Digital Image Processing (CS/ECE 545) 
Lecture 2: Histograms and Point Operations (Part 1) 

Prof Emmanuel Agu 

Computer Science Dept. 
Worcester Polytechnic Institute (WPI)
Histograms

- Histograms plots how many times (frequency) each intensity value in image occurs

- Example:
  - Image (left) has 256 distinct gray levels (8 bits)
  - Histogram (right) shows frequency (how many times) each gray level occurs
Histograms

- Many cameras display real time histograms of scene
- Helps avoid taking over-exposed pictures
- Also easier to detect types of processing previously applied to image
**Histograms**

- E.g. $K = 16$, 10 pixels have intensity value $i = 2$
- Histograms: only statistical information
- No indication of **location** of pixels
Histograms

- Different images can have same histogram
- 3 images below have same histogram

![Images showing different distributions](image)

- Half of pixels are gray, half are white
  - Same histogram = same statistics
  - Distribution of intensities could be different

- Can we reconstruct image from histogram? No!
Histograms

- So, a histogram for a grayscale image with intensity values in range
  \[ I(u, v) \in [0, K - 1] \]
  would contain exactly \( K \) entries

- E.g. 8-bit grayscale image, \( K = 2^8 = 256 \)

- Each histogram entry is defined as:
  \[ h(i) = \text{number of pixels with intensity } I \text{ for all } 0 < i < K. \]

- E.g: \( h(255) = \) number of pixels with intensity = 255

- Formal definition
  \[ h(i) = \text{card}\{(u, v) \mid I(u, v) = i\} \]

  Number (size of set) of pixels such that
Interpreting Histograms

- Log scale makes low values more visible

Difference between darkest and lightest

![Histogram Diagram](image)
Histograms

- Histograms help detect image acquisition issues
- Problems with image can be identified on histogram
  - Over and under exposure
  - Brightness
  - Contrast
  - Dynamic Range
- Point operations can be used to alter histogram. E.g
  - Addition
  - Multiplication
  - Exp and Log
  - Intensity Windowing (Contrast Modification)
Image Brightness

- Brightness of a grayscale image is the **average intensity** of all pixels in image

\[
B(I) = \frac{1}{wh} \sum_{v=1}^{h} \sum_{u=1}^{w} I(u, v)
\]

1. Sum up all pixel intensities
2. Divide by total number of pixels
Detecting Bad Exposure using Histograms

Exposure? Are intensity values spread (good) out or bunched up (bad)

(a) Underexposed
(b) Properly Exposed
(c) Overexposed
Image Contrast

- The contrast of a grayscale image indicates how easily objects in the image can be distinguished
- **High contrast image**: many distinct intensity values
- **Low contrast**: image uses few intensity values
Histograms and Contrast

**Good Contrast?** Widely spread intensity values
+ large difference between min and max intensity values

Image

Histogram

(a) Low contrast
(b) Normal contrast
(c) High contrast
Contrast Equation?

- Many different equations for contrast exist
- Examples:

\[
\text{Contrast} = \frac{\text{Change in Luminance}}{\text{Average Luminance}}
\]

- Michalson’s equation for contrast

\[
C_M(I) = \frac{\max(I) - \min(I)}{\max(I) + \min(I)}
\]
Contrast Equation?

- These equations work well for simple images with 2 luminances (i.e. uniform foreground and background)
- Does not work well for complex scenes with many luminances or if min and max intensities are small
**Histograms and Dynamic Range**

- **Dynamic Range**: Number of distinct pixels in image

- Difficult to increase image dynamic range (e.g. interpolation)

- HDR (12-14 bits) capture typical, then down-sample
High Dynamic Range Imaging

- **High dynamic range** means very bright and very dark parts in a single image (many distinct values)
- Dynamic range in photographed scene may exceed number of available bits to represent pixels
- Solution:
  - Capture multiple images at different exposures
  - Combine them using image processing
Detecting Image Defects using Histograms

- No “best” histogram shape, depends on application
- Image defects
  - **Saturation**: scene illumination values outside the sensor’s range are set to its min or max values => results in spike at ends of histogram
  - **Spikes and Gaps in manipulated images** (not original). Why?
Image Defects: Effect of Image Compression

- Histograms show impact of image compression
- Example: in GIF compression, dynamic range is reduced to only few intensities (quantization)

