ENGI 7854/9804 Image Processing and Applications

Laboratory 1

Geometric Transformation and Histogram Equalization

Introduction

The objective of this laboratory is to become familiar with geometric transformations and histogram equalization.

A geometric transformation of an image is a linear transformation applied to the spatial coordinate system. In this transformation each point of an image is mapped to a point in an new image. An affine transformation is defined as a transformation which preserves collinearity (straight lines are preserved), among other things. In general an affine transformation is a composition of rotations, translations, scaling and shear operations.

The histogram of an image is the intensity distribution of the image. Histogram equalization is a technique for adjusting image intensities to enhance contrast based on the underlying histogram. Histogram equalization is an invertible technique and it is not computationally intensive.

Geometric Transformation

An affine transformation of an image can be represented as a matrix multiplication.

$$[x\ y\ 1] \ = \ [u\ v\ 1]\mathbf{T}$$

A general transformation matrix can be given as:

$$\mathbf{T} = \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

Any complex affine transformation can be represented as a combination of translation, rotation, scale and shear transformation matrices. Transformation matrices for several common transformations are given bellow:

Transformation Name	Transformation Matrix
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Shear(vertical)	$\begin{bmatrix} 1 & s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Shear(horizontal)	$\begin{bmatrix} 1 & 0 & 0 \\ s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Histogram Equalization

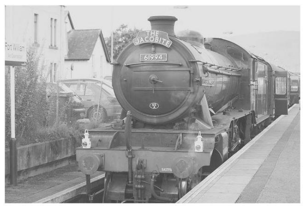
A histogram of a grayscale image gives the probability of the occurrence of all gray levels. Histogram can be represented as $h(r_k) = n_k/n$, where r_k is the k^{th} gray level, n_k is the number of pixels in the image having gray level r_k and n is the total number of pixels. The histogram of an image with low contrast is narrow and the histogram of a high-contrast image covers a broad range of the grayscale spectrum.

Histogram equalization is a method that is commonly used to transform a low contrast image in to a high contrast image by adjusting the pixel intensities in the image such that they occupy a broader range of gray values. Additionally the distribution will become uniform (or as close to uniform as possible).

The transformation function for histogram equalization can be expressed as:

$$s_k = \mathbf{T}(r_k) = \sum_{j=0}^{k} p_r(r_j) \tag{1}$$

where s_k is the new intensity for the pixels having original intensity r_k and $p_r(r_j) = n_j/n$.


(a) Low contrast image

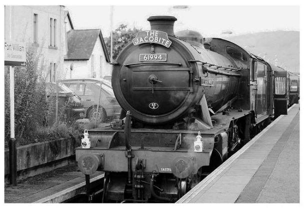

(b) High contrast image

Figure 1: Results of histogram equalization

Figure 1a shows an image with low contrast and Figure 1b shows the same image after histogram equalization.

Procedure

Geometric Transformation

1. Download the test image (im1.png) from Brightspace under *Lab 01* and save it in your working directory.

2. Read the image and convert the image to a grayscale image.

```
imc = imread('im1.png');% Read the image
img = rgb2gray(imc);     % Convert to grayscale. The supplied
    image is a grayscale, so you don't need to use this step.
imshow(img);             % View image
```

3. Define the transformation matrix with a rotation angle of 45 degrees.

```
theta = 0.25*pi;
R = [cos(theta) sin(theta) 0; ...
-sin(theta)     cos(theta) 0; ...
0 0 1];
```

4. Calculate the boundary of the transformed image and generate an empty image with the calculated size.

```
[y_max, x_max] = size(img); % Obtaining the size of the image
corners = [0,0,1; ...
        x_max,0,1; ...
        0,y_max,1; ...
        x_max,y_max,1]; % Corner points of the image
new_corners = corners*R % New location of corners
new_width = round(max(new_corners(:,1)) - ...
        min(new_corners(:,1)));
new_height = round(max(new_corners(:,2)) - ...
        min(new_corners(:,2)));
new_img = zeros(new_height,new_width);
```

