Computer Vision

Lecture 5: Point Analysis

October 5, 2021

Prof. Sébastien Lefèvre
sebastien.lefevre@univ-ubs.fr

Preliminary notations

We will write a pixel $\mathrm{p} = (\mathrm{x}, \mathrm{y})$ with x and y its spatial coordinates.

We can write an image as a function $\mathrm{f} : \mathrm{E} \subset \mathbb{Z}^2 \to \mathrm{V} \subset \mathbb{Z}$
mapping each pixel p to its gray value v, i.e. $\mathrm{f}(\mathrm{p}) = \mathrm{f}(\mathrm{x}, \mathrm{y}) = \mathrm{v}$.
$\mathrm{E} = [1 \ldots \mathrm{H}] \times [1 \ldots \mathrm{W}]$ is the set of pixel coordinates,
$\mathrm{V} = [1 \ldots 255]$ the set of pixel values

Preliminary notations

We will write a pixel $\mathrm{p} = (\mathrm{x}, \mathrm{y})$ with x and y its spatial coordinates.

We can write an image as a function $\mathrm{f} : \mathrm{E} \subset \mathbb{Z}^2 \to \mathrm{V} \subset \mathbb{Z}$
mapping each pixel p to its gray value v, i.e. $\mathrm{f}(\mathrm{p}) = \mathrm{f}(\mathrm{x}, \mathrm{y}) = \mathrm{v}$.
$\mathrm{E} = [1 \dots \mathrm{H}] \times [1 \dots \mathrm{W}]$ is the set of pixel coordinates,
$\mathrm{V} = [1 \dots 255]$ the set of pixel values

A point-based operator g relies solely on the pixel value,
thus ignoring the neighbourhood information: $\mathrm{g}(\mathrm{f}(\mathrm{x}, \mathrm{y})) = \mathrm{g}(\mathrm{v})$.

Preliminary notations

We will write a pixel $\mathrm{p} = (\mathrm{x}, \mathrm{y})$ with x and y its spatial coordinates.

We can write an image as a function $\mathrm{f} : \mathrm{E} \subset \mathbb{Z}^2 \to \mathrm{V} \subset \mathbb{Z}$
mapping each pixel p to its gray value v, i.e. $\mathrm{f}(\mathrm{p}) = \mathrm{f}(\mathrm{x}, \mathrm{y}) = \mathrm{v}$.
$\mathrm{E} = [1 \dots \mathrm{H}] \times [1 \dots \mathrm{W}]$ is the set of pixel coordinates,
$\mathrm{V} = [1 \dots 255]$ the set of pixel values

A point-based operator g relies solely on the pixel value,
thus ignoring the neighbourhood information: $\mathrm{g}(\mathrm{f}(\mathrm{x}, \mathrm{y})) = \mathrm{g}(\mathrm{v})$.

Question

> Examples of point-based operators?

Single-image operations

The operator g can take multiple forms:

- a linear function $g(x) = ax + b$
- a logarithmic function $g(x) = c\log(1 + x)$
- a power-law function $g(x) = cx^{\gamma}$
- a piecewise linear function

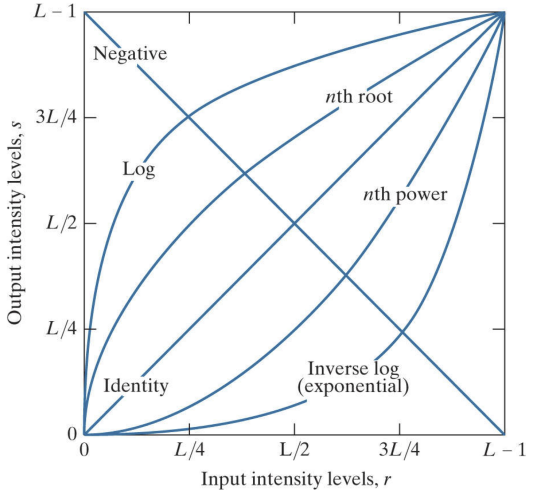


FIGURE 3.3
Some basic intensity transformation functions. Each curve was scaled *independently* so that all curves would fit in the same graph. Our interest here is on the *shapes* of the curves, not on their relative values.

