DIGITAL IMAGE PROCESSING

Histogram Equalization, Histogram Matching & Spatial Domain Filtering (Gonzalez & Woods)

1. Histogram Equalization

1.1 Introduction

The histogram of a digital image with gray levels in the range [0, L−1] is a discrete function:

$$h(r_k) = n_k$$

where rk is the k-th gray level and nk is the number of pixels in the image with that gray level. It is common practice to normalize the histogram by dividing each value by the total number of pixels, MN:

$$p(r_k) = n_k / (MN), \quad k = 0, 1, 2, \ldots, L-1$$

p(rk) gives an estimate of the probability of occurrence of gray level rk. Histogram equalization (also called histogram linearization) is a technique for obtaining a uniform histogram — an image with contrast enhanced automatically by remapping gray levels so they are spread as evenly as possible over the full range of intensities.

1.2 Theoretical Basis (Continuous Case)

For a moment, consider intensity levels as continuous quantities in the range [0, L−1], and let pr(r) denote the probability density function (PDF) of the intensity levels in a given image, r representing the intensities. Suppose a transformation of the form:

$$s = T(r), \quad 0 \le r \le L - 1$$

is applied to produce output intensity s, and suppose that T(r) satisfies the conditions:

- (a) T(r) is a monotonically increasing function in the interval $0 \le r \le L-1$, and
- (b) $0 \le T(r) \le L-1$ for $0 \le r \le L-1$.

Condition (a) preserves the increasing order from black to white in the output image, and condition (b) guarantees that the output intensities are in the same range as the input. If T(r) is strictly monotonic, the inverse transformation $r = T^{-1}(s)$ exists.

From basic probability theory, if pr(r) and T(r) are known and T(r) is continuous and differentiable, the PDF of the transformed variable s is:

$$p_s(s) = p_r(r) \left| dr/ds \right|$$

A transformation function of particular importance in image processing is:

$$s = T(r) = (L-1) \int_0^r p_r(w)\, dw$$

where w is a dummy variable of integration. The right side of this equation is the cumulative distribution function (CDF) of random variable r. Using this transformation, ps(s) can be shown to be a uniform density function:

$$p_s(s) = 1 / (L-1), \quad 0 \le s \le L-1$$

This means the transformation function of the above equation generates an image whose intensity levels are equally likely, producing an increase in the dynamic range of pixel intensities — i.e., contrast enhancement.

1.3 Discrete Formulation

For discrete values, probabilities and summations are used instead of probability density functions and integrals. The probability of occurrence of gray level rk is:

$$pr(rk) = nk / n, \quad k = 0, 1, 2, ..., L-1$$

where n is the total number of pixels in the image, nk is the number of pixels with intensity rk, and L is the number of possible intensity levels. The discrete form of the transformation is:

$$sk = T(rk) = (L-1) \Sigma(j=0 \text{ to } k)\ pr(rj) = [(L-1)/n] \Sigma(j=0 \text{ to } k)\ nj$$

Thus, an output image is obtained by mapping each pixel of intensity rk in the input image into a corresponding pixel of level sk in the output image, using the equation above. This is called histogram equalization or histogram linearization.

1.4 Algorithm — Steps

1. Compute the histogram h(rk) = nk of the input image (count of pixels at each gray level).
2. Compute the normalized histogram (PDF): pr(rk) = nk / n, for k = 0, 1, ..., L−1.
3. Compute the cumulative distribution function (CDF): Σ(j=0 to k) pr(rj) for each k.
4. Compute the transformation (mapping): sk = (L−1) × CDF(k), then round sk to the nearest integer.
5. Map every pixel with original intensity rk to the new intensity sk to obtain the equalized output image.

