AUDIOVISUAL COMMUNICATIONS

Laboratory Session on the JPEG Standard

Fernando Pereira

The objective of this lab session about the JPEG (Joint Photographic Experts Group) standard is to get the students familiar with many aspects of this image coding standard jointly developed by ISO/IEC and ITU-T. For that purpose, the application "Comunicação de Imagem"[1] will be used which includes among others a module showing how a JPEG baseline (DCT-based) codec works.

The JPEG standard is a generic standard which target several types of applications where photographic image coding (multilevel and color) is important. The JPEG standard includes lossy and lossless coding modes: while the lossy modes are DCT-based, the lossless mode is based on a simple spatial predictor. While the lossless mode is not mandatory for all JPEG codecs, the lossy modes are always present, notably the sequential mode include in the so called baseline process which specifies the basic set of functionalities and tools mandatory present in all JPEG compliant codecs.

This lab guide is organized in three parts corresponding to the three main modules in the image communication/storage system available:

* JPEG image encoder (see Figure 1)
* Transmission channel (with corruption of the coded bitstream)
* JPEG image decoder (see Figure 2)

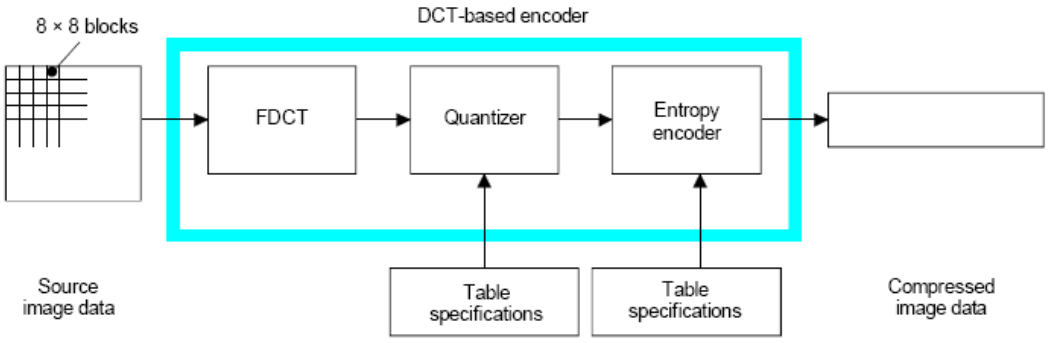

Figure 1 – DCT base JPEG encoder.

[1] The application "Comunicação de Imagem" has been developed by Pedro Fernandes in the context of his 'Trabalho Final de Curso'. I would like to express here my appreciation for his work considering the possibilities this application has opened for the lab sessions of the Audiovisual Communications course.

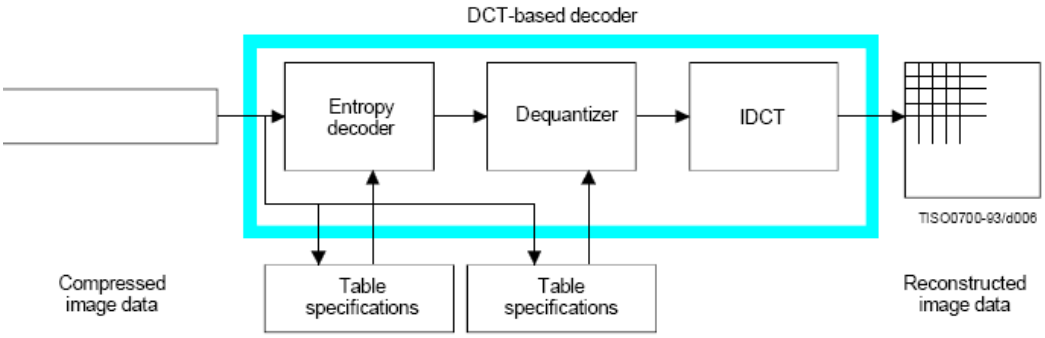

Figure 2– DCT based JPEG decoder.

1. JPEG ENCODER

This first module simulates a JPEG encoder.

1.1 Select Image

This button allows selecting the image to code from the set of available images. The available images include pictures with people, landscapes, buildings, etc. with different characteristics, notably in terms of spatial variation/frequency.

After selecting an image, it will be shown on the screen; then, the user may select the size of the visualization window with the mouse.

