

Histogram Equalization using Neighborhood Metrics

Mark Eramian
Department of Computer Science
110 Science Place
Saskatoon, SK
Canada, S7N 5C9
eramian@cs.usask.ca

David Mould
Department of Computer Science
110 Science Place
Saskatoon, SK
Canada, S7N 5C9
mould@cs.usask.ca

Abstract

We present a refinement of histogram equalization which uses both global and local information to remap the image greylevels. Local image properties, which we generally call neighborhood metrics, are used to subdivide histogram bins that would be otherwise indivisible using classical histogram equalization (HE). Choice of the metric influences how the bins are subdivided, affording the opportunity for additional contrast enhancement. We present experimental results for two specific neighborhood metrics and compare the results to classical histogram equalization and local histogram equalization (LHE). We find that our methods can provide an improvement in contrast enhancement versus HE, while avoiding undesirable over-enhancement that can occur with LHE and other methods. Moreover, the improvement over HE is achieved with only a small increase in computation time.

1. Introduction

The ideal greyscale image histogram is perfectly flat and makes use of every available grey value in the image format [11, 12, 13]. In general, classical histogram equalization (HE) [11, 12] cannot come close to this ideal. Often the histogram resulting from HE contains “gaps”, that is, empty bins between very full bins. Figure 1(b), which depicts the result of equalizing the histogram in Figure 1(a), illustrates this phenomenon. The large bins in this histogram have not been subdivided; they have only been spread out. However, the small bins in the right of the original histogram have been combined to form bins of close to the optimal size; thus, if the large bins could be subdivided into bins of less than optimal size, we would be able to recombine them such that bins in the final histogram would be close to optimal size. If an image is N pixels by M pixels in size, then the

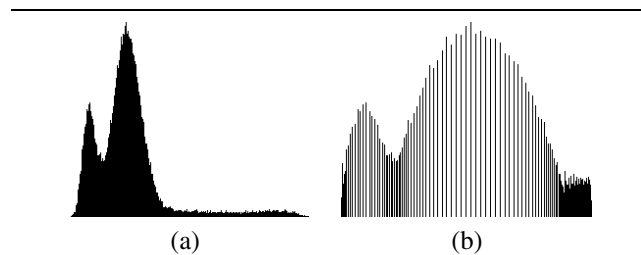


Figure 1. (a) an image histogram, (b) the result of (a) after classical histogram equalization.

optimal bin size B is the total number of pixels in the image divided by the number of greylevel intensities D (referred to in the sequel as the image depth):

$$B = \frac{MN}{D}.$$

If all bins are of size B , then the histogram has the ideal characteristics described above.

One way of avoiding large bins is to add a small amount of uniform noise to each pixel's intensity value [11, 12] prior to equalization. While this approach produces a flat histogram, the noise is added without any regard to the original structure of the image, potentially reducing image quality. A similar idea [11] is to compute the average greylevel in a neighborhood around each pixel, assigning lower output values to pixels with lower local averages.

Local histogram equalization (LHE) [8, 11] uses a sliding window method in which, for each pixel, local histograms are computed from the windowed neighborhood to produce a local greylevel remapping for each pixel. The greylevel of the pixel at the center of the neighborhood is changed according to the local greylevel remapping for that pixel. LHE is capable of great contrast enhancement which can sometimes be considered

over-enhancement. LHE-based methods are generally require more computation than other methods because a local histogram needs to be built and processed for every image pixel.

Researchers have designed many variations of LHE. Stark [14] uses a method in which the cumulation function can be altered to control the degree of enhancement. Caselles et al. [3] propose a local histogram contrast enhancement algorithm which preserves the level sets of an image. Paranjape et al. [10] generate adaptive neighborhoods for each pixel by differentiating foreground and background pixels within the neighborhood and using only the foreground pixels to build the local histograms. Dale-Jones and Tjahjadi [6] describe a method in which the window size for building local histograms is varied over the image depending on local image characteristics. All of these methods share the drawback of requiring a separate histogram equalization process for every image pixel.

Leu [9] describes a method which amplifies contrast along edges. Edge pixels are located by standard edge-detection methods and thresholding, and a histogram is created using only edge pixels. This histogram is used to identify intensity levels that should not be merged, in order to maximize contrast around edges. An intensity transformation function is derived from this information and applied to the entire image. Leu's, and similar methods [1], depend heavily on successful detection of edges – a well-known, but difficult problem in real images.

