BSB 663
Image Processing

Aykut Erdem
Dept. of Computer Engineering
Hacettepe University

Point Operations
Histogram Processing

Today’s topics
• Point operations
• Histogram processing

Digital images
• Sample the 2D space on a regular grid
• Quantize each sample (round to nearest integer)

Image thus represented as a matrix of integer values.

Source: K. Grauman, S. Belongie
**Image Transformations**

- \( g(x,y) = T[f(x,y)] \)

\( g(x,y) \): output image  
\( f(x,y) \): input image  
\( T \): transformation function

1. Point operations: operations on single pixels
2. Spatial filtering: operations considering pixel neighborhoods
3. Global methods: operations considering whole image

**Basic types of operations**

- **Point operations:** range only  
  \( g(x,y) = f(t_x(x,y), t_y(x,y)) \)

- **Domain operations**  
  \( g(x,y) = f(t_x(x), t_y(y)) \)

- **Neighborhood operations:** domain and range

**Point operations**

- Smallest possible neighborhood is of size 1x1
- Process each point independently of the others
- Output image \( g \) depends only on the value of \( f \) at a single point \((x,y)\)
- Map each pixel’s value to a new value
- Transformation function \( T \) remaps the sample’s value:  
  \[ s = T(r) \]

  where
  - \( r \) is the value at the point in question
  - \( s \) is the new value in the processed result
  - \( T \) is an intensity transformation function
Point operations
- Is mapping one color space to another (e.g. RGB2HSV) a point operation?
- Is image arithmetic a point operation?
- Is performing geometric transformations a point operation?
  - Rotation
  - Translation
  - Scale change
  - etc.

Sample intensity transformation functions
- Image negatives
- Log transformations
  - Compresses the dynamic range of images
- Power-law transformations
  - Gamma correction

Sample intensity transformation functions
Original
Darken
Lower Contrast
Non-Linear
Lower Contrast

Invert
Lighten
Raise Contrast
Non-Linear
Raise Contrast

Source: S. Narasimhan

Point Processing Examples
produces an image of higher contrast than the original by darkening the intensity levels below k and brightening intensities above k
produces a binary (two-intensity level) image
Dynamic range

- Dynamic range \( R_d = \frac{I_{\text{max}}}{I_{\text{min}}} \) or \( \frac{(I_{\text{max}} + k)}{(I_{\text{min}} + k)} - \) determines the degree of image contrast that can be achieved
- a major factor in image quality

- Ballpark values
  - Desktop display in typical conditions: 20:1
  - Photographic print: 30:1
  - High dynamic range display: 10,000:1

Point Operations: Contrast stretching and Thresholding

- Contrast stretching: produces an image of higher contrast than the original

- Thresholding: produces a binary (two-intensity level) image

Intensity encoding in images

- Recall that the pixel values determine how bright that pixel is.
- Bigger numbers are (usually) brighter

- Transfer function: function that maps input pixel value to luminance of displayed image

\[
I = f(n) \quad f : [0, N] \rightarrow [I_{\text{min}}, I_{\text{max}}]
\]

- What determines this function?
  - physical constraints of device or medium
  - desired visual characteristics

Point Operations: Intensity-level Slicing

- highlights a certain range of intensities

Intensity encoding in images

- Recall that the pixel values determine how bright that pixel is.
- Bigger numbers are (usually) brighter

- Transfer function: function that maps input pixel value to luminance of displayed image

\[
I = f(n) \quad f : [0, N] \rightarrow [I_{\text{min}}, I_{\text{max}}]
\]

- What determines this function?
  - physical constraints of device or medium
  - desired visual characteristics
What this projector does?