1.5 Numerical Example

Consider a 3-bit (L = 8, levels 0–7) image of size 64 × 64 (n = 4096 pixels) with the following histogram data:

rk	nk	pr(rk) = nk/n
r0 = 0	790	0.19
r1 = 1	1023	0.25
r2 = 2	850	0.21
r3 = 3	656	0.16
r4 = 4	329	0.08
r5 = 5	245	0.06
r6 = 6	122	0.03
r7 = 7	81	0.02

Applying sk = (L−1) Σ pr(rj) = 7 × CDF(k):

k	CDF up to k	sk = 7×CDF	Rounded sk
0	0.19	1.33	s0 = 1
1	0.44	3.08	s1 = 3
2	0.65	4.55	s2 = 5
3	0.81	5.67	s3 = 6

k	CDF up to k	sk = 7×CDF	Rounded sk
4	0.89	6.23	s4 = 6
5	0.95	6.65	s5 = 7
6	0.98	6.86	s6 = 7
7	1.00	7.00	s7 = 7

Since several input levels (s4 with s3, s6/s7 with s5) map to the same output level, the equalized histogram has fewer distinct gray levels (5, 6, and 7 all merge some inputs) — this is expected because histogram equalization on discrete data is only an approximation to a truly uniform output histogram.

1.6 Properties

- The mapping sk = T(rk) is always monotonically non-decreasing, so the relative order of intensities is preserved.
- The transformation automatically determined by the equalization is adaptive — it depends entirely on the histogram of the input image, no parameters need to be specified.
- The resulting histogram is only approximately uniform in the discrete case, since pixels can only be merged (never split) into new bins.

1.7 Advantages

- Fully automatic contrast enhancement technique — no manual parameter tuning.
- Simple and computationally efficient (single pass to build histogram, one more to build CDF and remap).
- Effective for images with poor contrast, where intensities are concentrated in a narrow range.

1.8 Disadvantages / Limitations

- It enhances the contrast of the whole image indiscriminately; it cannot be controlled to produce a specific, desired histogram shape.
- Can over-enhance noise and background clutter, especially in images with large uniform areas.
- May produce unnatural-looking, washed-out results for some images.

1.9 Applications

- Enhancement of X-ray images, satellite and remote-sensing images.
- Preprocessing step for object detection, feature extraction, and pattern recognition.
- Enhancing low-contrast images captured under poor illumination.

2. Histogram Matching (Histogram Specification)

2.1 Introduction

Histogram equalization automatically determines a transformation that seeks to produce an output image with a uniform histogram. This is a good approach when automatic enhancement is needed, but there are applications where a specific histogram shape — not necessarily uniform — is desired for the output image, so that particular features of the image are enhanced. The method used to generate an output image that has a specified histogram is called histogram matching or histogram specification.

2.2 Theoretical Basis (Continuous Case)

Let r denote intensities of the input image, and z denote intensities of the desired output image, treated as continuous random variables with probability density functions pr(r) and pz(z) respectively. pr(r) can be estimated from the input image, while pz(z) is the specified PDF that we wish the output image to have.

Let s be a random variable with the property:

$$s = T(r) = (L-1) \int_0^r pr(w)\, dw$$

This is the same histogram-equalization transformation seen earlier. Suppose next that we define a random variable z with the property:

$$G(z) = (L-1) \int_0^z pz(v)\, dv = s$$

where v is a dummy variable of integration. From these two equations, G(z) = T(r) = s, which means that z must satisfy the condition:

$$z = G^{-1}(s) = G^{-1}[T(r)]$$

where $G^{-1}(s)$ is the inverse transformation of G(z). This equation indicates that the desired output image with the specified probability density pz(z) can be obtained by: (i) equalizing the input image using T(r) to get s, then (ii) applying the inverse transformation function G^{-1} to s, to map it to the desired intensity z. Because G is the transformation function for histogram equalization of the desired image, G^{-1} must be a mapping that performs the inverse (histogram de-equalization) operation.

2.3 Discrete Formulation

In the discrete case, the equalization transformation of the input image is:

$$sk = T(rk) = (L-1) \sum_{j=0 \text{ to } k} pr(rj), \quad k = 0, 1, \ldots, L-1$$

and the equalization transformation of the specified histogram (desired output) is:

$$G(zq) = (L-1) \sum_{i=0 \text{ to } q} pz(zi) = sk$$

from which:

$$zq = G^{-1}(sk)$$

Because G(zq) is a discrete, monotonic transformation function that has been computed from the specified histogram, finding zq for a given value of sk generally involves an iterative search for the value zq that makes G(zq) closest to sk.