Cheng et al. [4] describe a method where a local homogeneity measurement is used to control the degree to which local contrast is enhanced. The homogeneity measure is based on local measures such as edge value, standard deviation, entropy, and others. The level of enhancement is controlled by an amplification constant which is determined from the local homogeneity and the global histogram. High homogeneity lowers the constant and narrower histograms increase the constant (respectively vice versa). Although this method is capable of avoiding over- or under-enhancement, it is more computationally intensive than other methods.

Our proposed contrast enhancement method also uses local information about pixels as well as global histogram information. It is a generalization of the previously mentioned method of subdividing bins based on average neighborhood grey values [11]. We formulate our solution as histogram equalization using an arbitrary neighborhood metric, and we investigate some old and new neighborhood metrics.

We shall see that our proposed method has the following characteristics: its simplicity and small neighborhood size yield a fast algorithm; it usually produces a very flat histogram; it is able to amplify edge contrast without explicitly detecting edge pixels; and it preserves image structure by not overly-enhancing image contrast.

Algorithm 1. Classical Histogram Equalization

```

for each pixel  $p$  in the image do
    deposit  $p$  in temporary bin  $b_{g(p)}$ .
end for
 $j \leftarrow 0$ 
for each temporary bin  $b_i$  do
    Copy pixels  $b_i$  into histogram bin  $h_{(j/D)}$ .
     $j \leftarrow j + |b_i|$   $\{ |b_i| = \text{number of pixels in } b_i \}$ 
end for
for  $i = 0$  to  $D - 1$  do
    Set greylevel of each pixel in bin  $h_i$  to  $i$ .
end for

```

We are motivated directly by the basic goals of histogram equalization: to produce a perfectly flat histogram that uses evenly the entire dynamic range of intensity values. This means that we want B pixels of each possible intensity level. This cannot normally be achieved in HE because images frequently have histogram bins with pixel counts well in excess of B , and these bins cannot be subdivided. Our method uses neighborhood metrics to determine if and how these large bins can be subdivided.

In the next section we introduce the notion of sorting functions to be used for subdivision of histogram bins and define some specific metrics which we will study in detail. Section 3 discusses our experimental method, Section 4 details our results, and concluding remarks appear in Section 5.

2. Histogram Equalization with Neighborhood Metrics

Let $[a, b]$ denote the closed integral interval from a to b . An image with dimensions N by M and depth D is a function $g : [0, N - 1] \times [0, M - 1] \rightarrow [0, D - 1]$ where $g(x, y)$ is the grey level of pixel (x, y) . We may also write $g(p)$ to denote the greylevel of a pixel p .

The classical histogram equalization algorithm can be expressed as in Algorithm 1. This specific formulation of classical histogram equalization allows for a simple generalization to multiple sorting functions. Algorithm 2 will illustrate this generalization.

2.1. Sorting Functions

We generalize the classical histogram algorithm to allow any number of *sorting functions* on image pixels in place of $g(p)$ in Algorithm 1. The range of the sorting functions define a set of temporary bins for the algorithm. This allows us to choose functions that can order pixels using different cri-

Algorithm 2. Histogram Equalization with Generalized Sorting Functions

Let the sorting functions be λ_1 through λ_k .
for each pixel p in the image **do**
 deposit p in temporary bin $b_{(\lambda_1(p), \lambda_2(p), \dots, \lambda_k(p))}$.
end for
Sort temporary bins using λ_1 as the primary sort key, λ_2 as the secondary sort key, etc.
 $j \leftarrow 0$
for each temporary bin b_i in sorted order **do**
 Copy pixels b_i into histogram bin $h_{(j/D)}$.
 $j \leftarrow j + |b_i|$ $\{|b_i| = \text{number of pixels in } b_i\}$
end for
for $i = 0$ to $D - 1$ **do**
 Set greylevel of each pixel in bin h_i to i .
end for

teria, and to separate pixels that would be in the same bin in the original histogram into several of the temporary bins defined by the sorting functions. Allowing multiple sorting functions allows a more complete ordering of pixels by multiple sort keys. This generalized histogram equalization algorithm is given as Algorithm 2.