- Something like this:

\[ I(n) \]

\[ n = 64 \]
\[ n = 128 \]
\[ n = 192 \]

\[ I = 0.25 \quad I = 0.5 \quad I = 0.75 \]

Constraints on transfer function

- Maximum displayable intensity, \( I_{\text{max}} \)
  - how much power can be channeled into a pixel?
    - LCD: backlight intensity, transmission efficiency (<10%)
    - projector: lamp power, efficiency of imager and optics

- Minimum displayable intensity, \( I_{\text{min}} \)
  - light emitted by the display in its "off" state
    - e.g. stray electron flux in CRT, polarizer quality in LCD

- Viewing flare, \( k \): light reflected by the display
  - very important factor determining image contrast in practice
    - 5% of \( I_{\text{max}} \) is typical in a normal office environment [sRGB spec]
    - much effort to make very black CRT and LCD screens
    - all-black decor in movie theaters

Transfer function shape

- Desirable property: the change from one pixel value to the next highest pixel value should not produce a visible contrast
  - otherwise smooth areas of images will show visible bands

- What contrasts are visible?
  - rule of thumb: under good conditions we can notice a 2% change in intensity
  - therefore we generally need smaller quantization steps in the darker tones than in the lighter tones
  - most efficient quantization is logarithmic

How many levels are needed?

- Depends on dynamic range
  - 2% steps are most efficient:
    - \( 0 \leftrightarrow I_{\text{min}} \quad 1 \leftrightarrow 1.02I_{\text{min}} \quad 2 \leftrightarrow (1.02)^2I_{\text{min}} \ldots \)
    - \( \log 1.02 \) is about \( 1/120 \), so 120 steps per decade of dynamic range
      - 240 for desktop display
      - 360 to print to film
      - 480 to drive HDR display

- If we want to use linear quantization (equal steps)
  - one step must be < 2% (1/50) of \( I_{\text{min}} \)
  - need to get from \( 0 \) to \( I_{\text{min}} \) \( \cdot R_d \) so need about 50 \( R_d \) levels
    - 1500 for a print; 5000 for desktop display; 500,000 for HDR display

- Moral: 8 bits is just barely enough for low-end applications
  - but only if we are careful about quantization

Source: S. Marschner

Adapted from S. Marschner

Source: S. Marschner

[Image of a page with text and diagrams related to projector capabilities, transfer functions, and image processing constraints.]
Intensity quantization in practice

- **Option 1: Linear quantization**  
  \[ I(n) = \frac{n}{N} I_{\text{max}} \]  
  - pro: simple, convenient, amenable to arithmetic  
  - con: requires more steps (wastes memory)  
  - need 12 bits for any useful purpose; more than 16 for HDR

- **Option 2: Power-law quantization**  
  \[ I(n) = \left( \frac{n}{N} \right)^\gamma I_{\text{max}} \]  
  - pro: fairly simple, approximates ideal exponential quantization  
  - con: need to linearize before doing pixel arithmetic  
  - con: need to agree on exponent  
  - 8 bits are OK for many applications; 12 for more critical ones

- **Option 3: Floating-point quantization**  
  \[ I(x) = \frac{x}{w} I_{\text{max}} \]  
  - pro: close to exponential; no parameters; amenable to arithmetic  
  - con: definitely takes more than 8 bits  
  - 16-bit “half precision” format is becoming popular

Source: S. Marschner

Why gamma?

- Power-law quantization, or gamma correction, is most popular
- Original reason: CRTs are like that  
  - intensity on screen is proportional to (roughly) voltage^2
- Continuing reason: inertia + memory savings  
  - inertia: gamma correction is close enough to logarithmic that there’s no sense in changing  
  - memory: gamma correction makes 8 bits per pixel an acceptable option

Source: S. Marschner

Gamma quantization

- Close enough to ideal perceptually uniform exponential

Source: S. Marschner

Gamma correction

- Sometimes (often, in graphics) we have computed intensities \( a \) that we want to display linearly
- In the case of an ideal monitor with zero black level,  
  \[ I(n) = \left( \frac{n}{N} \right)^\gamma \]  
  (where \( N = 2^n - 1 \) in \( n \) bits). Solving for \( n \):  
  \[ n = \frac{\gamma}{\log_2 a} \]
- This is the “gamma correction” recipe that has to be applied when computed values are converted to 8 bits for output  
  - failing to do this (implicitly assuming gamma = 1) results in dark, oversaturated images