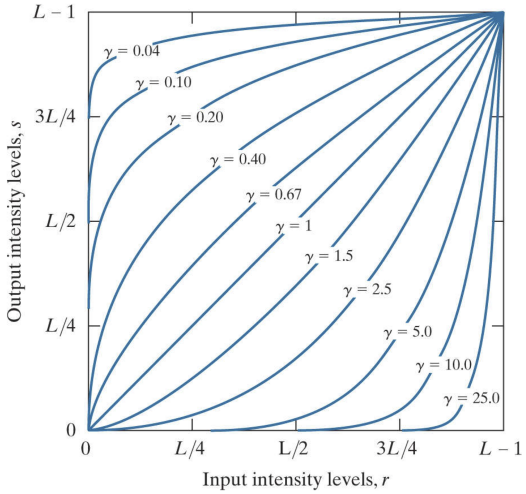


FIGURE 3.6
Plots of the gamma equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases). Each curve was scaled *independently* so that all curves would fit in the same graph. Our interest here is on the *shapes* of the curves, not on their relative values.

Linear mapping

The operator $g(x) = ax + b$ is parameterized by a and b.

Linear mapping

The operator $g(x) = ax + b$ is parameterized by a and b.

1. if $a = 1$, then we have $g(x) = x + b$.
   $b > 0 \Rightarrow$ intensity increased, brighter image.
   $b < 0 \Rightarrow$ intensity decreased, darkener image.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

Linear mapping

The operator $g(x) = ax + b$ is parameterized by a and b.

1. if $a = 1$, then we have $g(x) = x + b$.
   $b > 0 \Rightarrow$ intensity increased, brighter image.
   $b < 0 \Rightarrow$ intensity decreased, darkener image.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

2. if $b = 0$, then we have $g(x) = ax$.
   $a > 1 \Rightarrow$ intensity higher and wider range, higher contrast.
   $a < 1 \Rightarrow$ intensity lower and narrower range, lower contrast.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

Linear mapping

The operator $g(x) = ax + b$ is parameterized by a and b.

1. if $a = 1$, then we have $g(x) = x + b$.
   $b > 0 \Rightarrow$ intensity increased, brighter image.
   $b < 0 \Rightarrow$ intensity decreased, darkener image.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

2. if $b = 0$, then we have $g(x) = ax$.
   $a > 1 \Rightarrow$ intensity higher and wider range, higher contrast.
   $a < 1 \Rightarrow$ intensity lower and narrower range, lower contrast.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

Adjusting contrast without adjusting brightness is possible.
it requires to operate on the central point c (e.g. 127):
$g(x) = a(x - c) + c$

We still need to clip values outside $[0, 255]$.

Linear mapping

The operator $g(x) = ax + b$ is parameterized by a and b.

1. if $a = 1$, then we have $g(x) = x + b$.
   $b > 0 \Rightarrow$ intensity increased, brighter image.
   $b < 0 \Rightarrow$ intensity decreased, darkener image.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

2. if $b = 0$, then we have $g(x) = ax$.
   $a > 1 \Rightarrow$ intensity higher and wider range, higher contrast.
   $a < 1 \Rightarrow$ intensity lower and narrower range, lower contrast.
   Caution: $g(x)$ should remain in $[0 \ldots 255]$.

3. generally, both a and b are set, with among popular examples

- the negative operator defined by $a = -1$ and $b = 255$
- a adjusts (a.k.a. gain factor) the contrast,
  while b (a.k.a. offset) adjusts the brightness.

Remark: if $V = [0, 1]$, then usually $b \in [-1, 1]$.

Logarithmic mapping

These mappings are useful for more advanced contrast stretching.

- Logarithmic mapping $g(x) = b \ln(ax + 1) + c$, with

  $a > 0$ controls the curvature of the logarithmic function,
  b scales the output value to fall within a given range,
  the shift 1 avoids the zero value,
  and c is a global shift constant.

