Image Processing
• An image can be represented by functions of two spatial variables $f(x,y)$, where $f(x,y)$ is the brightness of the gray level of the image at a spatial coordinate $(x,y)$

• A multispectral image is a vector-valued function with components $(f_1, f_2, \ldots, f_n)$; a special case is a color image in which the components measure the brightness values of each of three wavelengths, that is:

\[
\mathbf{f}(\mathbf{x}) = \{f_{red}(\mathbf{x}), f_{green}(\mathbf{x}), f_{blue}(\mathbf{x})\}
\]

\[
\mathbf{x} = (x, y)
\]
(0,0) is (usually) the top-left corner. Other standards exist...
RGB planes decomposed…

\[ f(x) = \{f_{\text{red}}(x), f_{\text{green}}(x), f_{\text{blue}}(x)\} \]
Point Operations

• In a point operation each pixel in the output image is a function of the grey-level (or color value) of the pixel at the corresponding position in the input image

\[ \text{out}(x,y) = f(\text{in}(x,y)) \]

• For example: photometric decalibration, contrast stretching, thresholding, background subtraction…
Histogram

- A grey level histogram is a function that gives the frequency of occurrence of each gray level in the image.
- If the gray levels are quantized in $n$ values (usually 256), the value of the histogram at a particular gray level $p$, $h(p)$, is the number of pixels in the image with that gray level.
- Often it is expressed in terms of fraction of pixels.

(image 512x512)
How do we compute the histogram

\[ \text{function histo=computeHisto(A)} \]

\[ \text{histo=zeros(1,256);} \]

\[ \text{R=size(A,1);} \]
\[ \text{C=size(A,2);} \]

\[ \text{for } r=1:R \]
\[ \quad \text{for } c=1:C \]
\[ \quad \quad \text{index=A(r,c);} \]
\[ \quad \quad \text{histo(index+1)=histo(index+1)+1;} \]
\[ \quad \text{end} \]
\[ \text{end} \]
Some characteristic histograms…

A black image

A white image

A gray image
shadows

midtones

highlights
function S=negative(A)

R=size(A,1);
C=size(A,2);

%prepare image
S=zeros(R,C);

... for r=1:R
    for c=1:C
        S(r,c)=255-double(A(r,c));
    end
end
Threshold

- Produces a two-level image
- We pick a threshold \( t \), we set to 255 all pixels whose value > \( t \), 0 all the others
Histogram Stretch

- From the histogram it is possible to see if there are levels in the image that are not used.
- We can map the levels of the image to expand the histogram.

![Histogram Stretch Graph](image)
function S=stretchHisto(A, min, max)

%%%%%% build look up table
lut=zeros(1,256);
for i=0:255
    if (i<min)
        lut(i+1)=0;
    elseif (i>max)
        lut(i+1)=255;
    else
        lut(i+1)=(i-min)*255/(max-min);
    end
end

%%%%%

R=size(A,1);
C=size(A,2);

%prepare image
S= zeros(R,C);

for r=1:R
    for c=1:C
        index= A(r,c)+1;
        S(r,c)=lut(index);
    end
end

SINA – 11/12
Histogram equalization

- Equally use all gray levels
- Find a transformation $y=f(x)$ to “flatten” the histogram

#pixels is unchanged:
$$p(y) \cdot dy = p(x) \cdot dx$$
Assume $x$ and $y$ between 0 and 1

We would like $p(y) = \text{constant} = 1$

$$p(y) \cdot dy = p(x) \cdot dx$$

$$dy = p(x) \cdot dx \Rightarrow \frac{dy}{dx} = p(x)$$

$$y(x) = \int_0^x p(u)du$$

cumulative probability distribution

- low $p(x) \rightarrow$ smooth $f(x) \rightarrow$ narrow $d(y)$
- large $p(x) \rightarrow$ steep $f(x) \rightarrow$ large $d(y)$
Discrete case

- \( x \) and \( y \) assume discrete values between \([0, L-1]\) (often \( L=256 \))

\[
y(x) = \int_0^x p(u)du \quad \rightarrow \quad y'(x) = \sum_{0}^{x} P_i
\]

- \( P_i = \frac{n_i}{N} \) \( \leftarrow \) number of pixels that have value \( i \)

- \( N = \text{total number of pixels (HxW)} \)

- \( y' \) assumes values between \([0,1]\) \( \rightarrow \) needs to be scaled to \([0, L-1]\):

\[
y = \left\lfloor \frac{y' - y'_{\min}}{1 - y'_{\min}} (L - 1) + 0.5 \right\rfloor
\]

- integral part of a real number
Example

Note: the resulting histogram is not perfectly flat because we cannot separate pixels having the same gray level, the resulting cumulative distribution, however, would approximate a linear ramp.

