Computer Vision & Digital Image Processing

Histogram Processing I

- The histogram of a digital image, \( f \), (with intensities \([0, L-1]\)) is a discrete function:
  \[ h(r_k) = n_k \]
- Where \( r_k \) is the \( k \)th intensity value and \( n_k \) is the number of pixels in \( f \) with intensity \( r_k \)
- Normalizing the histogram is common practice:
  - Divide the components by the total number of pixels in the image
  - Assuming an \( M \times N \) image, this yields
    \[ p(r_k) = \frac{n_k}{MN} \text{ for } k=0,1,2,\ldots,L-1 \]
  - \( p(r_k) \) is, basically, an estimate of the probability of occurrence of intensity level \( r_k \) in an image
  \[ \sum p(r_k) = 1 \]

Uses for Histogram Processing

- Image enhancements
- Image statistics
- Image compression
- Image segmentation
- Simple to calculate in software
- Economic hardware implementations
  - Popular tool in real-time image processing
- A plot of this function for all values of \( k \) provides a global description of the appearance of the image (gives useful information for contrast enhancement)

Histogram Examples

- Four basic image types and their corresponding histograms:
  - Dark
  - Light
  - Low contrast
  - High contrast
- Histograms commonly viewed in plots as:
  - \( h(r_k) = n_k \text{ versus } r_k \)
  - \( p(r_k) = n_k/\text{MN} \text{ versus } r_k \)

MATLAB Histogram Function

```matlab
function [h]=histogram(f);
[xmax,ymax]=size(f);
h=linspace(0,0,256);
for x=1:xmax
  for y=1:ymax
    h(f(x,y))=h(f(x,y))+1;
  end;
end;
```

Histogram Equalization

- Histogram equalization is a process for increasing the contrast in an image by spreading the histogram out to be approximately uniformly distributed
- The gray levels of an image that has been subjected to histogram equalization are spread out and always reach white
  - The increase of dynamic range produces an increase in contrast
- For images with low contrast, histogram equalization has the adverse effect of increasing visual graininess
Histogram Equalization (assumptions)

- The intensity transformation function we are constructing is of the form
  \[ s = T(r) \quad 0 \leq r \leq L - 1 \]
- An output intensity level \( s \) is produced for every pixel in the input image having intensity \( r \)
- We assume
  - \( T(r) \) is monotonically increasing in the interval \( 0 \leq r \leq L - 1 \)
  - \( 0 \leq T(r) \leq L - 1 \) for \( 0 \leq r \leq L - 1 \)
- If we define the inverse
  \[ r = T^{-1}(s) \quad 0 \leq s \leq L - 1 \]
- Then \( T(r) \) should be strictly monotonically increasing

Histogram Equalization (continued)

- Histogram equalization requires construction of a transformation function \( s_k \)
  \[ s_k = T(c) = \sum_{j} \frac{n_j}{M \times N} \]
- where \( r_k \) is the \( k \)th gray level, \( n_k \) is the number of pixels with that gray level, \( M \times N \) is the number of pixels in the image, and \( k = 0,1, \ldots, L - 1 \)
- This yields an \( s \) with as many elements as the original image’s histogram (normally 256 for our test images)
- The values of \( s \) will be in the range \([0,1]\). For constructing a new image, \( s \) would be scaled to the range \([1,256]\)

An Interactive MATLAB Histogram Function

```matlab
function winhist(action)
    global p1
    global p2
    global FIG
    if nargin<1,
        action='initialize';
    end;
    if strcmp(action,'initialize'),
        figNumber=figure( ...
            'Name','Histogram Plot', ...
            'NumberTitle','off', ...
            'Position',[100 100 500 500], ...
            'Visible','off');
        colordef(figNumber,'black')
        p1=axes( ...
            'Position',[0.25 0.55 0.40 0.40]);
        p2=axes( ...
            'Position',[0.25 0.05 0.40 0.40]);
    end
    if strcmp(action,'histogram'),
        hold all;
        histc(p1,M(:),r,1);[histb,figNumber]=figure( ...
            'Name','Histogram Plot', ...
            'NumberTitle','off', ...
            'Position',[100 100 500 500], ...
            'Visible','off');
        colordef(figNumber,'black')
        p1=axes( ...
            'Position',[0.25 0.55 0.40 0.40]);
        p2=axes( ...
            'Position',[0.25 0.05 0.40 0.40]);
```

An Interactive MATLAB Histogram Function

```matlab
function winhist(action)
    global p1
    global p2
    global FIG
    if nargin<1,
        action='initialize';
    end;
    if strcmp(action,'initialize'),
        figNumber=figure( ...
            'Name','Histogram Plot', ...
            'NumberTitle','off', ...
            'Position',[100 100 500 500], ...
            'Visible','off');
        colordef(figNumber,'black')
        p1=axes( ...
            'Position',[0.25 0.55 0.40 0.40]);
        p2=axes( ...
            'Position',[0.25 0.05 0.40 0.40]);
```
% The LOAD IMAGE button
btnNumber=1;
yPos=top-(btnNumber-1)*(btnHt+spacing);
labelStr='Load Image';
callbackStr='winhist(''load'')';
% Generic button information
btnPos=[left yPos-btnHt btnWid btnHt];
uicontrol(...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',btnPos, ...
'String',labelStr, ...
'Callback',callbackStr);

% Now uncover the figure
set(figNumber,'Visible','on');

% The Histogram button
btnNumber=2;
yPos=top-(btnNumber-1)*(btnHt+spacing);
labelStr='Histogram';
callbackStr='winhist(''histogram'')';
% Generic button information
btnPos=[left yPos-btnHt btnWid btnHt];
uicontrol(...
'Style','pushbutton', ...
'Units','normalized', ...
'Position',btnPos, ...
'String',labelStr, ...
'Callback',callbackStr);

elseif strcmp(action,'histogram'),
    axes(p2);
h=histogram(FIG);
bar(h,'w'),axis([1 256 0 max(h)*1.10]);
elseif strcmp(action,'load'),
    axes(p1);
cd('L:\ece582\matlab');
[file,path]=uigetfile('*.bmp','Open');
[f, fmap]=bmpread(fullfile(path, file));
FIG=f;
image(f);colormap(gray(256));
end;