  It spreads low values and compresses high values,
  enhancing details in dark areas while removing those in light areas.

Logarithmic mapping

These mappings are useful for more advanced contrast stretching.

- Logarithmic mapping $g(x) = b \ln(ax + 1) + c$, with

  $a > 0$ controls the curvature of the logarithmic function,
  b scales the output value to fall within a given range,
  the shift 1 avoids the zero value,
  and c is a global shift constant.

  It spreads low values and compresses high values,
  enhancing details in dark areas while removing those in light areas.

- Inverse logarithmic (i.e. exponential) mapping $g(x) = be^{ax+1} + c$,
  whose parameters have the same effect that with the logarithmic mapping.

  It spreads high values and compresses low values,
  enhancing details in light areas while removing those in dark areas.

Power-law mapping

The general form is $g(x) = c(x + b)^{\gamma}$, with c and γ positive, and usually $b = 0$.
$\gamma > 1$ narrows the range of dark values while widening the range of bright values.
$\gamma < 1$ narrows the range of bright values while widening the range of dark values.

Gamma correction with $\gamma \in [1.8, 2.5]$ is often embedded in monitors.

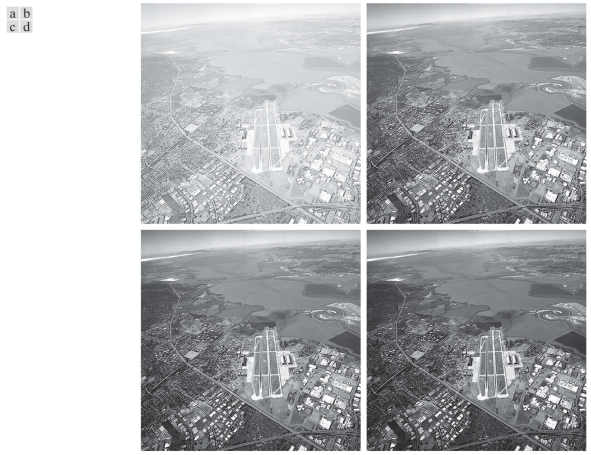


FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3-5) with $\gamma = 3.0$, 4.0, and 5.0, respectively. ($c = 1$ in all cases.) (Original image courtesy of NASA.)

Power-law mapping

The general form is $g(x) = c(x + b)^{\gamma}$, with c and γ positive, and usually $b = 0$.
$\gamma > 1$ narrows the range of dark values while widening the range of bright values.
$\gamma < 1$ narrows the range of bright values while widening the range of dark values.

Gamma correction with $\gamma \in [1.8, 2.5]$ is often embedded in monitors.

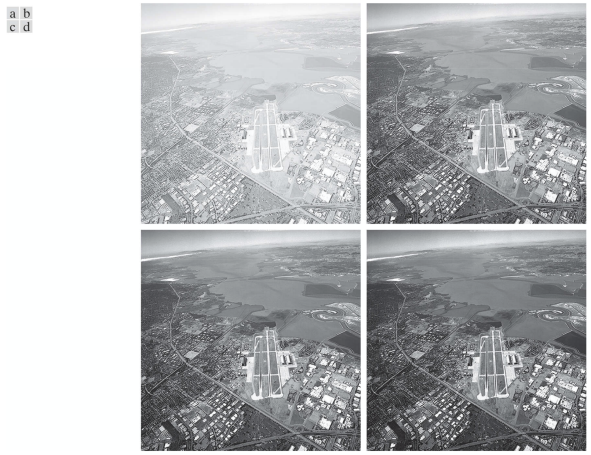


FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3-5) with $\gamma = 3.0$, 4.0, and 5.0, respectively. ($c = 1$ in all cases.) (Original image courtesy of NASA.)

Remark: For any mapping g, contrast is stretched when its derivative $g'(x) > 1$, shrinked when $g'(x) < 1$, and inverted when $g'(x) < 0$.