2.2. Neighborhood Metrics

We now consider in detail some sorting functions; we call them *neighborhood metrics* since they are functions evaluated using greylevels of pixels in a local neighborhood of the input pixel. In this section, \mathbb{R} denotes the real numbers, and γ is the function which extends an image function to be surrounded by a “background” of zero greylevel:

$$\gamma(x, y) = \begin{cases} g(x, y), & (x, y) \in [0, N - 1] \times [0, M - 1] \\ 0, & \text{otherwise} \end{cases}$$

For our first metric, we define within our framework the *neighborhood average metric* suggested by Rosenfeld and Kak[11] and discussed in the introduction. It is the function $\bar{\alpha}_m : [0, N - 1] \times [0, M - 1] \rightarrow \mathbb{R}$ such that

$$\bar{\alpha}_m(x, y) = \frac{\sum_{(x', y') \in R_m^{(x, y)}} \gamma(x', y')}{|R_m^{(x, y)}| - 1}$$

where $R_m^{(x, y)}$ is the set of pixels forming a square m by m neighborhood centered on (x, y) and m is a positive odd integer. Hence $\bar{\alpha}_m(x, y)$ is the average grey value of pixels in the m by m neighborhood centered on (x, y) .

We now introduce two new metrics, the first of which is the *inverted neighborhood average metric*. This metric, denoted α_m , is defined as the average grey value of the pixels in the square neighborhood of size m by m about pixel

(x, y) , subtracted from the grey value of (x, y) . Formally, we define $\alpha_m : [0, N - 1] \times [0, N - 1] \rightarrow \mathbb{R}$ as

$$\begin{aligned} \alpha_m(x, y) &= \gamma(x, y) - \frac{\sum_{(x', y') \in R_m^{(x, y)}} \gamma(x', y')}{|R_m^{(x, y)}| - 1} \\ &= \gamma(x, y) - \bar{\alpha}_m(x, y). \end{aligned}$$

We suggest this metric because for pixels with the same original greylevel, pixels with higher average greylevel of neighborhood pixels will be remapped to lower greylevels than pixels with lower average greylevel of neighborhood pixels if the original bin is split. The result is that pixels will tend to take on greylevels that are further away from their neighborhood average greylevel than under HE.

Our second new neighborhood metric we call the *neighborhood voting metric*. It requires the following *voting function*:

$$v(x, y, x', y') = \begin{cases} 1, & \gamma(x, y) > \gamma(x', y') \\ 0, & \text{otherwise} \end{cases}$$

The neighborhood voting metric, written β_m , is defined as the number of pixels in the m by m square neighborhood centered on (x, y) whose grey value is strictly less than that of $\gamma(x, y)$. The voting metric will tend to force pixels which have more neighbors with smaller grey level to a higher intensity (and vice versa) if and when the bin is subdivided. Formally, $\beta_m : [0, N - 1] \times [0, M - 1] \rightarrow [0, m^2]$ is the function

$$\beta_m(x, y) = \sum_{(x', y') \in R_m^{(x, y)}} v(x, y, x', y').$$

We propose to use these neighborhood metrics as sorting functions (as in Algorithm 2) to subdivide large bins in the original histogram. To accomplish this we will use Algorithm 2 with $\lambda_1 = g$ and λ_2 will be one of our neighborhood metrics. This will have the effect of subdividing the original histogram bins into sub-bins where pixels in each sub-bin share the same greylevel and neighborhood metric value. The sub-bins are then sorted by greylevel with ties broken by the neighborhood metric (λ_2). Pixels in each sub-bin are assigned in order to bins in the output histogram such that each output bin contains approximately the optimal number of pixels B . We expect that this subdivision of bins will result in very few bins that are larger than the optimal bin size, allowing us to build a very flat histogram that uses most, if not all, of the available greylevels.

For our experiments with the neighborhood metrics, we used a slightly modified instance of Algorithm 2. In the modified version, shown as Algorithm 3, we fill the output histogram bins h_j sequentially until they are “optimally full”. When this occurs we start filling the next output histogram bin. As stated in Section 1, the optimal bin size is B . Letting $|b|$ denote the number of pixels in a bin b , then the current histogram bin h_j in Algorithm 3 is considered “optimally full” if $|h_j|$ is at least $|b|/2$ less than B . That is, we

Algorithm 3. Histogram Equalization with Neighborhood Metric λ

for each pixel p in the image **do**
 deposit p in temporary bin $b_{g(p),\lambda(p)}$.
end for
Sort temporary bins using g as the primary sort key and λ as the secondary sort key.
 $j \leftarrow 0$
for each temporary bin b_i in sorted order **do**
 {The current bin h_j is considered “full” if it contains B pixels. Thus, if less than half of b_i fits into h_j then start filling h_{j+1} .}
 if $B - |h_j| < |b_i|/2$ **then**
 $j \leftarrow j + 1$
 end if
 Copy pixels in b_i into histogram bin h_j .
end for
if $j < D - 1$ **then**
 Respace bins h_0 through h_j evenly over h_0 through h_{D-1} .
end if
for $i = 0$ to $D - 1$ **do**
 Set greylevel of each pixel in bin h_i to i .
end for

never overfill a bin by more than half the size of the current temporary bin b .