Original Image

(a) Original Histogram

(b) Histogram after GIF conversion

Fix? Scaling image by 50% and Interpolating values recreates some lost colors

But GIF artifacts still visible
Effect of Image Compression

- Example: Effect of JPEG compression on line graphics
- JPEG compression designed for color images

Original histogram has only 2 intensities (gray and white)

JPEG image appears dirty, fuzzy and blurred

Its Histogram contains gray values not in original
Computing Histograms

```java
public class Compute_Histogram implements PlugInFilter {

    public int setup(String arg, ImagePlus img) {
        return DOES_8G + NO_CHANGES;
    }

    public void run(ImageProcessor ip) {
        int[] H = new int[256]; // histogram array
        int w = ip.getWidth();
        int h = ip.getHeight();

        for (int v = 0; v < h; v++) {
            for (int u = 0; u < w; u++) {
                int i = ip.getPixel(u, v);
                H[i] = H[i] + 1;
            }
        }

        ... // histogram H[] can now be used
    }

} // end of class Compute_Histogram
```
ImageJ Histogram Function

- ImageJ has a histogram function (`getHistogram()`)
- Prior program can be simplified if we use it

```java
public void run(ImageProcessor ip) {
    int[] H = ip.getHistogram();
    // ... // histogram H[] can now be used
}
```

Returns histogram as an array of integers
Large Histograms: Binning

- High resolution image can yield very large histogram
- Example: 32-bit image = $2^{32} = 4,294,967,296$ columns
- Such a large histogram impractical to display
- Solution? Binning!
  - Combine *ranges of intensity values* into histogram columns

So, given the image $I : \Omega \rightarrow [0, K - 1]$, the binned histogram for $I$ is the function

$$h(i) = \text{card}\{(u, v) | a_i \leq I(u, v) < a_{i+1}\},$$

where $0 = a_0 < a_1 < \ldots < a_B = K$. 

Number (size of set) of pixels such that Pixel’s intensity is between $a_i$ and $a_{i+1}$
Calculating Bin Size

- Typically use equal sized bins
- Bin size?
  \[
  \text{Bin size} = \frac{\text{Number of distinct values in image}}{\text{Number of bins}}
  \]
- Example: To create 256 bins from 14-bit image

\[
\text{Bin size} = \frac{2^{14}}{256} = 64
\]

\[
\begin{align*}
  h(0) & \leftarrow 0 \leq I(u, v) < 64 \\
  h(1) & \leftarrow 64 \leq I(u, v) < 128 \\
  h(2) & \leftarrow 128 \leq I(u, v) < 192 \\
  \vdots & \quad \vdots & \quad \vdots \\
  h(j) & \leftarrow a_j \leq I(u, v) < a_{j+1} \\
  \vdots & \quad \vdots & \quad \vdots \\
  h(255) & \leftarrow 16320 \leq I(u, v) < 16384
\end{align*}
\]
Binned Histogram

- To calculate which bin a pixel’s intensity belongs to

\[ \frac{I(u,v)}{k_B} = \frac{I(u,v)}{K/B} = I(u,v) \cdot \frac{B}{K} \]

- Previous example, \( B = 256, \ K = 2^{14} = 16384 \)

```java
int[] binnedHistogram(ImageProcessor ip) {
    int K = 256; // number of intensity values
    int B = 32; // size of histogram, must be defined
    int[] H = new int[B]; // histogram array
    int w = ip.getWidth();
    int h = ip.getHeight();

    for (int v = 0; v < h; v++) {
        for (int u = 0; u < w; u++) {
            int a = ip.getPixel(u, v);
            int i = a * B / K; // integer operations only!
            H[i] = H[i] + 1;
        }
    }

    // return binned histogram
    return H;
}
```

- Create array to store histogram computed
- Calculate which bin to add pixel’s intensity
- Increment corresponding histogram
Color Image Histograms

Two types:

1. **Intensity histogram:**
   - Convert color image to gray scale
   - Display histogram of gray scale

2. **Individual Color Channel Histograms:**
   - 3 histograms (R,G,B)
Color Image Histograms

- Both types of histograms provide useful information about lighting, contrast, dynamic range and saturation effects
- No information about the actual color distribution!
- Images with totally different RGB colors can have same R, G and B histograms
- Solution to this ambiguity is the Combined Color Histogram.
  - More on this later
Cumulative Histogram

- Useful for certain operations (e.g. histogram equalization) later
- Analogous to the **Cumulative Density Function (CDF)**
- Definition:

  \[ H(i) = \sum_{j=0}^{i} h(j) \quad \text{for} \quad 0 \leq i < K \]