Piecewise linear enhancement

Piecewise linear functions allow for more complex intensity mappings.

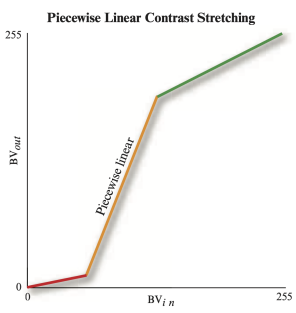
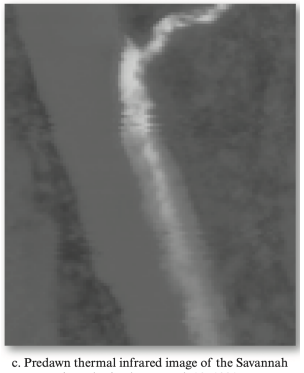
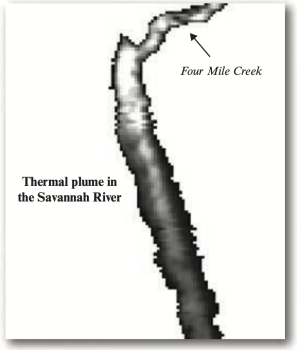


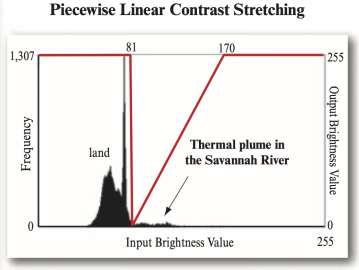
FIGURE 8–12 Logic of a piecewise linear contrast stretch for which three selective pieces of the histogram are linearly contrast stretched. Note that the slope of the linear contrast enhancement changes.



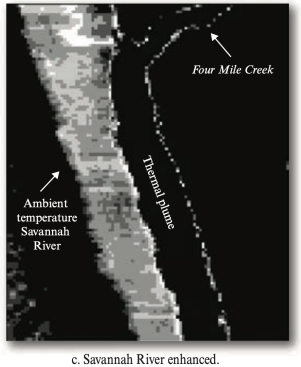
c. Predawn thermal infrared image of the Savannah River obtained on March 28, 1981.



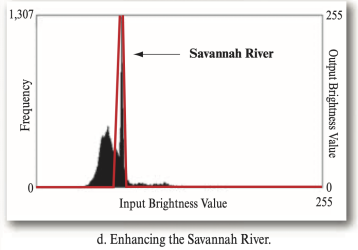
a. Thermal plume enhanced.



b. Enhancing the thermal plume in the Savannah River.



c. Savannah River enhanced.



d. Enhancing the Savannah River.

FIGURE 8–13 a,b) Thermal infrared imagery of the Savannah River enhanced using piecewise linear contrast stretching to highlight the thermal plume at the expense of the ambient river wa ter and the surrounding landscape. c,d) Piecewise linear contrast stretching to enhance the Savannah River at the expense of the thermal plume and the surrounding landscape.

They are often used together with the histogram of the image.

Examples of PL functions

Solarization:

$$g(x) = \begin{cases} 255 - x & \text{if } x < 128 \\ x & \text{otherwise} \end{cases}$$

Examples of PL functions

Solarization:

$$g(x) = \begin{cases} 255 - x & \text{if } x < 128 \\ x & \text{otherwise} \end{cases}$$

Intensity-level slicing (with $v = x$ or $v = 0$):

$$g(x) = \begin{cases} 255 & \text{if } T_1 \leq x \leq T_2 \\ v & \text{otherwise} \end{cases}$$

Binarization:

$$g(x) = \begin{cases} 0 & \text{if } x < T \\ 255 & \text{otherwise} \end{cases}$$

Image histogram

The (gray level) histogram of an image corresponds to the probability distribution function of the gray levels in the image.

In the discrete case, the histograms counts the number of occurrences of each gray level in the image.

Probabilities can be obtained by dividing each bin value by the number of pixels.