Source: S. Marschner
**Gamma correction**

- Corrected for $\gamma$ lower than display
- Corrected for $\gamma$ higher than display

**Color transformations**

- Some operations discussed on gray level images may be applied bankwise or globally on RGB images
  - Inversion
  - Brightness and contrast control
  - Linear RGB transformation
- Many operations are expressed in HSV or other color spaces
  - Hue transformations
  - Saturation transformations

**Examples of RGB Transformations**

- Inversion, brightness/contrast changes, equalization, non-linear transformation

**Linear RGB Transformations**

- Linear combinations with only positive components
- and with negative components
**Hue / Saturation Changes**

- Hue/saturation transformations are computed in the HSV color space.

**Arithmetical operations**

- Pixel values of two source images are added together or subtracted from each other.
  - The sum of two images is defined as:
    \[ K(x, y) = \text{Quant}_S(\text{Real}_I(I(x, y)) + \text{Real}_I(J(x, y))) \]
  - The difference of two images is defined as:
    \[ K(x, y) = \text{Quant}_S(\text{Real}_I(I(x, y)) - \text{Real}_I(J(x, y))) \]
  - A scale factor and an offset are needed to adjust the result.

**Multiple Image Point Operations**

- Point operations may be defined on two source images \( I \) and \( J \) in order to produce a result image \( K \):
  - Logical operations (for binary images)
  - Arithmetical operations (addition, subtraction, ...)
  - Minimum / maximum operations
  - ...
- Some operations may use additional pixel based parameters which are considered as mask layers:
  - Layer based image compositions
  - Layer based color transformations

**Linear Combinations**

- The previous operations can be generalized to linear combinations:
  - The combination of \( a \) times \( I \) and \( b \) times \( J \) is defined as:
    \[ K(x, y) = \text{Quant}_S(a \cdot \text{Real}_I(I(x, y)) + b \cdot \text{Real}_I(J(x, y))) + c \]
  - where:
    \[ s = \frac{1}{a + b} \]
    \[ t = \frac{2}{a + b} \]
    \[ x = a + b \]
Layer Based Image Composition

Layer Based Color Transformation

Instagram Filters

- How do they make those Instagram filters?

“It’s really a combination of a bunch of different methods. In some cases we draw on top of images, in others we do pixel math. It really depends on the effect we’re going for.” - Kevin Systrom, co-founder of Instagram

Example Instagram Steps

1. Perform an independent RGB color point transformation on the original image to increase contrast or make a color cast
Example Instagram Steps

2. Overlay a circle background image to create a vignette effect

3. Overlay a background image as decorative grain

4. Add a border or frame

Result

Javascript library for creating Instagram-like effects, see: http://alexmic.net/filtrr/
Today’s topics
• Point operations
• Histogram processing

Histogram
• Histogram: a discrete function \( h(r) \) which counts the number of pixels in the image having intensity \( r \)
• If \( h(r) \) is normalized, it measures the probability of occurrence of intensity level \( r \) in an image

What histograms say about images?
A descriptor for visual information

What they don’t?
– No spatial information

Images and histograms
• How do histograms change when
  – we adjust brightness?
  – we adjust contrast?
  shifts the histogram horizontally
  stretches or shrinks the histogram horizontally

Histogram
• Global histogram has no location information at all
• All of these images have the same color histogram!

Source: A. Farhadi

• Level Operations (Part 2)
Very useful on cameras

- Allows you to tell if you use the dynamic range entirely.