A side effect of this modified bin-filling strategy is that the non-empty output histogram bins may not span the entire range of greylevels (one of the goals of equalization). If this is the case, then, as a final step, we redistribute the nonempty bins equally over the entire range of bins. We note that doing so does not alter the “flatness” of the histogram (relative to the flatness if we did not redistribute the bins) in the way it is defined in the following section.

3. Experiments

Our experiments focus on the neighborhood metrics α_3 and β_3 . We compare Algorithm 3 using these two metrics to HE, LHE, and Algorithm 3 using the metric $\overline{\alpha}_3$. The comparison was made using three quality measures: contrast-per-pixel, histogram flatness and image distortion. These measures respectively reflect the three goals of improving contrast, flattening the histogram, and minimizing deleterious effects on image structure.

We define contrast-per-pixel C of an image as

$$C = \frac{\sum_{i=0}^N \sum_{j=0}^M \left(\sum_{(m,n) \in R_3^{(i,j)}} |\gamma(i,j) - \gamma(m,n)| \right)}{MN}.$$

Intuitively, this is the average difference in greylevel between adjacent pixels.

To measure the flatness σ of a histogram h we compute the variance of the bin sizes:

$$\sigma = \frac{\sum_{i=0}^{D-1} (|h_i| - \mu_h)^2}{D},$$

where $|h_i|$ is the size of the i -th bin of the image’s histogram, and $\mu_h = \frac{1}{D} \sum_{i=0}^{D-1} |h_i|$ is the mean histogram bin size. A smaller value of σ indicates a flatter histogram. The flatness measure indicates the degree of success towards both reducing the number of empty bins and ensuring that each bin has an equal number of pixels.

To measure the dissimilarity or distortion of the structure between two images with grey functions g_1 and g_2 , defined over $[0, M - 1] \times [0, M - 1]$, we compute the standard deviation of the ratios of pixel grey levels pairwise in g_1 and g_2 . This measure, δ , can be thought of as the standard deviation of local change in contrast. Formally it is defined as

$$\delta = \frac{1}{MN} \sum_{(i,j) \in [0, M-1] \times [0, N-1]} \left(\frac{g_1(i,j)}{g_2(i,j)} - \mu_{ij} \right)^2$$

where the quantity $\mu_{ij} = \frac{1}{MN} \sum_{i,j} \frac{g_1(i,j)}{g_2(i,j)}$ is the mean ratio. If $g_2(i,j) = 0$, then pixel (i,j) is excluded from the sums. Ratios of pixel greylevels have been used to measure (dis)similarity of images in fractal image coding and compression algorithms [5, 7].

4. Results

We tested our neighborhood metrics on 8-bit greyscale samples ($D = 256$) of all 112 Brodatz textures [2]. Each sample had dimensions of 256 by 256 pixels.

4.1. Aggregate Results

Table 1 contains the aggregate results of this experiment. Each row of the table gives results for one of our quality measures: contrast per pixel C , histogram flatness σ , and image distortion δ . The first data column in the upper table indicates the average result for classical histogram equalization (HE). Subsequent columns give the result for their respective neighborhood metrics relative to the result for HE as an average percentage. For example, the first data row of the lower table indicates that α_3 resulted in a value of C that was, on average, 7.48% greater than that of HE over the 112 images tested. Negative numbers indicate that the average result was less than that of HE.

Following the three goals of histogram equalization, we would like to see the neighborhood metrics maximize contrast while minimizing distortion and the flatness measure,

Table 1. Aggregate results for mean values of quality measures on Brodatz images processed with various algorithms. The absolute results for HE are given, and the rest are given relative to the performance of the HE algorithm.

