EE569 Homework #1

Watermark Technologies, Image Enhancement, Noise Removal

Report

6733-3385-34
YUE WU
wuyue@usc.edu
Sept. 17, 2009
Problem 1: Gray Scale Images and Visible Watermark Technologies (30%)
(1) Obtaining Gray-Scale Image from Color Image (10%)

Motivation: We were asked to change color image of "Doheny Labrary" to gray-scale image. We have the original color picture named "Doheny.raw", and want to change it into a gray image name "Dohenygray.raw".

Approach: In a gray-scale image, each pixel has the value of only one component to record its brightness (Y: 0 ~ 255). In a RGB color image, each pixel has the value of three color components (Red, Green, and Blue, respectively) to record the combined color. Suppose that the brightness Y of a pixel and its R, G, B components have the following relationship:

\[ Y = 0.299 \times R + 0.587 \times G + 0.114 \times B \]  \(1)\)

Procedures: First, we have to open the image "Doheny.raw". Then we can convert the color image into a gray image using the equation (1). We use function displayrawall('Doheny.raw',256,256) to open the color image and then execute function rgbtogray() to exchange color image into gray image.

Results:

Open "Doheny.raw"
The gray image converted from "Doheny.raw"

**Discussion:**
The color image which has three dimensions RGB, 256*256*3 is changed into a gray image which has only one dimension, 256*256*1.

**2(2) Embedding Watermarks into Original Image (20%)**

**Motivation:**
Digital watermarking is the process of embedding information into a digital signal. The signal may be audio, pictures or video, for example. In visible watermarking, the information is visible in the picture or video. Typically, the information is text or a logo which identifies the owner of the media.

In this problem, we need to implement the program to embed color USC logo watermark image (usc_logo_color.raw) into the center position of color image “Doheny.raw”. And we also need to implement the program to embed gray-scale USC logo watermark image (usc_logo_gray.raw) into the center position of “Doheny.raw” gray-scale image obtained from the above Problem 1 (1).

![usc_logo_color.raw](image1)

![usc_logo_gray.raw](image2)

**Approach:** First, examine the pixels of the watermark to see whether a pixel is white or
not white. If it is white, then get ride of it and keep the pixel of the original image; if it is not white, then replace the pixel of the original image.

**Procedures:** To embed a color logo to the color image, we first convert the watermark into gray-lever, then we can examine the brightness of the gray-lever image. To embed a gray logo to the gray image, since the logo is one dimension, we can directly examine the brightness of the logo. And decide whether to override the pixel of original image or just ignore it. The procedures is shown as the flow chat as follows.

**Results:**
"usc_logo_color.raw" is embedded into middle of "Doheny.raw"

"usc_logo_gray.raw" is embedded into the center position of "Doheny.raw" gray-scale image

**Discussion:**
We can change the location of the watermark through scanning pixels of different locations. If we want to put the watermark in the center of the original image, scan pixel through row 65 to 192, column 65 to 192 (128*128 usc_logo image, 256*256 Doheny image).
Problem 2: Noise Removal (30%)

(1) Gray-level image (10%)

Motivation:
We have to find a proper filters and parameters to remove the noise from image "pepper_uni.raw" and image "pepper_imp.raw", and compare with the original image "pepper.raw".

Approach:
We use the lowpass filter to reduce the uniform noise in "pepper_uni.raw", and use the median method to reduce the impulse noise in "pepper_imp.raw"

Procedures:
A. Using lowpass filter to reduce uniform noise. The sudden change of gray-level can be seen as high frequency in the frequency domain, the lowpass filter filters the high frequency of the image and reduce the noise. We chose a 3*3 mask.

\[
G = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}
\]

The gray-level of the middle pixel of the mask can be replaced by the average of the gray-level of the 9 pixels. That is:

\[
G(2,2) = [G(1,1)+G(1,2)+G(1,3)+G(2,1)+G(2,2)+G(2,3)+G(3,1)+G(3,2)+G(3,3)]/9
\]

Results:

Pepper_uni.raw  pepper_uni.raw through the lowpass filter

B. 
**Procedures:** The median method to reduce the impulse noise is explained as follows. We use a 3*3 mask to compute the median of the 9 numbers.

\[
A = \begin{bmatrix}
X_1 & X_2 & X_3 \\
X_4 & X_5 & X_6 \\
X_7 & X_8 & X_9 \\
\end{bmatrix}
\]

\[
B_1 = \max[\min(X_1, X_2, X_3), \min(X_2, X_3, X_4), \min(X_3, X_4, X_5)]
\]

\[
C_1 = \max[\min(X_5, X_6, X_7), \min(X_6, X_7, X_8), \min(X_7, X_8, X_9)]
\]

\[
B_2 = \min[\max(X_1, X_2, X_3), \max(X_2, X_3, X_4), \max(X_3, X_4, X_5)]
\]

\[
C_2 = \min[\max(X_5, X_6, X_7), \max(X_6, X_7, X_8), \max(X_7, X_8, X_9)]
\]

\[
\text{Median} = \frac{1}{2} \times \max[B_1, C_1] + \frac{1}{2} \times \min[B_2, C_2]
\]

We use the value of Median to replace the value of \(X_5\), to reduce the impulse noise.

**Result:**

![Pepper_imp.raw](image1.png)  ![reduce the noise by lowpass filter](image2.png)
Reduce the noise by median method for only one time   Reduce the noise by median method for two times

Reduce the noise by median method for three times   original image "pepper.raw"

**Discussion:**
If it is uniform noise, the noises are almost at a high frequency, so that we can just use a lowpass filter to cut the high frequency. If it is impulse noise, some of the noises are white pixels and some are black noises, so that we can use the nonlinear method -- median method to get rid of the white or black noises. However, as we increase the times to reduce the noise, the edge blurs. It would have better effect if we can use some method to sharp the edge. But, this is not included in my program.
(2) Color image (20%)

**Motivation:**

"rose_color_noise.raw" is a color image, where each channel is embedded with mixed noises. As a result, colors are disrupted. We are required to try our best to remove the noise, and compare with the original image "rose_color.raw".

**Approach:**

As "rose_color" is a color image, the image has three dimensions GRB, so we have to reduce the RGB noise separately. And then use the outlier noise cleaning method to reduce the noise.

**Procedures:**

We use a 3*3 mask, compute the average of the 8 values around the pixel X we want to process. If the difference between the average value and the value of X is less than limen L, we then keep the value of X, if their difference is larger than L, we then override X with the average value of the 8 number around it.

\[
A = \begin{bmatrix}
X_1 & X_2 & X_3 \\
X_4 & X & X_5 \\
X_6 & X_7 & X_8
\end{bmatrix}
\]

\[
\text{if} \quad |X - \frac{1}{8} \sum_{i=1}^{8} X_i| > L \\
\text{then} \quad X = \frac{1}{8} \sum_{i=1}^{8} X_i
\]

**Results:**
G=displayrawrose('rose_color_noise.raw',256,256);
K=roenoisegg(G,I);

**Discussion:**
change the value of I, the effect of the "reduced noise" rose image differ a lot. Here pick I=100,I=50,I=20,I=10; Choose the value of I that can best reduce the noise.

**Problem 3: Image Enhancement (40%)**
**(1) Contrast Manipulation (10%)**

**Motivation:**
"griffith_night.raw" is a picture taken at Griffith observatory, one of famous tourist sites in Los Angeles. As this picture was taken at night, it is difficult to see the boundaries between sky, grass, and the Griffith observatory building. However, we can use image enhancement to make boundaries of different objects clearer in this picture. We have to implement two contrast manipulation techniques: full range linear scaling and histogram equalization, to enhance "griffith_night.raw".
**Approach:**
We first convert the color image "griffith_night.raw" into gray-lever image, and then stretch the range of the image gray-lever through linear method and Histogram equalization.