Histogram equalization

- A good quality image has a nearly uniform distribution of intensity levels. Why?
- Every intensity level is equally likely to occur in an image

- Histogram equalization: Transform an image so that it has a uniform distribution
  - create a lookup table defining the transformation
Histogram as a probability density function

- Recall that a normalized histogram measures the probability of occurrence of an intensity level \( r \) in an image.
- We can normalize a histogram by dividing the intensity counts by the area

\[
p(r) = \frac{h(r)}{\text{Area}}
\]

Histogram equalization: Continuous domain

- Define a transformation function of the form

\[
s = T(r) = (L - 1) \int p(w) \, dw
\]

where
- \( r \) is the input intensity level
- \( s \) is the output intensity level
- \( p \) is the normalized histogram of the input signal
- \( L \) is the desired number of intensity levels

(Continuous) output signal has a uniform distribution!
Histogram equalization: Discrete domain

- Define the following transformation function for an $M \times N$ image

$$s_k = T(r_k) = (L-1) \sum_{j=0}^{N} \frac{n_j}{MN} = \frac{(L-1)}{MN} \sum_{j=0}^{N} n_j$$

for $k = 0, \ldots, L-1$

where

- $r_k$ is the input intensity level
- $s_k$ is the output intensity level
- $n_j$ is the number of pixels having intensity value $j$ in the input image
- $L$ is the number of intensity levels

(Discrete) output signal has a nearly uniform distribution!

A toy example

<table>
<thead>
<tr>
<th>$r_k$</th>
<th>$n_k$</th>
<th>$p(r_k)$</th>
<th>$n_k/MN$</th>
<th>$\sum_{u} p(u)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>700</td>
<td>0.19</td>
<td>0.19</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1023</td>
<td>0.25</td>
<td>0.25</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>850</td>
<td>0.21</td>
<td>0.21</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>656</td>
<td>0.16</td>
<td>0.16</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>329</td>
<td>0.08</td>
<td>0.08</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>245</td>
<td>0.06</td>
<td>0.06</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>122</td>
<td>0.03</td>
<td>0.03</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>81</td>
<td>0.02</td>
<td>0.02</td>
<td>1</td>
</tr>
</tbody>
</table>

Histogram equalization

Goal: Given $n \times m$ image $f$ with 8 bpp (brightness values 0 to 255), create a new image $g$ that has about $nm/255$ pixels of each brightness value

- Compute $f$’s histogram: $H(i), 0 \leq i \leq 255$
- Compute $f$’s cumulative histogram: $C(i) = \sum_{j=0}^{i} H(j)$
- Compute mapping and output image:

$$g(i, j) = \text{round} \left( \frac{255 \left( C(i, j) - C_{\text{min}} \right)}{nm - C_{\text{min}}} \right)$$

where $C_{\text{min}}$ = minimum non-zero value in $C$

A toy example
**Histogram Specification**

- Given an input image \( f \) and a specific histogram \( p_2(r) \), transform the image so that it has the specified histogram.

- How to perform histogram specification?

- Histogram equalization produces a (nearly) uniform output histogram.

- Use histogram equalization as an intermediate step.

**3D Color Histograms**

- The color histogram of a RGB image is a sparse 3D array.
  - It can be represented by a 3D color plot.
  - Alternatively, 2D or 1D projections may be used.

**1D Histograms of Color Images**

- Instead of 3D color histograms, three (one for each color bank) independent 1D histograms are often used.
  - It corresponds to projection of the color histogram on the R, G, B axis.
  - Primary color correlation is lost!

**Histogram Specification**

1. Equalize the histogram of the input image
   \[
   T_1(r) = (L - 1) \int_0^r p_1(w) \, dw
   \]

2. Histogram equalize the desired output histogram
   \[
   T_2(r) = (L - 1) \int_0^r p_2(w) \, dw
   \]

3. Histogram specification can be carried out by the following point operation:
   \[
   s = T(r) = T_2^{-1}(T_1(r))
   \]
Next week

• Spatial filtering