2.4 Algorithm — Steps

6. Compute the histogram pr(rk) of the input image and obtain the equalizing (rounded) transformation sk = (L−1) Σ pr(rj).
7. Compute all values of the transformation function G(zq) using the specified (desired) histogram pz(zi): G(zq) = (L−1) Σ pz(zi).
8. Precompute a mapped intensity level zq for each value of sk, by finding the zq whose value G(zq) is closest to sk, i.e., $G^{-1}(sk)$. Where more than one value of zq satisfies the equation, the smallest value is conventionally chosen.
9. Store all mappings T and G^{-1} found in steps 1–3.
10. For every pixel in the original image, map its value rk first to sk using T (histogram equalization), then map sk to its corresponding zq using the G^{-1} mapping found in step 3, to produce the final histogram-specified output image.

2.5 Numerical Example

Consider again a 3-bit (L = 8) image whose equalized values (sk, rounded) were computed as: s0 = 1, s1 = 3, s2 = 5, s3 = 6, s4 = 6, s5 = 7, s6 = 7, s7 = 7 (from Q1).

Suppose the desired specified histogram pz(z) is given as:

zq	pz(zq)
0	0.00
1	0.00
2	0.00
3	0.15
4	0.20
5	0.30
6	0.20
7	0.15

Computing G(zq) = 7 × Σ pz(zi):

zq	CDF	G(zq) = 7×CDF (rounded)
0	0.00	0
1	0.00	0
2	0.00	0
3	0.15	1
4	0.35	2
5	0.65	5

zq	CDF	G(zq) = 7×CDF (rounded)
6	0.85	6
7	1.00	7

For each equalized value sk, we now find the zq whose G(zq) is closest: e.g., for s0 = 1, the closest G(zq) is G(3) = 1, so r0 → z = 3. For s2 = 5, G(5) = 5, so r2 → z = 5. For s5 = 7, G(7) = 7, so r5 → z = 7. Repeating this search for every sk gives the complete mapping table from input gray level rk directly to output gray level zq, via s.

2.6 Comparison: Histogram Equalization vs Histogram Matching

Aspect	Histogram Equalization	Histogram Matching (Specification)
Output histogram	Approximately uniform	Matches any specified/desired shape
Control	No user control over result	User specifies the target histogram
Transformation	Single forward CDF mapping T(r)	Forward T(r) followed by inverse $G^{-1}(s)$
Use case	General automatic contrast enhancement	When a particular contrast/appearance is desired
Complexity	Lower — one CDF computation	Higher — requires an additional inverse mapping/search

2.7 Advantages

- Provides control over the shape of the output histogram, unlike plain equalization.
- Useful when a reference image's contrast/appearance needs to be matched.
- Can avoid over-enhancement of noise that uniform equalization sometimes causes.

2.8 Disadvantages / Limitations

- Requires a suitable target histogram to be known or specified in advance.
- Computationally more expensive due to the inverse mapping / iterative search step.
- Multiple input levels may still map to the same output level, so exact matching may not always be possible with discrete data.

2.9 Applications

- Matching the appearance of images taken by different sensors or under different lighting conditions.
- Normalizing a batch of images to a common reference histogram (e.g., in medical or satellite imaging).
- Producing specific contrast effects for visualization or further analysis.

2.10 Key Points to Remember

- Histogram equalization: $s_k = (L-1) \Sigma p_r(r_j)$ — automatic, produces near-uniform histogram.
- Histogram matching: forward-equalize input to get sk, then apply G^{-1} from the desired histogram to get zq.
- Histogram matching = Histogram Equalization + Inverse mapping to a specified target distribution.

3. Spatial Domain Filtering

3.1 Introduction

Spatial domain refers to the image plane itself, and spatial domain methods are procedures that operate directly on the pixels of an image, as opposed to frequency domain methods which operate on the Fourier (or other) transform of an image. Spatial domain processes are denoted by the expression:

$$g(x, y) = T[f(x, y)]$$

where f(x, y) is the input image, g(x, y) is the output (processed) image, and T is an operator on f, defined over a neighbourhood of point (x, y). When T operates on a single pixel (the neighbourhood is 1×1), the process is called an intensity or gray-level transformation. When T operates on a neighbourhood of pixels around (x, y), the process is called spatial filtering, and the neighbourhood together with a predefined operation is called a spatial filter (also referred to as a spatial mask, kernel, template, or window).

3.2 Mechanics of Spatial Filtering

A spatial filter consists of a neighbourhood (typically a small rectangular sub-image, such as 3×3) and a predefined operation performed on the image pixels encompassed by the neighbourhood. Filtering creates a new pixel with coordinates equal to those of the centre of the neighbourhood, whose value is the result of the filtering operation. If the operation is linear, the filter is called a linear spatial filter; otherwise, it is a nonlinear spatial filter.