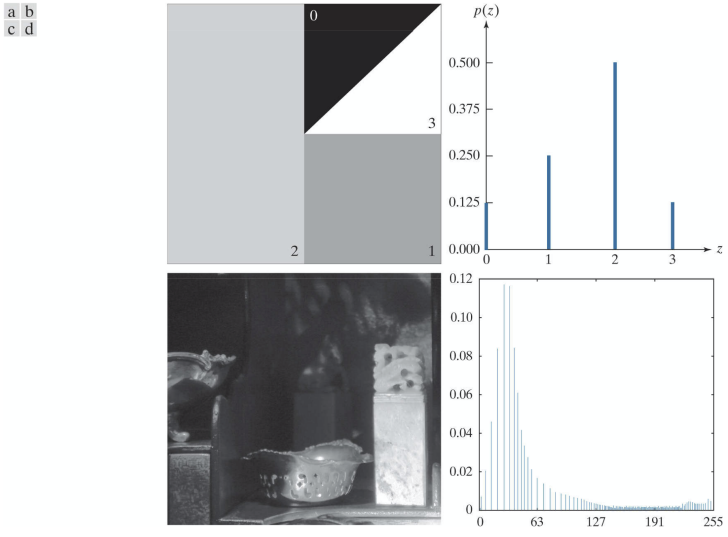


FIGURE 2.49
(a) A synthetic image, and (b) its histogram. (c) A natural image, and (d) its histogram.

Figures 2.49(c) shows a more complex image, and Fig. 2.49 (d) is its histogram. Here, we are interested in what the shape of the histogram can tell us about the general appearance of the image. For instance, the shape of the histogram is biased toward the dark end of the intensity scale, indicating that the image has a dark appearance. Note also that the histogram values span the entire intensity scale; this indicates that the image has reasonable contrast. As you will see next, histograms can be used also for computing numerical descriptors that further quantify image properties based on intensity.

Image histogram

The histogram is a very useful information to understand the image content.
And is used in many image processing and analysis tasks.

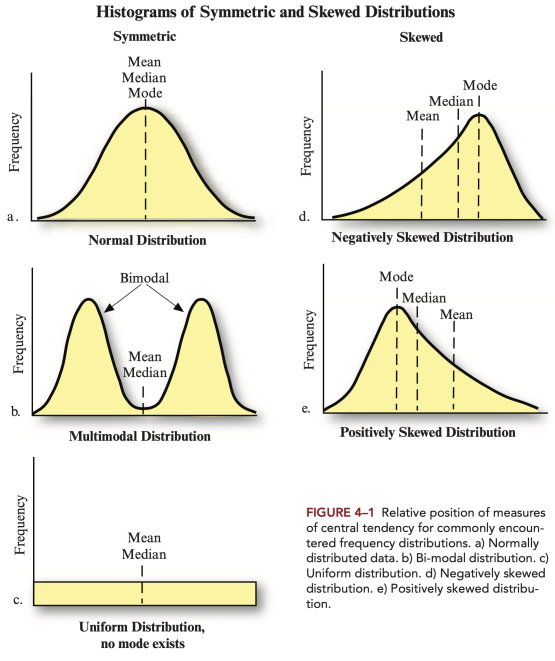


FIGURE 4–1 Relative position of measures of central tendency for commonly encountered frequency distributions. a) Normally distributed data. b) Bi-modal distribution. c) Uniform distribution. d) Negatively skewed distribution. e) Positively skewed distribution.