- Recursive definition

  \[ H(i) = \begin{cases} h(0) & \text{for } i = 0 \\ H(i-1) + h(i) & \text{for } 0 < i < K \end{cases} \]

- Monotonically increasing

  \[ H(K-1) = \sum_{j=0}^{K-1} h(j) = M \cdot N \]
Point Operations

- Point operations change a pixel’s intensity value according to some function (don’t care about pixel’s neighbor)
  \[ a' \leftarrow f(a) \]
  \[ I'(u, v) \leftarrow f(I(u, v)) \]

- Also called a **homogeneous operation**

- New pixel intensity depends on
  - Pixel’s previous intensity \( I(u, v) \)
  - Mapping function \( f( ) \)

- Does not depend on
  - Pixel’s location \( (u, v) \)
  - Intensities of neighboring pixels
Some Homogeneous Point Operations

- **Addition (Changes brightness)**
  \[ f(p) = p + k \]
  E.g. \[ f_{\text{bright}}(p) = p + 10 \]

- **Multiplication (Stretches/shrinks image contrast range)**
  \[ f(p) = k \times p \]
  E.g. \[ f_{\text{contrast}}(p) = p \times 1.5 \]

- **Real-valued functions**
  \[ \exp(x), \log(x), (1/x), x^k, \text{etc.} \]

- **Quantizing pixel values**
- **Global thresholding**
- **Gamma correction**
Point Operation Pseudocode

- **Input:** Image with pixel intensities $I(u,v)$ defined on $[1 .. w] \times [1 .. H]$
- **Output:** Image with pixel intensities $I'(u,v)$

```plaintext
for v = 1 .. h
  for u = 1 .. w
    set $I(u, v) = f(I(u, v))$
```
Non-Homogeneous Point Operation

- New pixel value depends on:
  - Old value + pixel’s location \((u, v)\)

- \[ a' \leftarrow g(a, u, v) \]
- \[ I'(u, v) \leftarrow g(I(u, v), u, v) \]
Clamping

- Deals with pixel values outside displayable range
  - If \( a > 255 \), \( a = 255 \);
  - If \( a < 0 \), \( a = 0 \);
- Function below will **clamp** (force) all values to fall within range \([a,b]\)

\[
f(p) = \begin{cases} 
  a & \text{if } p < a \\
  p & \text{if } a \leq p \leq b \\
  b & \text{if } p > b 
\end{cases}
\]
Example: Modify Intensity and Clamp

- Point operation: increase image contrast by 50% then clamp values above 255

```java
1  public void run(ImageProcessor ip) {
2      int w = ip.getWidth();
3      int h = ip.getHeight();
4
5      for (int v = 0; v < h; v++) {
6          for (int u = 0; u < w; u++) {
7              int a = (int) (ip.get(u, v) * 1.5 + 0.5);
8              if (a > 255)
9                  a = 255;  // clamp to maximum value
10              ip.set(u, v, a);
11          }
12      }
13  }
```

Increase contrast by 50%
Inverting Images

\[ f_{\text{invert}}(a) = -a + a_{\text{max}} = a_{\text{max}} - a \]

- 2 steps
  1. Multiple intensity by -1
  2. Add constant (e.g. \( a_{\text{max}} \)) to put result in range \([0, a_{\text{max}}]\)

- Implemented as ImageJ method `invert()`

Original Image

Inverted Image
Image Negatives (Inverted Images)

- Image negatives useful for enhancing white or grey detail embedded in dark regions of an image
  - Note how much clearer the tissue is in the negative image of the mammogram below

Thresholding

- Input values below **threshold** \( a_{th} \) set to \( a_0 \)
- Input values above **threshold** \( a_{th} \) set to \( a_1 \)

\[
f_{\text{threshold}}(a) = \begin{cases} a_0 & \text{for } a < a_{th} \\ a_1 & \text{for } a \geq a_{th} \end{cases}
\]

- Converts grayscale image to binary image (binarization) if
  - \( a_0 = 0 \)
  - \( a_1 = 1 \)

- Implemented as imageJ method **threshold**()
Thresholding Example

Original Image

Thresholded Image
Thresholding and Histograms

- Example with $a_{th} = 128$

- Thresholding splits histogram, merges halves into $a_0 a_1$
Basic Grey Level Transformations