	HE	$\bar{\alpha}_3$	LHE
Contrast C	29.75	-5.49%	+50.13%
Flatness σ	691.88	-94.04%	-82.12%
Distortion δ	43.24	+8.79%	+225.08%

	α_3	β_3
Contrast C	+7.48%	+3.12%
Flatness σ	-94.03%	-66.12%
Distortion δ	+15.08%	+6.63%

for which lower values mean a flatter histogram with fewer gaps. We note that contrast and distortion are competing measures, since increased contrast also increases distortion. The best result is one that increases contrast significantly while increasing distortion only slightly.

This table shows that the $\bar{\alpha}_3$ metric produces, on average, an image with less contrast than HE, although the histogram is very flat and distortion is fairly low. The α_3 neighborhood metric produces a good combination of results – contrast enhancement is better than HE by an average of 7.48%, the histogram is 94% flatter, and not too much distortion is introduced in excess of that for HE (8.79%). The β_3 neighborhood metric also shows a contrast improvement over HE of 3.12% on average, a 66.12% flatter histogram and has a very small amount of distortion – only 6.63% more than HE. While LHE improves contrast by an average of about 50% and produces a histogram that is on average 82% flatter, LHE distorts the images by about 225% more than HE on average.

In Section 4.2 we will show that for images whose histograms consist mainly of narrow peaks, the β_3 metric produces a result that is superior to that of the other methods tested. However, for images that have high local variance in greylevel and broader histograms, the differences between the methods we are investigating become far less pronounced.

Consider the scatter plot in Figure 2. This plot shows for each of the Brodatz textures the contrast per pixel of the image after classical equalization plotted against the contrast per pixel of the same image after equalization with neighborhood metric β_3 . The dashed line is the line of slope 1. Thus, points that fall above the line indicate images for which β_3 resulted in higher contrast. Only 17 of 112 test points fall below the line.

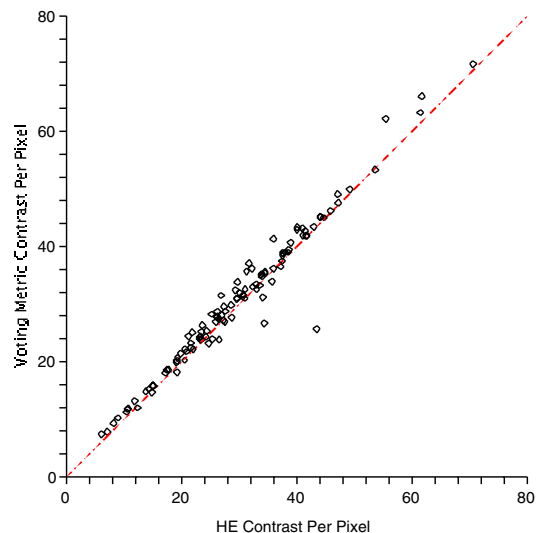


Figure 2. Scatter plot of contrast per pixel of the Brodatz images after classical equalization vs contrast per pixel of same images after equalization with neighborhood metric β_3 .

Figure 3 shows the same plot for the flatness quality measure. We see that all points fall considerably below the line. Since lower flatness measures are more desirable, this demonstrates that β_3 achieves a dramatic improvement in flatness over classical equalization. All 112 test points fall below the line.

Similar plots were examined for each neighborhood metric and quality measure. Though omitted due to space constraints, these plots confirm that although our methods do not always result in drastic improvement over HE, they usually result in higher contrast than HE and always produce a flatter histogram than HE.

4.2. Specific Examples

The benefits of our methods are best observed in images whose histograms consist of closely spaced narrow peaks. Figures 4 and 5 show two examples of such images and the results for each of the 5 tested methods. We see that over-enhancement is best suppressed by β_3 . Figure 4 depicts the results for Brodatz texture D25 (brick). Figures 4(b), 4(c) and 4(d) reveal what appear to be variations in the reflectance of the surface which are not apparent in the original and are not overly enhanced by β_3 (Figure 4(e)).

