REU Site Program in CVMA

Lab 6 – Exercises on Color Image Processing and Two Mini-Projects

Problem 1: Color Image Processing

The task here is to help a robot to identify a bright orange ball in its surrounding. The ***ball.bmp*** is an image obtained from a camera mounted on the robot.

a) The Matlab toolbox provides functions for converting between the RGB color space and three commonly used color spaces: YIQ, HSV, and YC_bC_r. Load and convert ***ball.bmp*** to these three color spaces by calling appropriate Matlab functions. Display the three resultant images together with each of their three color channels side-by-side with appropriate titles on figures 1 through 3, respectively. Use the Matlab display command to explain which component in each color space corresponds to the grayscale information of a color image.

b) Find the edges of the image ***ball.bmp*** using the Sobel, Prewitt, Roberts, Laplacian of Gaussian, Zero-Crossing, and Canny methods on the **Value image** obtained from calling the Matlab function, which converts the RGB color image to the HSV image. Display the edges obtained from these six methods on figure 4 with the appropriate title for each subplot. Use the Matlab display command to explain which methods give you good results. In general, how can you tell which edge detector is appropriate for the specific application?

c) In H-space, apply appropriate image processing techniques to separate the ball from the background. Find the centroid of the ball and indicate its location by a cross on the original color image.

Close all figures and clear all variables in the workplace.

Problem 2: Simple Color Image Retrieval

Content-based image retrieval is a challenging research problem. It uses an image as a query to search for similar images in a large image database. Below is a simple algorithm to search for a small image database using visual features, i.e., color histogram.

1) Compute Color Histogram: An image histogram refers to the probability mass function of the image intensities. This is extended for color images to capture the joint probabilities of the intensities of the three color channels. More formally, the color histogram is defined by:

$$h_{A,B,C}(a,b,c) = N \times \Pr ob(A = a, B = b, C = c)$$

where A , B and C represent the three color channels (R, G, B or H, S, V) and N is the number of pixels in the image. Computationally, the color histogram is formed by discretizing the colors within an image and counting the number of pixels of each color. Implement a **Cal256binHSV** function to compute the 256-bin HSV color histogram (i.e., 16-bin H, 4-bin S, and 4-bin V) for the input image. The function prototype should be: Hist = Cal256binHSV(Im). Here, Hist will be a 1-D vector for storing the counts for 256 bins.

2) Assuming that you have a small image database with 4 images, namely Elephant1.jpg, Elephant2.jpg, Horse1.jpg, and Horse2.jpg. Compute the 256-bin HSV color histogram for each image in this small database by calling your function implemented in a). Plot these four histograms on figure 1 with the appropriate title for each subplot. Respectively use each image as a query to search this small database and display the retrieval results on figures 2 through 5. Make sure that each figure displays all retrieved images for each query with the corresponding ranking (top match, second top match, third top match, and fourth top match) and **the associated similarity score**, which is computed by the histogram intersection defined as follows: The intersection of histograms ***h*** and ***g*** is given by:

$$d(h,g) = \frac{\sum_A \sum_B \sum_C \min(h(a,b,c), g(a,b,c))}{\min(|h|,|g|)}$$

where |***h***| and |***g***| gives the magnitude of each histogram, which is equal to the number of samples.

3) Use the Matlab display command to summarize your findings. Particularly, you need to answer at least the following questions: Can we use the color histogram as visual features to represent an image to solve the image retrieval problem? Why?

   Close all figures and clear all variables in the workplace.

Problem 3: A Simple Watermarking Technique in Wavelet Domain

Digital watermarking techniques are viable solutions for copyright protection. Below is a simple algorithm to embed a watermark (a random binary sequence) into an original image and extract the embedded watermark from the watermarked image.

Embedding Procedure:
Apply a 3-level "db9" wavelet decomposition on *Lena* by using an appropriate Matlab function. Randomly generate a sequence ***b*** of 0's and 1's whose length is equal to the size of the approximation image (i.e., the top left subband). Sequentially embed each value in ***b*** into each approximation coefficient ***H*** in a raster-scan order (i.e., from the left to the right and from the top to the bottom). For each paired ***b*** and ***H***, conduct the following operation using $\beta = 30$, 60, and 90, respectively:

$$H'(i,j) = \begin{cases} H(i,j) - \left(H(i,j) \bmod \beta\right) + 0.75\beta & \text{if } b(k) = 1 \text{ and } \left(H(i,j) \bmod \beta\right) \geq 0.25\beta \\ \left[H(i,j) - 0.25\beta\right] - \left[\left(H(i,j) - 0.25\beta\right) \bmod \beta\right] + 0.75\beta & \text{if } b(k) = 1 \text{ and } \left(H(i,j) \bmod \beta\right) < 0.25\beta \\ H(i,j) - \left(H(i,j) \bmod \beta\right) + 0.25\beta & \text{if } b(k) = 0 \text{ and } \left(H(i,j) \bmod \beta\right) \leq 0.75\beta \\ \left[H(i,j) + 0.5\beta\right] - \left[\left(H(i,j) - 0.5\beta\right) \bmod \beta\right] + 0.25\beta & \text{if } b(k) = 0 \text{ and } \left(H(i,j) \bmod \beta\right) > 0.75\beta \end{cases}$$

Independently perform the inverse wavelet transform after the operation with each of the three β values. For each of the reconstructed images generated by using different β's, display the original image, the watermarked image, and the difference image (i.e., the difference between the original image and the reconstructed image) on a figure. Due to the small differences, you need to apply a scaling operation for a proper display. **Note: Here, the three reconstructed images are the watermarked images which store/hide the binary sequence. These three reconstructed images should be the type of uint8.**

Extraction Procedure:
For each of the three reconstructed (watermarked) images, perform the following operation:
Apply a 3-level "db9" wavelet decomposition. Locate each approximation coefficient using the same raster-scan order as defined in the embedding procedure. For each approximation coefficient ***H'***, if $H'(i) \bmod \beta > \beta/2$, $b'(i) = 1$. Otherwise, $b'(i) = 0$. Use the Matlab display command to show the comparison results in terms of the percentage of the number of the matched bits between each extracted ***b'*** sequence and the original ***b*** sequence for each of the three β values.

Use the Matlab display command to explain the reasons for the difference between the three difference images and analyze the reasons for the possible differences in three extracted ***b'*** sequence.