more complicated and it is also one specific form with no flexibility. For that reason, we are exploring here a different approach that (1) creates more flexibility in the selection likelihood and (2) stays largely within the computational framework of L2 metrics.

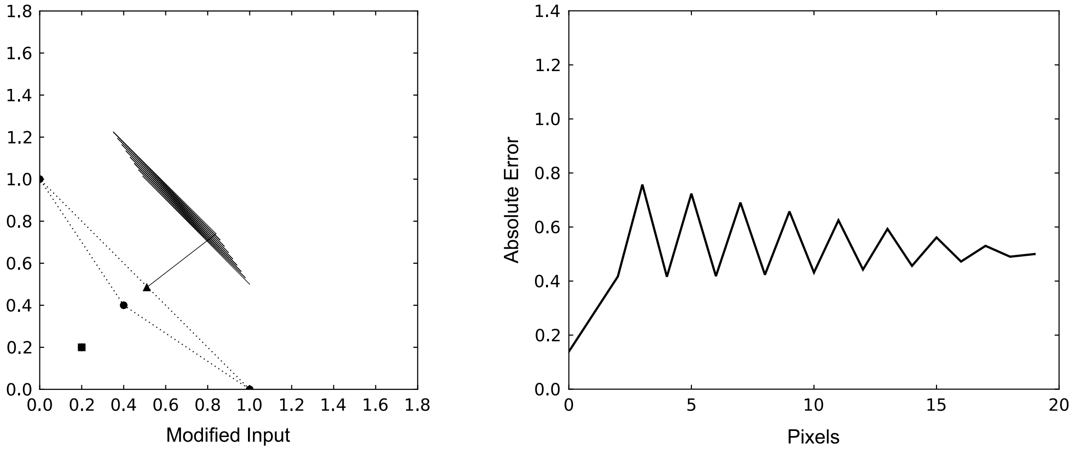
As stated before, in the standard distance metrics, the values tend to approach the circumcenter of the three states defining the area. Changing one of the states will change the circumcenter and thus also change which state will be chosen as output. Let us introduce a Phantom Level, i.e., a mathematical output state that does not have a physical output state associated with it and let us call that state L'. Note that we are not able to actual print that state L' since the Phantom Level is purely mathematical. For simplicity we will also locate L' be on the same vertex as L. This is done in the scope of this article to show the general effect and the flexibility of the Phantom Level concept, and not to indicate an optimal Phantom Level location. It is clear that for the new triangle (less obtuse) the corresponding circumcenter CC' will move toward the triangle and its centroid (note that it will be generally different from the centroid, but closer to it than the original circumcenter). The further we move L' from L the more we reduce the likelihood that L will be chosen. However, since L' is a purely mathematical level, we will only use it in the distance metric, as will be explained below. Thus, if L' is the closest to the current modified input, we will take this as indicating to use the actual level L as output state and also calculate the error with respect to L, thus maintaining the original error-diffusion approach. L' thus only influences when output state L is chosen.



(a)

(b)

Figure 11. Scenario used for the Experimental Result section. The output states [circles in (a)] are [(1, 0), (0, 1), (0.4, 0.4)] with a circumcenter [square in (a)] of (1.7, 1.7) and the associated absolute error in (b).



(a)

(b)

Figure 12. Adding a Phantom Level at (0.2, 0.2) [black square in (a)] limits the Modified Input (a) and clearly reduces the absolute error (b).

With the introduction of the Phantom Level, we achieved two things that we assumed to be desirable for the processing: (1) we can maintain the current distance metric implemented in most—if not all—algorithms and (2) we can modify the likelihood that each of the three output states are chosen.

EXPERIMENTAL RESULTS

In order to visualize the proposed approach, we picked a less extreme set of values compared to the previous case. The actual usable output states are (1, 0), (0, 1) and (0.4, 0.4) with the third state being the obtuse L and the one that will be augmented using a Phantom Level L'. For this triangle, we get a circumcenter location CC = (1.7, 1.7) and a centroid location CT = (0.467, 0.467).