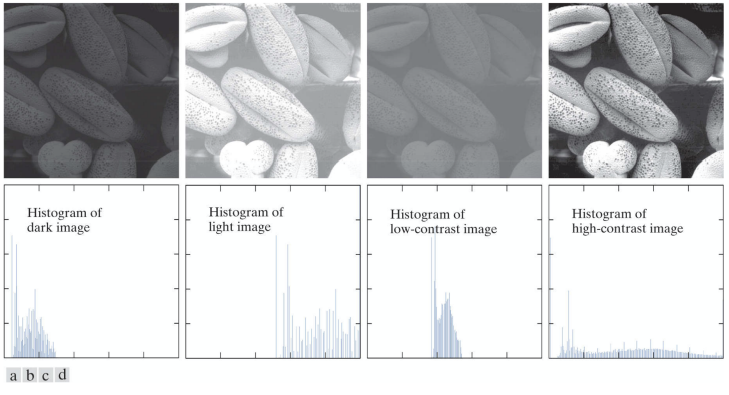


FIGURE 3.16
Four image types and their corresponding histograms. (a) dark; (b) light; (c) low contrast; (d) high contrast. The horizontal axis of the histograms are values of r_k and the vertical axis are values of $p(r_k)$.

Introductory Digital Image Processing: A Remote Sensing Perspective; Jensen; Pearson

Digital image processing; Gonzalez, Woods; Pearson

Computing an histogram

A naive approach would be:

```
hist=np.zeros(256,dtype=int)
values=input.flatten()
for v in range(len(hist)):
  for i in values:
    if i==v:
      hist[v]+=1
```

Why we should avoid to do so?

Computing an histogram

A naive approach would be:

```
hist=np.zeros(256,dtype=int)
values=input.flatten()
for v in range(len(hist)):
  for i in values:
    if i==v:
      hist[v]+=1
```

Why we should avoid to do so?

Its computational complexity is $\mathcal{O}(\mathrm{V} \times \mathrm{P})$
with V the number of values (radiometric resolution),
and P the size of the image (number of pixels).

Computing an histogram

A naive approach would be:

```
hist=np.zeros(256,dtype=int)
values=input.flatten()
for v in range(len(hist)):
  for i in values:
    if i==v:
      hist[v]+=1
```

Why we should avoid to do so?

Its computational complexity is $\mathcal{O}(\mathrm{V} \times \mathrm{P})$
with V the number of values (radiometric resolution),
and P the size of the image (number of pixels).

A more efficient approach is

```
hist=np.zeros(256,dtype=int)
values=input.flatten()
for i in values:
  hist[i]+=1
```

Computing an histogram

Python is a script (a.k.a. interpreted) language (not a compiled one).
The previous code may remain inefficient due to the loop over all pixels.

It is better to use predefined Python functions:

```
hist=numpy.histogram(input,256,range=(0,255))[0]
# histogram return both counts and bin edges, we keep only the first
```

Note the flattening operation performed by default.

Computing an histogram

Python is a script (a.k.a. interpreted) language (not a compiled one).
The previous code may remain inefficient due to the loop over all pixels.

It is better to use predefined Python functions:

```
hist=numpy.histogram(input,256,range=(0,255))[0]
# histogram return both counts and bin edges, we keep only the first
```

Note the flattening operation performed by default.

The histogram can then be displayed:

```
import matplotlib.pyplot as plt
plt.plot(hist)
plt.show(hist)
```

Computing an histogram

Python is a script (a.k.a. interpreted) language (not a compiled one).
The previous code may remain inefficient due to the loop over all pixels.

It is better to use predefined Python functions:

```
hist=numpy.histogram(input,256,range=(0,255))[0]
# histogram return both counts and bin edges, we keep only the first
```

Note the flattening operation performed by default.

The histogram can then be displayed:

```
import matplotlib.pyplot as plt
plt.plot(hist)
plt.show(hist)
```

or with a single function:

```
plt.hist(im.flatten(),bins=256,range=[0,255])
```

Automatic contrast stretching

It is not always easy to set a and b in $g(x) = ax + b$.

These parameters can be defined automatically to use the full range of values, and to have a mapping $[min, max] \mapsto [0, 255]$.

Automatic contrast stretching

It is not always easy to set a and b in $g(x) = ax + b$.

These parameters can be defined automatically to use the full range of values, and to have a mapping $[min, max] \mapsto [0, 255]$.

We shift the reference $min \mapsto 0$ through $x \mapsto x - min$.

We stretch the dynamics $(max - min) \mapsto 255$ with $\times \frac{255}{max - min}$.