SINA – 11/12
Detect Changes

- Take the difference between each pixel in two images $A$ and $B$ (grayscale):
  \[ B=\text{“background”} \]
  \[ A=\text{new image} \]
  \[ D=\text{abs}(A-B) \]

- Extend the concept to a sequence of images
- At each instant in time we take the difference between the current frame and the previous one:
  \[ D=\text{abs}(A(t)-A(t-1)) \]

Detection can be done by thresholding:
\[ Out=\text{threshold}(D,th); \]
function imageDiff(basename, start, last)

cFrame=sprintf('%s%d.ppm', basename, start);
A=imRead(cFrame);
PREV=rgb2Gray(A);

for i=start:last
    cFrame=sprintf('%s%d.ppm', basename, i);
    A=imRead(cFrame);
    G=rgb2Gray(A);
    D=double(G)-double(PREV);
    D=abs(D);
    diff_th=im2bw(uint8(D),50/255);
    PREV=G;

    figure(1), subplot(1,2,1), imShow(uint8(A)), drawnow;
    figure(1), subplot(1,2,2), imShow(uint8(255*diff_th)), drawnow;

    pause(0.05); %%%wait some time
end
Another option

• Model the background by taking into account more than a single frame:

\[ B = a \cdot A(t-1) + (a-1) \cdot B \]
\[ D = \text{abs}(A(t)-B) \]

\(a\) determines how fast we update the background:

\(a = 1\) → image difference
\(a = 0\) → persistent background (never updated)
Color Histograms

• Count the color of the pixels of the images
• It is a statistical description of the color of the image, useful to characterize a particular object
• Appealing because invariant to translation and rotation, slowly changing with scale and viewpoint
  – r,g,b $\rightarrow$ 3D function, intensity dependent, easily too large (es: 256x256x256x32 $\sim$ 64MB)
  – discard luminance, use H,S or r,g $\rightarrow$ 2D
Color Histogram: examples

bin size: 16x16
Comparing Histograms

• Suppose we want to compare two histograms I and M, each with \( n \) bins
• Useful to solve the *identification problem*: compare two images M and I and decide if they are similar
• Intersection, the number of pixels from the model that correspond to pixels of the same color in the image, formally:

\[
\sum_{j=1}^{n} \min(I_j, M_j)
\]

• Normalize by the number of pixels in the histogram M:

\[
H(I, M) = \frac{\sum_{j=1}^{n} \min(I_j, M_j)}{\sum_{j=1}^{n} M_j}
\]

Swain and Ballard 1991
\[ M(j) \]

\[ I(j) \]

\[ \min(I(j), M(j)) \]
Histogram Backprojection

- Assume we have a model of an object (its color histogram)
- Localization problem: where in the image are the colors of the object being looked for?
- The histogram gives the probability of occurrence of the colors of the object, or \( p(\text{color/object}) \)
- We can approximate:

\[
p(\text{object} | \text{color}) = \frac{p(\text{color} | \text{object}) \cdot p(\text{object})}{p(\text{color})} \quad \square \quad p(\text{color} | \text{object})
\]

\( (h,s) \leftarrow \text{in}(r,c) \)

\( H(h,s) \rightarrow \text{out}(r,c) \)
• Similar approach, compute the “ratio histogram” (Swain and Ballard, 1991):

\[
R_i = \min\left(\frac{M_i}{I_i}, 1\right)
\]

• Perform backprojection of \( R \) into the image
• Heuristic to deemphasize colors that are not in the object looked for (for which \( I > M \))
• Search for a uniform region whose size matches the one of the object
Compute histogram

Backprojection (ratio histogram)
Examples:

• Swain and Ballard 1991, use color histograms to recognize objects

• Skin detection, preprocessing for face detection…
  – Example (Peer 2003)
  Assume (r,g,b) space (and daylight illumination)
  classify (r,g,b) as skin if:
  \[
  r > 95 \quad \text{and} \quad g > 40 \quad \text{and} \quad b > 20, \\
  Max\{r,g,b\} – min\{r,g,b\} > 15, \quad \text{and} \\
  |r-b| > 15 \quad \text{and} \quad r > g \quad \text{and} \quad r > b
  \]