- 3 most common gray level transformation:
  - Linear
    - Negative/Identity
  - Logarithmic
    - Log/Inverse log
  - Power law
    - $n^{th}$ power/$n^{th}$ root
Logarithmic Transformations

- Maps narrow range of input levels => wider range of output values
- Inverse log transformation does opposite transformation
- The general form of the log transformation is
  \[ s = c \cdot \log(1 + r) \]
- Log transformation of Fourier transform shows more detail
Power Law Transformations

- Power law transformations have the form
  
  \[ S = c \cdot r^\gamma \]

- Map narrow range of dark input values into wider range of output values or vice versa
- Varying \( \gamma \) gives a whole family of curves
Power Law Example

- Magnetic Resonance (MR) image of fractured human spine
- Different power values highlight different details

Intensity Windowing

- A clamp operation, then linearly stretching image intensities to fill possible range
- To window an image in \([a,b]\) with max intensity \(M\)

\[
f(p) = \begin{cases} 
0 & \text{if } p < a \\
M \times \frac{p-a}{b-a} & \text{if } a \leq p \leq b \\
M & \text{if } p > b
\end{cases}
\]
Intensity Windowing Example

Original Image

Windowed Image

Contrasts easier to see
Point Operations and Histograms

- Effect of some point operations easier to observe on histograms
  - Increasing brightness
  - Raising contrast
  - Inverting image

- Point operations only shift, merge histogram entries
- Operations that merge histogram bins are irreversible

Combining histogram operation easier to observe on histogram
Automatic Contrast Adjustment

- Point operation that modifies pixel intensities such that available range of values is fully covered
- Algorithm:
  - Find high and lowest pixel intensities $a_{\text{low}}, a_{\text{high}}$
  - Linear stretching of intensity range

\[
f_{\text{ac}}(a) = a_{\text{min}} + (a - a_{\text{low}}) \cdot \frac{a_{\text{max}} - a_{\text{min}}}{a_{\text{high}} - a_{\text{low}}}
\]

If $a_{\text{min}} = 0$ and $a_{\text{max}} = 255$

\[
f_{\text{ac}}(a) = (a - a_{\text{low}}) \cdot \frac{255}{a_{\text{high}} - a_{\text{low}}}
\]
Effects of Automatic Contrast Adjustment

Original Result of automatic Contrast Adjustment

Linearly stretching range causes gaps in histogram

Original

Result of automatic Contrast Adjustment
Modified Contrast Adjustment

- Better to map only certain range of values
- Get rid of tails (usually noise) based on predefined percentiles \((s_{\text{low}}, s_{\text{high}})\)

\[
\hat{a}_{\text{low}} = \min\{i \mid H(i) \geq M \cdot N \cdot s_{\text{low}}\}.
\]

\[
\hat{a}_{\text{high}} = \max\{i \mid H(i) \leq M \cdot N \cdot (1 - s_{\text{high}})\}.
\]

\[
f_{\text{mac}}(a) = \begin{cases} 
  a_{\text{min}} & \text{for } a \leq \hat{a}_{\text{low}} \\
  a_{\text{min}} + (a - \hat{a}_{\text{low}}) \cdot \frac{a_{\text{max}} - a_{\text{min}}}{\hat{a}_{\text{high}} - \hat{a}_{\text{low}}} & \text{for } \hat{a}_{\text{low}} < a < \hat{a}_{\text{high}} \\
  a_{\text{max}} & \text{for } a \geq \hat{a}_{\text{high}}
\end{cases}
\]
Histogram Equalization

- Adjust 2 different images to make their histograms (intensity distributions) similar
- Apply a point operation that changes histogram of modified image into **uniform distribution**
Histogram Equalization

Spreading out the frequencies in an image (or equalizing the image) is a simple way to improve dark or washed out images.

Can be expressed as a transformation of histogram:

- \( r_k \): input intensity
- \( s_k \): processed intensity
- \( k \): the intensity range (e.g. 0.0 – 1.0)

\[
 s_k = T(r_k)
\]
Equalization Transformation Function

Equalization Transformation Functions

Different equalization function (1-4) may be used
Equalization Examples

Equalization Examples
Equalization Examples

References

  - Histograms (Ch 4)
  - Point operations (Ch 5)
- University of Utah, CS 4640: Image Processing Basics, Spring 2012
- Rutgers University, CS 334, Introduction to Imaging and Multimedia, Fall 2012