Automatic contrast stretching

It is not always easy to set a and b in $g(x) = ax + b$.

These parameters can be defined automatically to use the full range of values, and to have a mapping $[min, max] \mapsto [0, 255]$.

We shift the reference $min \mapsto 0$ through $x \mapsto x - min$.

We stretch the dynamics $(max - min) \mapsto 255$ with $\times \frac{255}{max - min}$.

So the automatic contrast stretching is defined by

$$g(x) = \frac{255(x - min)}{max - min}$$

Automatic contrast stretching

It is not always easy to set a and b in $g(x) = ax + b$.

These parameters can be defined automatically to use the full range of values, and to have a mapping $[min, max] \mapsto [0, 255]$.

We shift the reference $min \mapsto 0$ through $x \mapsto x - min$.

We stretch the dynamics $(max - min) \mapsto 255$ with $\times \frac{255}{max - min}$.

So the automatic contrast stretching is defined by

$$g(x) = \frac{255(x - min)}{max - min}$$

min and max computation have complexity of $\mathcal{O}(P)$,
but note that min and max are first and last non-zero values of the histogram, leading to a lower complexity $\mathcal{O}(V)$ and usually requiring only a few operations:

```
nz=np.nonzero(hist)[0]
min,max=nz[0],nz[-1]
```

Percentile contrast stretching

The automatic contrast stretching (a.k.a. dynamic expansion) only focuses on $\min$ and $\max$, but these two extrema may may not be representative of the actual dynamics (e.g. in noisy images).

In this case, it is useful to ignore the α% extrema when computing $\min$ and $\max$, and clipping them to 0 and 255.

This can be (again) efficiently implemented with the histogram, or even better the cumulative histogram, i.e. the cumulative distribution function of gray levels.

In case of a normal distribution, standard deviation contrast stretching selects $\min$ and $\max$ as $\mu \pm \alpha\sigma$ (e.g. $\alpha = 1$).

Cumulative histogram

An histogram measures an image pdf (probability density function),
i.e. the probability that a pixel value equals a given value v.

A cumulative histogram represents the image cdf (cumulative density function),
i.e. the probability that a pixel values is less or equal to v.

Cumulative histogram

An histogram measures an image pdf (probability density function),
i.e. the probability that a pixel value equals a given value v.

A cumulative histogram represents the image cdf (cumulative density function),
i.e. the probability that a pixel values is less or equal to v.

`chist` can be easily computed from `hist` (previously normalized)

```
chist=hist
for i in range(1,len(hist)):
  chist[i]=chist[i-1]+hist[i]
```

α-min is obtained by looking at the first element from `chist` s.t. `chist[i]`$>\alpha$.

α-max is obtained by looking at the first element (in the reverse order)
from `chist` s.t. `chist[i]`$<1-\alpha$.

Histogram equalization

Histogram equalization aims to flat the histogram of the image (aiming to a uniform distribution).

$g(x)$ is simply `chist(x) * 255`!

a b c

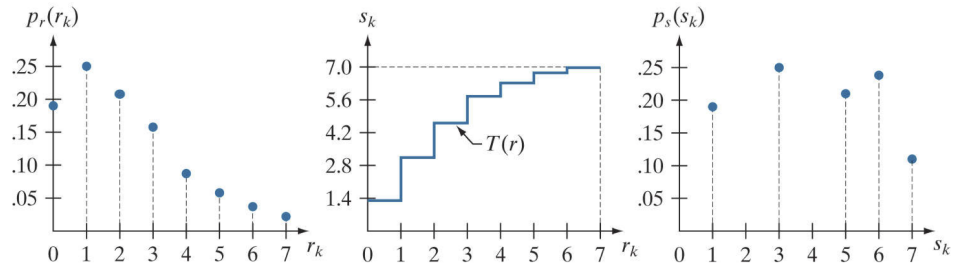


FIGURE 3.19
Histogram equalization. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

Histogram equalization

Histogram equalization aims to flat the histogram of the image (aiming to a uniform distribution).