**A.Linear scaling**

**Procedure:**
From observation, we can tell that most pixels has the gray-lever from 0 to 50, and it has a very the narrow range. So we can stretch (0,50) to (0,255) to enhance the contrast. Followed is the linear scale transfer function. F is the input image and the output image. Through H, we stretch the gray-lever of F.

\[
F \rightarrow | \quad H \quad | \rightarrow G
\]

\[
G = H(F)
\]

\[
= G_{\text{min}} + \left( \frac{G_{\text{max}} - G_{\text{min}}}{F_{\text{max}} - F_{\text{min}}} \right) (F - F_{\text{min}})
\]

**Result:**
B. Histogram equalization

Procedure:
Assume the input is F and the output image is G.

\[ F \rightarrow H \rightarrow G \]

Actually, it is to decide the cumulative distribution function (CDF). Suppose a certain pixel G has its gray level is f, then we can integrate the grey-level histogram from 0 to f.

\[ D_F(f) = \int_0^f P_F(\tilde{f})d\tilde{f} \]

\[ D_F(f) = \sum_{i=0}^{f} \frac{n_i}{N} \quad (i=0,1,2...255)(N= \text{number of all the pixels}) \]

\[ 0 \leq f \leq 1 \]

\[ D_G(g) = \int_0^g P_G(\tilde{g})d\tilde{g} = g \]
**Result:**

Histogram equalization griffith_night.raw

**Discussion:**

Below is the histograms of input and output images. histograms of "griffith_night.raw", histograms of "Linear scaling  griffith_night.raw", and histograms of "Histogram equalization griffith_night.raw".
histograms of "griffith_night.raw"

histograms of "Linear scaling   griffith_night.raw"
histograms of "Histogram equalization griffith_night.raw". And we can see, the histograms of "Histogram equalization griffith_night.raw" has a relatively more flat curve. So it has a stronger contrast than linear scaling. And it can be seen clearly when compare picture "Linear scaling griffith_night.raw" and picture "Histogram equalization griffith_night.raw".

(2) Color Manipulation (20%)

Motivation:
We were asked to add colors to the image in "Griffith_night.raw" blue color to the sky and green color to the grass in front of the building. We can use a picture of Griffith observatory taken during the day shown in "Griffith_day.raw" to get appropriate RGB values of sky and grass. (Although there are numerous RGB values both for sky and grass, we may use any single RGB value for sky and grass each.)
Griffith_day.raw

**Approach:**
First, We analyze the picture "Griffith_night", and determine which part belongs to blue sky and which part is green grass.
Then, we analyze the RGB elements in "Griffith_day.raw", and add these color information of blue sky and green grass to "Griffith_night.raw".

**Procedure:**
Analyse the image "Griffith_day.raw", we can divide the image into three parts, from top to the bottom.
In the first top part of the image, all the pixels belong to the sky, so we can just add the value of blue sky to RGB.
In the middle part of the image, it is mixed with building and sky, so we analyse the gray level of the "Linear scaling griffith_night.raw" to tell building and sky. In the typical area, where there is a very bright area in the center of the image "griffith_night.raw", we use specific restrict to distinct the building and sky.
The bottom part of the image is mixed with grasses and road. Analyse the gray lever in the "Linear scaling griffith_night.raw". The gray lever of the pixels that consist the road is generally larger then that consist the grasses. So we can set a limen to tell where is road and where is grasses.

**Result:**
Add color to Linear scaled griffith

**Discussion:**
I choose the linear scale method to add color.

**3) Color Manipulation after Applying Low Pass Filter (10%)**

**Motivation:**
apply a low pass filter to the gray scale image obtained after applying contrast manipulation in part (1). Then try to take advantage of this image to select pixels in the sky and grass region, which need to be colored with blue and green colors. Compare this coloring result with the result from (2).

**Approach:**
First, apply a low pass filter to the gray scale image to reduce noise. Secondly, use (2)'s algorithm to this image.

**Procedure:**
Let the gray scale image to go through a low pass filter that is used in Problem2(2) (reduce the noise of image "rose_color_noise.raw") and then apply the algorithm used the Problem3(2).

**Result:**
Discussion:
The effect would be better if we reduce the noise before we add color to the image.