5. Transform each pixel with the transformation matrix and assign the intensity to the new location.

```
min_width = abs(min(new_corners(:,1)));
min_height = abs(min(new_corners(:,2)));

min_width = round(min_width)+1;
min_height = round(min_height)+1;

rot_img = zeros(new_height,new_width);

for i = 1:y_max
  for j = 1:x_max
    temp = ([j-1 i-1 1])*R;
    x_new = round(temp(1,1))+ min_width;
    y_new = round(temp(1,2))+ min_height;
    rot_img(y_new,x_new) = img(i,j);
  end
end
```

6. View the transformed image

```
figure;
imshow(rot_img,[])
```

7. Complete steps 3-6 for angles $\theta = 90^0$ and $\theta = -25^0$.

8. It can be seen that in some cases, when the image is transformed the output image has black dots. In these cases, the image can be improved using the following method:

   i Obtain the boundary of transformed image and generate an empty image of same size as the transformed image.

   ii Map each pixel of the new, empty image using the inverse of the transformation matrix.

   iii Assign the value of the pixel closest to the mapped location to the point of new image (If the mapped location is outside the image keep the value of the point zero).

9. Modify the given code to implement the method given in 8. Provide a breif discussion on why some images have these black pixels.

10. Use the modified code in 9 to calculate the following transformations.

   i Translation $(t_x = 50, t_y = 45)$

   ii Scale $(c_x = 0.5, c_y = 1.5)$

   iii Shear (vertical) $(s_v = 0.2)$

iv Shear (horizontal) ($s_v = 0.3$)

v Rotation ($\theta = 50^0$) followed by translation (($t_x = 25, t_y = 30$))

Histogram Equalization

1. Download the test images (im2.png) from Brightspace under *Lab 01* and save it in your working directory.

2. Read the image and convert the image to a grayscale image.

```
imc = imread('im2.png');% Read the image
img = rgb2gray(imc);     % Convert to grayscale
imshow(img);             % View image
```

3. Grayscale images have 256 gray levels. Therefore create an empty matrix having dimensions of 1×256 and generate the histogram.

```
        H=size(img,1); % Read the height of the image
        W=size(img,2); % Read the width of the image

Hist_arr=zeros(1,256); % Array for holding the (original) histogram
Hist_eq_arr=zeros(1,256); % Array for holding the (equalized)
   histogram

        CDF_array=zeros(1,length(Hist_arr)); % array to hold
           cumulative distribution function (CDF)
        hist_eq_img=uint8(zeros(H,W)); % A 2D array for keeping
           histogram equalized image (intensity in 8 bit integers)
        for i=1:H,
          for j=1:W,
            Hist_arr(1,img(i,j)+1)=Hist_arr (1,img(i,j)+1)+1 ;
          end
        end
```

4. Calculate $p_r(r_k)$ for the image. (Total number of pixels $= y_{max} \times x_{max}$)

```
Hist_arr_pdf=Hist_arr/(H*W); % PDF
```

5. Calculate the CDF from the PDF

```
dummy1=0; % A dummy variable to hold the summation results
for k=1:length(Hist_arr); % Generating the CDF from PDF
 dummy1=dummy1+Hist_arr_pdf(k);
 CDF_array(k)= dummy1;
end
```

6. Calculate the equalized histogram using equation 1 and map the original gray levels to the new gray levels.

```
for l=1:H, % Histogram equalization
  for m=1:W,
    hist_eq_img(l,m)= round(CDF_array(img(l,m)+1)*...
      (length(Hist_arr_pdf)-1)); % scale to 255 and round to
          nearest integer
    Hist_eq_arr(1, hist_eq_img(l,m)+1)=...
      Hist_eq_arr (1,hist_eq_img(l,m)+1)+1 ;  % Its histogram
  end
end
```

7. Display both images, before and after histogram equalization.

Performance evaluation

1. Download or use any image that is larger than 640×640 and use your code to rotate the image by 90 degrees. Use the inbuilt functions (e.g., **maketform** and **imtransform** in MATLAB Image Processing Toolbox) to perform the rotation of the same image. Comment on the difference in speed between the two approaches.

2. Comment on the shape of the histogram resulting from the histogram equalization process. Is it a perfectly uniform histogram? If not, comment on why it is not uniform.

Notes

1. You must include all of your code, results and discussion in your lab report.

2. This is an independent lab. You must work on your own.

3. Submit your work to Brightspace in the appropriate dropbox folder.