In Figure 5 are the results for Brodatz texture D102 (cane). We see that HE, $\bar{\alpha}_3$, and α_3 (Figures 5(b), (c) and (d)) again over-enhance the slight intensity variation in the

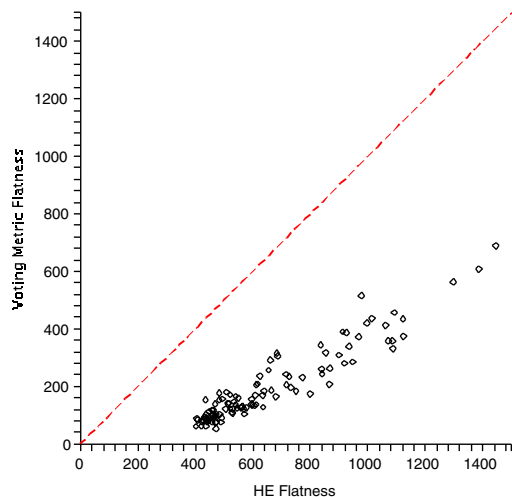


Figure 3. Scatter plot of flatness of the Brodatz images after classical equalization vs flatness of same images after equalization with neighborhood metric β_3 .

cane while β_3 in Figure 5(e) is much more faithful to the original while revealing some subtle texture on the cane not visible in the original.

4.3. Inherent Edge Enhancement

The neighborhood metrics α_m and β_m also have the ability to visually enhance edges, as demonstrated in the series of images in Figure 6. Figure 6(a) shows an image of vertical bars of increasing greylevel. Figures 6(b) and 6(c) show the image after histogram equalization with α_3 and β_3 respectively. Figures 6(d), 6(e) and 6(f) show a closeup of an edge in the images in Figures 6(a), 6(b) and 6(c) respectively. We can see that edge contrast is enhanced by adjustments made to the grey values of pixels very close to the edges of bands. This illustrates how these neighborhood metrics can enhance contrast by tending to shift pixel grey values further away from those of their neighbors when the subdivision of an original histogram bin occurs. Note that the edge pixels are not explicitly detected, but rather the chosen neighborhood metrics are sensitive to the relationship between the grey level of a pixel and that of its neighbors. Thus, our methods can target edges as areas where contrast should be maintained or improved without the computational overhead of explicitly identifying edge pixels.

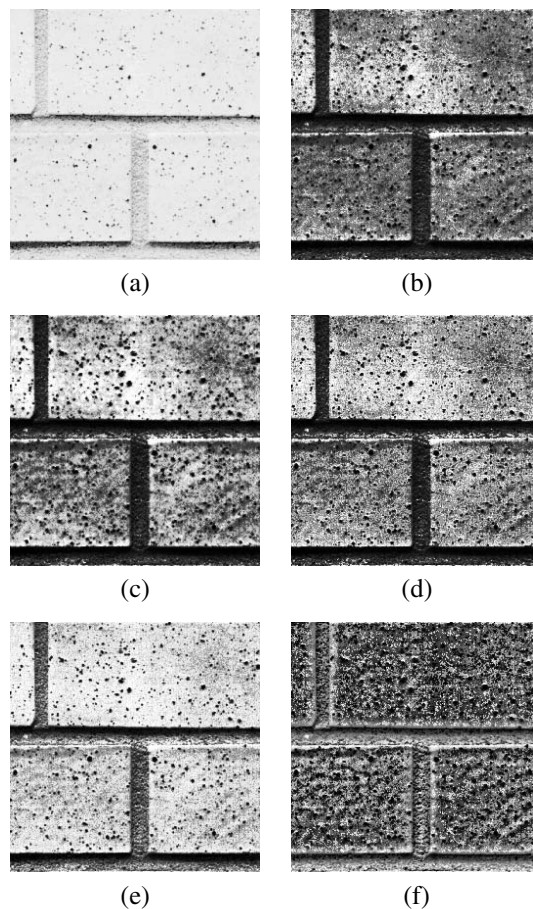


Figure 4. Results for D25 (brick). (a) Original, (b) Equalized using HE, (c) α_3 , (d): α_3 , (e): β_3 , (f): LHE.

4.4. Complexity Analysis

We analyze the complexity of Algorithm 3 for an M by N pixel image and a neighborhood size of m . Computation of the neighborhood metric for each pixel requires $O(MNm^2)$ time. Adding each pixel to its appropriate temporary bin is $O(MN)$. In the worst case, each pixel is placed in its own temporary bin, requiring that we sort MN temporary bins. We can sort these in time proportional to $O(MN \log(MN))$.

The running time of the while loop is proportional to the number of temporary bins. In the worst case, there is one bin for each pixel, so the entire while loop executes in time $O(MN)$.

The re-spacing requires $O(D)$ time since we don't have to move the pixels to different bins. Relabeling of the bins suffices.