$g(x)$ is simply `chist(x) * 255`!

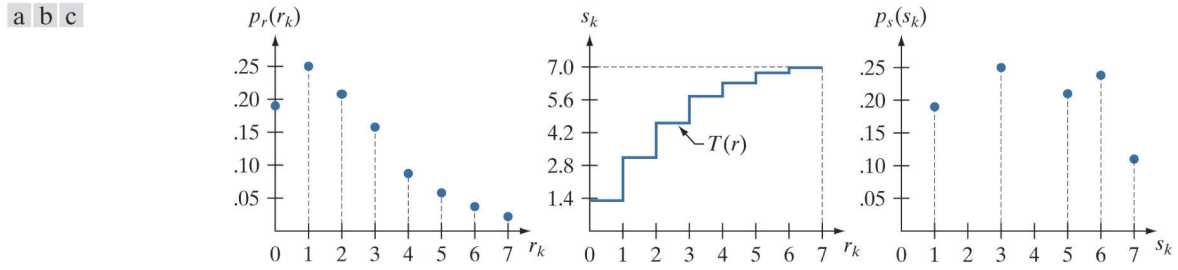


FIGURE 3.19
Histogram equalization. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

Aiming at a uniform distribution usually brings too much contrast.
It is possible to target a more specific distribution, e.g. a Gaussian distribution.
This more generic process is called histogram specification.

Efficient implementation with LUT

All point-based operators do not rely on pixels coordinates.
Since usually $\mathrm{V} \ll \mathrm{P}$, numerous computations are unnecessarily repeated (for pixels sharing the same value).

The computational cost can be greatly reduced by precomputing the mappings $\mathbf{v} \mapsto \mathrm{g}(\mathbf{v}) \quad \forall \mathbf{v}$.

These mappings are stored in a so-called Look-Up Table (LUT) of size V, where the cell $\mathbf{v}$ contains $\mathrm{g}(\mathbf{v})$.

This is particularly useful is the function g is complex (e.g. requiring floating point operations).

Working with multiple images

Various arithmetic operators can be involved:

- image addition (and average)
- image substraction (differencing)
- image multiplication
- image division (ratio)

Working with multiple images

Various arithmetic operators can be involved:

- image addition (and average)
- image substraction (differencing)
- image multiplication
- image division (ratio)

The images can either be

1. the bands of a single multiband image: band rationing,
2. or different images.

Working with multiple images

Various arithmetic operators can be involved:

- image addition (and average)
- image substraction (differencing)
- image multiplication
- image division (ratio)

The images can either be

1. the bands of a single multiband image: band rationing,
2. or different images.

Discussion:

popular indices based on band rationing?
possible applications of multi-images operations in remote sensing?

Examples of multi-image operators

- Image averaging
  - reduce noise
  - map a multispectral image into a gray scale one
  - summarize a satellite image time series in a single image

Examples of multi-image operators

- Image averaging
  - reduce noise
  - map a multispectral image into a gray scale one
  - summarize a satellite image time series in a single image
- Image differencing
  - band difference
  - change detection
  - background (noise) removal

Examples of multi-image operators

- Image averaging
  - reduce noise
  - map a multispectral image into a gray scale one
  - summarize a satellite image time series in a single image
- Image differencing
  - band difference
  - change detection
  - background (noise) removal
- Image multiplication
  - masking

Examples of multi-image operators

- Image averaging
  - reduce noise
  - map a multispectral image into a gray scale one
  - summarize a satellite image time series in a single image
- Image differencing
  - band difference
  - change detection
  - background (noise) removal
- Image multiplication
  - masking
- Image division
  - band ratio

Questions?

Labs

Steps:

1. Load an image.
2. Compute its negative.
3. Without & with a LUT, apply a gamma correction (report the efficiency gain).
4. Code the histogram equalization function.
5. Check if `histeq(negative(image))==negative(histeq(image))`.
6. Perform a contrast stretch clipping the 5% extrema.
7. Play with multi-image operators (e.g. NDVI extraction, change detection, RGB to gray)