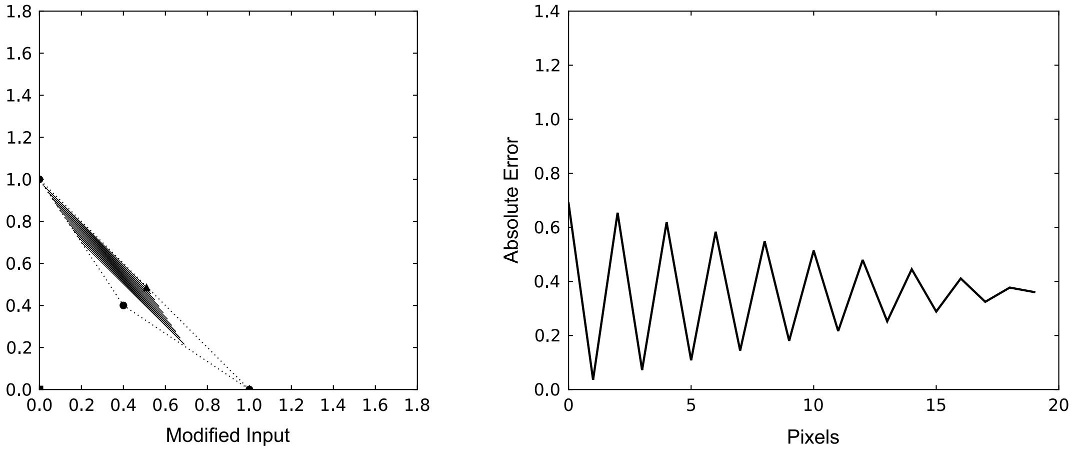
As can be seen on the left of Figure 11, the absolute error approaches 1.2 within a few pixels to stabilize for the next few pixels. As had been shown in Fig. 8 these short term stabilizations are common and the Modified Input will continue to increase to approach circumcenter (1.7, 1.7), indicated as square in (a). We stopped after 25 pixels since the effect is already visible and the traces for the Modified Error are easier to follow.

We can now introduce a Phantom Level. For simplicity we will pick Phantom Levels on the 45° line ($y = x$) and to the left of L to create a less obtuse triangle. With our argumentation, a Phantom Level of $L' = (0.2, 0.2)$ would move its circumcenter to $(0.76, 0.76)$ and thus should reduce the error both in absolute as well as in the vectorial sense of Modified Input.

This is shown in Figure 12, where the black square indicates the Phantom Level.

Note that the error has come down in both definitions as predicted in the previous sections.

One can further move the Phantom Level, say to $L' = (0, 0)$ and would expect that the error is even further decreased. The circumcenter for this case is at $(0.5, 0.5)$ and thus inside (on the boundary) of the triangle defined by our



(a)

(b)

Figure 13. Using the same output states and input data as in Figs. 11 and 12, but moving the Phantom Level to (0, 0) [black square in (a)], further reducing both Modified Input (a) and absolute error (b).

available physical output states. This is shown in Figure 13 where the Modified Input is further reduced (in (a)) and the absolute error (in (b)) is now predominantly within the dynamic range.

From Figs. 11–13 it is clear that the error follows the expected behavior and that we are thus able to reduce the local error to within one dynamic range, as we are commonly experiencing scalar error diffusion.

Note, that in this article we are not making a judgment about the visual quality of a print, rather we are looking at the mathematical errors and attempt to control them via a Phantom Level.

CONCLUSIONS AND FUTURE WORK

Vector Error diffusion exhibits some non-intuitive behaviors that can manifest themselves as large local error leading to image artifacts. These large errors will only occur in some specific areas of color space where the output states define strongly obtuse triangles. It is those areas only that need to be modified following the described approach. Since the affected colors might be part of the calibration sheets that are used to define the system, they still potentially have an impact on the overall color reproduction and it is thus beneficial to control the error accumulation.

We have traced these errors to the circumcenter locations created by an L2 distance metric and proposed the use of Phantom Levels to maintain an L2-norm algorithm, while simultaneously changing the decision boundaries for the output states. The Phantom Level was used to create a virtual circumcenter that more closely resembles a centroid of the relevant triangle and this was used in the decision process to select the output. Whenever the Phantom Level was selected as output, the actual available level was substituted and the actual error caused by this was used in the error feedback. In that manner, the overall error correction is maintained and only locally modified. We could show that this strongly reduces the errors in the process and at that a simple mathematical prediction for the resultant error exists.

This approach has the advantage that the error-diffusion process stays largely the same and that the Phantom Level only gets introduced for very few input areas. Even there only the selection mechanism was changed, maintaining the actual output states and the overall error compensation.