The last step of generating g^l is $O(MN)$. The total time

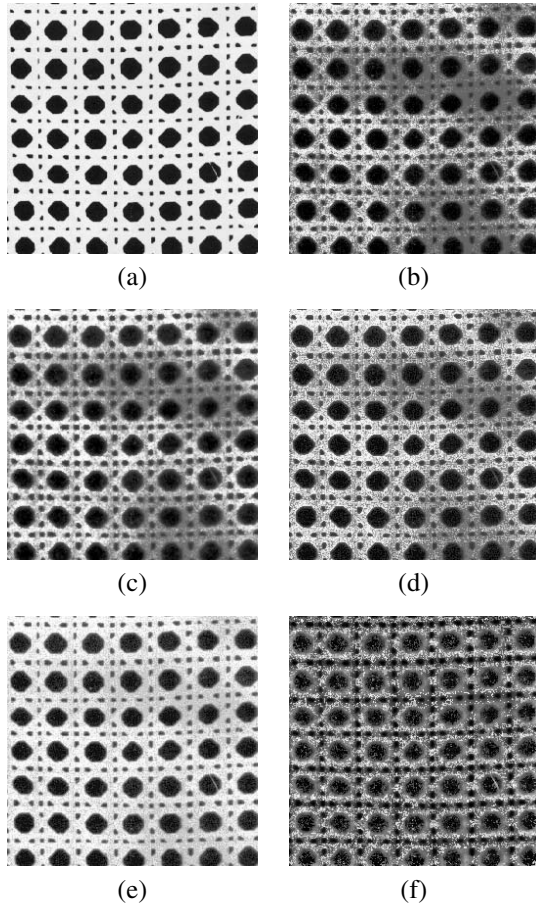


Figure 5. Results for D102 (cane). (a) Original, (b) Equalized using HE, (c) α_3 , (d) α_3 , (e) β_3 , (f) LHE.

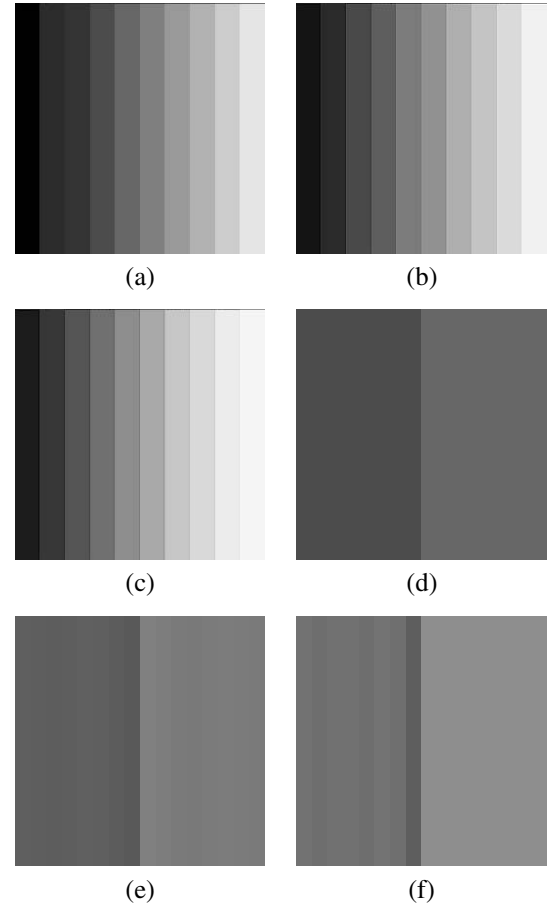


Figure 6. Demonstration of edge enhancement when using histogram equalization with neighborhood metrics. (a) Original image, (b) Algorithm 3 with α_3 , (c) Algorithm 3 with β_3 , (d) edge closeup of (a), (e) edge closeup of (b), (f) edge closeup of (c).

complexity is thus

$$\begin{aligned} & O(D + 3MN + MNm^2 + MN \log MN) \\ &= O(MN(m^2 + \log MN) + D). \end{aligned}$$

In the specific case where $\lambda_2 = \beta_m$, the sorting can be done in $O(MN)$ time because a radix sort can be used to perform the sorting. since the range of β_m is the integral interval $[0, 8]$. This reduces the overall running time to $O(MNm^2 + D)$. For comparison, the time complexity of the basic form of LHE is $O(MN(mn + D^2))$ and the m by n window size can be as large as 100 by 100 in some applications; the time complexity is higher yet for most extensions of the basic LHE method.

