Homework #5

Due Date: 11:59pm Friday 25 April 2014

In this homework you will write a C++ program that can quantize images using Lloyd-Max and its generalization. Your program will read the image on `stdin` and outputs the resulting image on `stdout`, and get in the required operation on the command line.

Please present a report containing your answers as well as a zip file containing all your code.

1. [2 points] Implement the function `compute_snr()` and `quanize_scalar()` to compute the Signal-to-Noise Ratio (SNR) and to perform uniform scalar quantization.

   The SNR is defined as:

   $$SNR = 10 \log_{10} \frac{\sum_i x_i^2}{\sum_i (x_i - \hat{x}_i)^2}$$

   where x_i is pixel i and $\hat{x}_i$ is the reconstruction value (approximation) for pixel x_i.

   The uniform scalar quantization reconstruction value of pixel x_i is defined as:

   $$\hat{x}_i = \left\lfloor \frac{x_i}{\Delta} \right\rfloor \Delta + \frac{\Delta}{2}$$

   where Δ is the step size and equals 256 / M where M is the number of quantization levels (reconstruction levels).

   Try the function on the small images `msg1.pgm`, `msg2.pgm` and `msg3.pgm` to make sure it's working correctly.

2. [2 points] Try it on the four images (sena, sensin, earth, omaha) using different values for the quantization levels (M = 2, 4, 8, 16, 32), and compute the compression ratio and the SNR. What do you notice? Include sample output quantized images in your report.

3. [3 points] Implement the functions `distance()`, `find_closest()`, `compute_mean()`, and `quantize_kmeans()` to implement k-means quantization. You need to decide on the number of centroids k, the dimensionality of the input vectors n^2 where n is the width/height of the block of pixels. Setting n = 1 means scalar k-means quantization. The distance used is the Euclidean distance:

   $$dist(x, y) = \sum_i (x_i - y_i)^2$$

   where x and y are two L-dimensional vectors.

Note: You will need to round the centroids to the nearest integer (convert from double to integer) before writing into the output quantized image, because all pixels should have values in the range [0, 255].

4. [2 points] Try it on the four images (sena, sensin, earth, omaha) using different values for different values for the parameters:
   1. Try scalar quantization (n = 1), and different values of k = 2, 4, 8, 16, 32. Compare the compression ratio and the SNR with the results of the uniform quantizer in step (2) above.
   2. Try vector quantization (n = 2, 4, 8, 16) with different values of k = 2, 4, 8, 16. Compare the compression ratio and the SNR with the scalar quantization above. What do you notice? Include sample output quantized images in your report.

   Note: when computing the compression ratio for *k*-means, include the storage required to transmit the *codebook* to the receiver in the size of the compressed image.

Command Line

You need to modify the main file `hw05.cpp` to include the required functionality. Your program should be named `hw05`, and should be called as follows:

- To perform scalar uniform quantization on the input image and print out the SNR on the first line followed by the quantized image:

  ```
  ./hw05 -uni_quantize M < input.ppm
  ```

  where *M* is number of quantization levels and the input image is called `input.ppm` and the output is written to `stdout`. For example, to use uniform scalar quantization with 4 levels on `sena.ppm`, you could run:

  ```
  ./hw05 -uni_quantize 4 < sena.ppm
  ```

- To perform *k*-means quantization on the input image and print out the SNR on the first line followed by the quantized image:

  ```
  ./hw05 -km_quantize K N < input.ppm
  ```

  where *K* is number of quantization levels (centroids), *N* is the block size of each pixel where each block is *N*x*N* pixels, and the input image is called `input.ppm` and the output is written to `stdout`. For example, to use vector k-means quantization with 4 centroids on blocks of 8x8 pixels on `sena.ppm`, you could run:

  ```
  ./hw05 -km_quantize 4 8 < sena.ppm
  ```

Instructions

- All code should be implemented in C++ under Linux.
- Please submit your homework in one zip file named as follows: *CMPN206.HW##.FirstName.LastName.zip*, so for example if your name is Mohamed Aly and this is homework #1, then the file name should be *CMPN206.HW01.Mohamed.Aly.zip*.
- Please include all your code and sample output in the zip file, with a README file to explain what you did. Failure to follow these instructions will cause deductions from your grade.
- You are allowed to discuss the problems among yourselves. However, **copying** any part of the code will result a grade of **ZERO**. No exceptions.

Grading

- 9 points: requirements above
- 1 point: submission instructions