For linear spatial filtering, the response is given by a sum of products between the filter (mask/kernel) coefficients and the corresponding image pixels in the area spanned by the filter mask. For an m×n mask (m = 2a+1, n = 2b+1), the response R at a point (x, y) is:

$$R = \Sigma(s=-a \text{ to } a)\ \Sigma(t=-b \text{ to } b)\ w(s, t)\ f(x+s, y+t)$$

where w(s, t) denotes the mask coefficients. This operation is known as correlation. When the mask is rotated by 180° before performing the same sum-of-products operation, the operation is called convolution. For a symmetric mask, correlation and convolution give the same result.

3.3 Classification of Spatial Filters

Category	Purpose	Examples
Smoothing (Lowpass) Filters	Blurring / noise reduction	Averaging (box) filter, Weighted average, Gaussian filter, Median filter
Sharpening (Highpass) Filters	Highlighting fine detail, edge enhancement	Laplacian filter, Unsharp masking, High-boost filtering, Gradient (Sobel, Prewitt) filters

3.4 Smoothing (Lowpass) Spatial Filters

Smoothing spatial filters are used for blurring and for noise reduction. Blurring is used in preprocessing steps, such as removal of small details from an image prior to object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplished by blurring with a linear filter, or by nonlinear filtering.

(a) Averaging (Box) Filter:

The output of a smoothing linear spatial filter is simply the average of the pixels contained in the neighbourhood of the filter mask. These filters are also called averaging filters or lowpass filters. A simple 3×3 averaging (box) filter has all coefficients equal to 1/9:

Box filter mask (3×3), all coefficients = 1/9
[1 1 1]
[1 1 1] × (1/9)
[1 1 1]

By replacing the value of every pixel with the average of the intensity levels in the neighbourhood defined by the filter mask, this process reduces sharp transitions in intensities, and the net effect is that sharp transitions (like noise and edges) are blurred, since abrupt changes are 'smoothed' by the averaging.

(b) Weighted Average Filter:

A weighted average gives more importance (weight) to some pixels at the expense of others, in order to reduce blurring in the smoothing process. A common example is a mask that weights the centre pixel and its immediate neighbours more heavily:

Weighted average mask (3×3), sum of weights = 16
[1 2 1]
[2 4 2] × (1/16)
[1 2 1]

(c) Order-Statistic (Median) Filter:

Order-statistic filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixel values contained in the image area encompassed by the filter, then replacing the value of the centre pixel with the value determined by the ranking result. The best-known filter in this category is the median filter, which replaces the value of a pixel with the median of the intensity values in the neighbourhood of that pixel:

$$\hat{f}(x, y) = \text{median}\{ g(s, t) : (s, t) \in Sxy \}$$

Median filters are quite effective for removing impulse noise (also called salt-and-pepper noise) while preserving edges better than a linear averaging filter of similar size, since the median is not as strongly affected by outlier (extreme) values as the mean is.

3.5 Sharpening (Highpass) Spatial Filters

The principal objective of sharpening is to highlight fine detail in an image, or to enhance detail that has been blurred. Since averaging is analogous to integration, it is logical to conclude that sharpening can be accomplished by spatial differentiation. Foundationally, the strength of the response of a derivative operator is proportional to the degree of intensity discontinuity of the image at the point at which the operator is applied — image differentiation enhances edges and other discontinuities (such as noise) and de-emphasizes areas with slowly varying intensities.

(a) First-Order Derivative (Gradient):

The first-order derivative of a one-dimensional function f(x) is defined as the difference:

$$\partial f/\partial x = f(x+1) - f(x)$$

The gradient of an image f at coordinates (x, y) is the vector:

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \partial f/\partial x \\ \partial f/\partial y \end{bmatrix}$$

with magnitude (often approximated by absolute values for computational efficiency):

$$M(x, y) = mag(\nabla f) = \sqrt{(g_x^2 + g_y^2)} \approx |g_x| + |g_y|$$

Well-known first-derivative-based masks used for computing gx and gy include the Roberts cross-gradient operators and the Sobel operators (which include additional weighting to provide smoothing, reducing sensitivity to noise). The Sobel masks are:

Sobel gx mask	*Sobel gy mask*
[-1 0 1]	[-1 -2 -1]
[-2 0 2]	[0 0 0]
[-1 0 1]	[1 2 1]

(b) Second-Order Derivative (Laplacian):

The second-order derivative of a one-dimensional function f(x) is defined as the difference:

$$\partial^2 f/\partial x^2 = f(x+1) + f(x-1) - 2f(x)$$

The Laplacian for a function (image) f(x, y) of two variables is defined as:

$$\nabla^2 f = \partial^2 f/\partial x^2 + \partial^2 f/\partial y^2$$
$$= [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y)$$

This equation can be implemented using the following filter mask, which gives an isotropic result for rotations in increments of 90°:

Laplacian mask (3×3)

0	1	0
1	-4	1
0	1	0

Because the Laplacian is a derivative operator, it highlights sharp intensity transitions and de-emphasizes regions of slowly varying intensity, tending to produce images with grayish edge lines and other discontinuities superimposed on a dark, featureless background. To restore the background while keeping the sharpening effect, the Laplacian image is added back to the original:

$$g(x, y) = f(x, y) + c[\nabla^2 f(x, y)]$$

where c = −1 if the centre coefficient of the Laplacian mask is negative, and c = 1 if it is positive.

(c) Unsharp Masking and High-Boost Filtering:

Unsharp masking is a sharpening technique used for many years in the publishing industry to sharpen images. It consists of subtracting a blurred (unsharp) version of an image from the original image itself:

$$f_s(x, y) = f(x, y) - \bar{f}(x, y)$$

where fs(x, y) is the sharpened image and $\bar{f}(x, y)$ is a blurred version of f(x, y). A slight generalization of unsharp masking is called high-boost filtering, where the original image is multiplied by an amplification factor $A \geq 1$ before subtracting the blurred image:

$$f_{hb}(x, y) = A \cdot f(x, y) - \bar{f}(x, y) = (A-1) f(x, y) + f_s(x, y)$$

High-boost filtering allows the contribution of the original image to be increased relative to the sharpening (mask) component. When A = 1, high-boost filtering reduces to regular unsharp masking; as A increases past 1, the contribution of the original image becomes progressively more dominant.

3.6 Combining Spatial Enhancement Methods

In many applications, image enhancement is done using a combination of several complementary methods, rather than relying on a single technique, to obtain the best possible visual result. A typical example combines Laplacian sharpening (to bring out fine detail), a Sobel gradient (smoothed and used as a mask to selectively sharpen high-detail areas while suppressing noise amplification), and gray-level transformations (like power-law transformation) to further increase the dynamic range of intensity levels — showing that these tools are frequently used together rather than in isolation.

3.7 Advantages of Spatial Domain Filtering

- Conceptually simple and intuitive — the operation is performed directly on pixel intensities.
- Computationally efficient for small masks, as it avoids the overhead of transforming to and from another domain (e.g., frequency domain).
- Flexible — a wide variety of linear and nonlinear operations (smoothing, sharpening, edge detection, noise removal) can all be implemented using the same neighbourhood-processing framework.

3.8 Disadvantages / Limitations

- Large filter masks can be computationally expensive compared to equivalent frequency-domain filtering for large images.
- Linear smoothing filters blur edges along with noise, since they cannot distinguish between the two.
- Choice of an inappropriate mask size or type can lead to over-smoothing, over-sharpening, or amplification of noise.

3.9 Applications

- Noise reduction and preprocessing before segmentation (smoothing filters).
- Edge detection and boundary extraction for object recognition (gradient and Laplacian filters).
- Enhancement of medical, satellite, and forensic images for better visual interpretation.
- Sharpening of blurred images in photography and printing (unsharp masking, high-boost filtering).

3.10 Key Points to Remember

- Spatial filtering = neighbourhood operation using a mask/kernel; response computed via correlation (or convolution for a rotated mask).
- Smoothing filters (averaging, Gaussian, median) reduce noise but blur edges; median filter is best for salt-and-pepper noise.
- Sharpening filters use derivatives: first-order (gradient/Sobel) for edge detection, second-order (Laplacian) for fine-detail enhancement.
- Unsharp masking / high-boost filtering sharpen images by subtracting a blurred version from (a scaled version of) the original.