An additional advantage of the Phantom Level is the ability to simply generalize to arbitrary number of dimensions for the data and a standard tetrahedral description. This should also allow us to generalize to spectral error diffusion where for a standard L norm large errors have been observed in the past.

It should be noted that we concentrated on the error accumulation and that resultant changes in visual quality are not a subject of this article. Texture development, edge/detail reproduction are important topics that will be examined in subsequent work. Here it is understood that locally increasing the error might actually have some visual advantage.[12] Future work will examine this and includes the visual effects of limiting the local errors as well as its influence on local structures and textures. Additional future work will be on the generalization to spectral data, considering the introduction of Phantom Spectra to achieve a similar error limitation as in the above description.

REFERENCES

[1] J. A. G. Hale, Proc. S.I.D. **17**, 63 (1976).
[2] R. W. Floyd and L. Steinberg, Proc. S.I.D. **17**, 75 (1976).
[3] J. F. Jarvis, C. N. Judice, and W. H. Ninke, “A survey of techniques for the display of continuous tone pictures on bilevel displays,” Comput. Graph. Image Process. **5**, 13–40 (1976).
[4] J.-N. Shiau and Z. Fan, “Set of easily implementable coefficients in error diffusion with reduced worm artifacts,” Proc. SPIE **2658**, 222 (1996).
[5] J. Kim and K. Gill, “Design of optimal filters for error feedback quantization of monochrome pictures,” Inf. Sci. **39**, 285–298 (1986).
[6] B. W. Kolpatzik and C. A. Bouman, “Optimized error diffusion for image display,” J. Electron. Imaging **1**, 277–292 (1992).
[7] R. Eschbach, “Reduction of artifacts in error diffusion using input dependent weights,” J. Electron. Imaging **2**, 352–358 (1993).

[8] R. Eschbach and Z. Fan, "Complex-valued error diffusion for off-axis computer generated holograms," Appl. Opt. **32**, 3130–3136 (1993).

[9] C. Yu, X. Li, and D. J. Metcalfe, "Re-ordered error diffusion for fast implementation," US-A 9,007,659 (2015).

[10] D. W. Couwenhoven, K. E. Spaulding, and R. L. Miller, "Adaptive error diffusion method," US-A 5,757,517 (1998).

[11] C. Billotet-Hoffmann and O. Bryngdahl, "On the error diffusion technique for electronic halftoning," Proc. S.I.D. **24**, 75–77 (1983).

[12] R. Eschbach and K. T. Knox, "Error diffusion algorithm with edge enhancement," J. Opt. Soc. Am. A **8**, 1844–1850 (1991).

[13] M. Broja, K. Michalowski, and O. Bryngdahl, "Error diffusion concept for multi-level quantization," Opt. Commun. **79**, 280–284 (1990).

[14] K. T. Knox and R. Eschbach, "Threshold modulation in error diffusion," J. Electron. Imaging **2**, 185–192 (1993).

[15] M. T. Orchard and C. A. Bouman, "Color quantization of images," IEEE TIP **29**, 2677–2690 (1991).

[16] M. Broja, R. Eschbach, and O. Bryngdahl, "Stability of active binarization processes," Opt. Commun. **60**, 353 (1986).

[17] R. V. Klassen, R. Eschbach, and K. A. Bharat, "Method and apparatus for vector error diffusion with output color control," US-A 5, 621, 546 (1997).

[18] T.-Y. Park, Y.-H. Cho, M.-Y. Lee, and Y.-H. Ha, "Fast multi-level vector error diffusion based on adaptive primary color selection," Proc. SPIE **5293** (2004).

[19] D. Nyström and O. Norberg, "Improved spectral vector error diffusion by dot gain compensation," Proc. SPIE **8652**, 86520L (2013).

[20] J. Gerhardt, "*Spectral color reproduction: model based and vector error diffusion approaches*," Ph.D thesis (Ecole Nationale Supérieure des Télécommunications, 2007).

[21] J. Gerhardt and J. Y. Hardeberg, "Colour reproduction by vector error diffusion," *Proc. IS&T CGIV2006: 3rd European Conf. on Colour in Graphics, Imaging, and Vision* (IS&T, Springfield, VA, 2006), pp. 469–473.

[22] R. Eschbach, "Some funny things about error diffusion," Proc. SPIE **5293** (2004).