4.5. Effect of Neighborhood Size

In experimenting with the value of m , we found that for the metrics tested, values of m greater than 3 did not im-

prove the results which, as expected, converged with the results for HE as the value of m increased. Since, for the neighborhood metrics studied, there appears to be no advantage to using a window size larger than 3, we may consider m as a constant value, resulting in a time complexity of $O(MN \log(MN) + D)$ for histogram equalization using our neighborhood metrics.

5. Summary and Conclusions

We described a variation of histogram equalization which uses both local and global information in order to achieve a more strict partial order on the image pixels. Accordingly, our method is able to achieve flatter histograms and better use of the available greylevels than

HE while mitigating the overenhancement that often occurs with LHE.

We proposed a general framework which orders pixels based on a sequence of sorting functions. We suggested using the original image greylevel as the primary key and a neighborhood metric as the secondary key. We investigated the *neighborhood average metric*, the *inverted neighborhood average metric*, and the *voting metric*, applying them to the Brodatz textures and comparing them to HE and LHE using three quality metrics. We found that the inverted neighborhood average and voting metrics gave an improvement in contrast and histogram flatness over classical histogram equalization, while having significantly less distortion than local adaptive histogram equalization. The previously proposed neighborhood average metric increases flatness, but also increases distortion while decreasing contrast. For images with histograms consisting of very narrow peaks, the voting metric gave results that were visually superior to those of any of the other methods.

In summary, the neighborhood metrics achieve better histogram characteristics than HE while avoiding the large distortions and computational overhead of LHE. Also, our method is simpler than most of the other methods described in Section 1 and achieves additional enhancement of edge contrast without the explicit detection of edge pixels. Our method requires approximately 1/10 of a second for a 256 by 256 image on a Pentium 4, 3GHz PC. The computation times for LHE and its variations on comparably sized images are on the order of a few to several seconds.

We have not exhausted the space of possible neighborhood metrics. Future work might involve the design of additional metrics, either for use in specific domains or with desirable properties for image processing in general. We may also investigate using more than two, possibly competing metrics. More elaborate measures can be considered in the case of color images.

Acknowledgments: The authors thank Matthew Drotar for his implementation of some of the algorithms in this paper. This research was supported in part through NSERC grant number RGPIN262027-03, in part through an NSERC Undergraduate Research Award, and in part through institutional grants from the University of Saskatchewan.

References

- [1] A. Beghdadi and A. Le Negrate. Contrast enhancement technique based on local detection of edges. *CVGIP*, 46(2):162–174, May 1989.
- [2] P. Brodatz. *Textures; A Photographic Album for Artists and Designers*. Dover Publications, 1999.
- [3] V. Caselles, J.-L. Lisani, J.-M. Morel, and G. Sapiro. Shape preserving local histogram modification. *IEEE Transactions on Image Processing*, 8(2):220–224, February 1999.
- [4] H.-D. Cheng, M. Xue, and X. Shi. Contrast enhancement based on a novel homogeneity measurement. *Pattern Recognition*, 36(11):2687–2697, November 2003.
- [5] K. Culik II and J. Kari. Image compression using weighted finite automata. *Comput. and Graphics*, 17(3):305–313, 1993.
- [6] R. Dale-Jones and T. Tjahjadi. A study of modification of the local histogram equalization algorithm. *Pattern Recognition*, 26(9):1373–1381, 1993.
- [7] M. G. Eramian. *Image Texture Analysis using Weighted Finite Automata*. PhD thesis, The University of Western Ontario, 2002.
- [8] D. J. Ketchum. Real-time image enhancement techniques. *Proc. SPIE/OSA*, 74:120–125, 1976.
- [9] J.-G. Leu. Image contrast enhancement based on intensities of edge pixels. *CVGIP: Graphical Models and Image Processing*, 54(6):497–506, 1992.
- [10] R. B. Paranjape, W. M. Morrow, and R. M. Rangayyan. Adaptive-neighborhood histogram equalization for image enhancement. *CVGIP: Graphical Models and Image Processing*, 54(3):259–267, May 1992.
- [11] A. Rosenfeld and A. C. Kak. *Digital Picture Processing*. Computer Science and Applied Mathematics. Academic Press, 1982.
- [12] L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice-Hall, Inc., New Jersey, 2001.
- [13] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, 1999.
- [14] J. A. Stark. Adaptive image contrast enhancement using generalizations of histogram equalization. *IEEE Transactions on Image Processing*, 9(5):889–896, May 2000.