

Project Building-Block-2 on Still-Image Coding

Due 10/23/2001 Tuesday 1:59pm EDT

Please hand in a hard-copy write-up with the required answers, observations, and/or inductions. This write-up should be legible and be put in the TA's mailbox by the due time. In the mean time, please create a directory of "enee631" with a subdirectory of "BB2" under your public directory and make them readable (i.e., they should be accessible via <http://your-web-address/enee631/BB2/>). Put the required image results and all of your source codes in this directory by the due time. Include a plain text file with filename "README.txt" to briefly explain what each image/source files included in the directory is. If the images are too large, you can use "gzip" or other lossless techniques to compress the file.

Please do this assignment using C/C++. Please see instructor if you prefer other language. No functions other than those written by you or given with this assignment are allowed to be used without instructor's permission.

➤ DCT-based Compression of Grayscale Lenna and Baboon Images

In BB-1, we have studied block DCT transforms and implemented fast algorithms. You are now asked to compress an image using DCT based techniques similar to JPEG baseline algorithm. The original Lenna and Baboon *pgm* images can be found through the assignment link of the course web page. Your compressed stream need not be compliant with JPEG syntax. With the future part of the project (video codec) in mind, you may want to implement key steps in functions so that you can reuse later.

- (1) Subtract 128 from the pixel values, and apply block-DCT with block size 8x8. Design an automatic scheme to perform bit allocation and quantization that are specifically tuned for each image. The goal is to use as few bits as possible and to introduce distortions that are just unnoticeable. You may determine the relation between such distortion and the overall bit budget empirically [e.g., with respect to coefficient variance]. Optionally, you can use more sophisticated models.
- (2) Use DPCM for DC coefficients; and use zig-zag scan and find run-length pairs for AC coefficients in each block.
- (3) For each image, use computer to find the statistical distribution of the run-length pairs and construct Huffman codes. Use this set of codes to encode the image and obtain a compressed stream. Note that you also need to store the necessary side information that a decoder will have to know. For simplicity, the side information can be stored in a separate file and can be losslessly compressed using existing tools like zip/gzip/LZW.
- (4) Observe the overall compression ratio you can achieve. For each image, use a multiplicative factor of "q" to scale the quantization step size of your quantizer in step-1. Use different q values and include in your report a plot of the resulting relations of MSE distortion vs. bit rate of the compressed image.
- (5) Write a program to decompress the coded image and store the decompressed image in *pgm* format.

Note that you need to design an algorithm automatically optimize the quantization table (bit allocation to each band) and the Huffman table so that they are best suited for the specific image [Hint: estimate the coefficient variance and the probability distributions for that image]. Do not use the default JPEG tables except for the purpose of testing other modules in your codec. You should put your executable program and source codes on web. Include an explanation of your implementation in your report. Also put both on web and in your report the decompressed Lenna and Baboon with distortion just unnoticeable (i.e., $q=1$). Indicate the file size and compression ratio of the corresponding compressed image you have generated. Compare and analyze your compression result for Lenna and Baboon images.