1.2 Properties

This button allows determining the coding mode and parameters to encode the selected image. The image name and its spatial resolution are always indicated in the encoder module window. The following encoding processes and parameters may be controlled by the user:

* ***DCT*** – Allows selecting the DCT coefficients used to code each 8×8 samples block. While reducing the number of DCT coefficients to code allows reducing the number of bits spent, also the decoded image quality will be reduced. The available possibilities in terms of DCT coefficients selection are:
  1) No restrictions (all non-null DCT coefficients after quantization are coded);
  2) All DCT coefficients with zig-zag order higher than X are not coded (see Figure 3);
  3) The first X non-null DCT coefficients in the zig-zag order are coded.

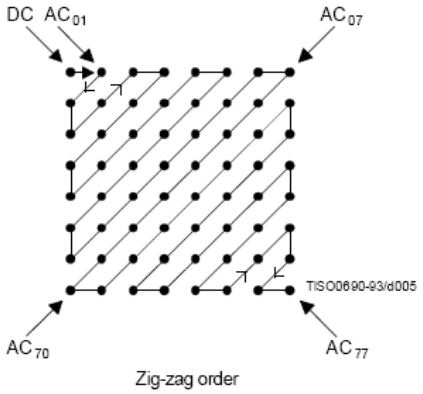

Figure 3 – Zig-zag order.

* ***Quantization*** – Allows establishing the scale factor to apply to some default quantization matrices (which still have to be transmitted); for example, for the luminance, see Figure 4.

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Figure 4 – Luminance quantization matrix.

* ***Synchronization markers –*** Allows specifying the frequency of usage of the synchronization markers targeting the capability to recover the decoding synchronism after transmission errors. The synchronization markers are introduced before the so called Minimum Coded Units (MCU), which correspond to the smallest image units in terms of the interleaving of the image components, e.g., R, G and R. While for a 4:4:4 format – all components with the same spatial resolution – an MCU corresponds to a 8×8 block of each component, this means *(1R, 1G, 1B),* for a 4:2:0 format – chrominances with half the luminance resolution in both directions – an MCU corresponds to 4 luminance blocks for each block of each chrominance, this means *(4Y, 1U, 1V).*

  In terms of the insertion of synchronization markers, the available options are:

  1) No synchronization markers at all for the whole image.
  2) One synchronization marker for each MCU.
  3) One synchronization marker for each row of MCUs.

* ***Encoding mode*** – Allows defining the coding mode to use, notably either the sequential (baseline process) or the progressive modes. While the sequential mode codes the image in a single layer/scanning, the progressive mode codes the image in multiple layers/scannings with successively better quality.

1) **Increasing Precision** – For each successive layer, the DCT coefficients for each block are coded with higher precision, from the most significant to the least significant bits.
2) **Spectral Selection** - For each successive layer, the DCT coefficients for each block correspond to higher frequencies, which means that the first layers code the lowest frequency coefficients and vice-versa.
3) **Combining Spectral Selection with Increasing Precision** - For each successive layer, more DCT coefficients are sent for each block or previous coefficients are sent with higher precision.

1.3 Code Sequence

This button allows coding the selected image in the defined coding conditions and parameters. The user is asked to define the name of the file to store the JPEG bitstream; it is recommended that the user always accepts the file names suggested by the application which consists on the image name with *jpg* as extension. When coding an image, the decoded image will be also shown in the screen.

1.4 Statistics

The application makes available the following metrics: compression factor, and SNR and PSNR, for the luminance and chrominances:

$$SNR = 10\log_{10} \sum_{i=1}^{M}\sum_{j=1}^{N} x^2_{ij} \; / \sum_{i=1}^{M}\sum_{j=1}^{N} (x_{ij} - y_{ij})^2 \quad (dB)$$

where x is the original sample value at position (i,j) and y is the corresponding decoded value. To obtain the PSNR (and not the SNR), the SNR must be computed using the peak value for the signal, this means 255 for samples with 8 bits:

$$PSNR = 10\log_{10} 255^2 \; / \; {}^{1}\!/_{MN} \sum_{i=1}^{M}\sum_{j=1}^{N} (x_{ij} - y_{ij})^2 \quad (dB)$$

2. TRANSMISSION CHANNEL

This module simulates the transmission of a coded bitstream through a non-ideal channel by inserting bit error corruption with two different error patterns: uniform and bursty.

2.1 Select Stream

This button allows selecting the file with the bitstream to corrupt. As mentioned above, the application suggested file name indicates the name of the image coded while its extension indicates the coding method used.

2.2 Properties

This button allows defining the type of bit errors to be introduced in the selected bitstream. The selected file is always mentioned in the dialogue window of the transmission channel module.

* **Uniform Errors –** For uniform bit errors, the user must insert the desired bit error probability which corresponds to the probability than a bit is corrupted independently of all the others.

* **Burst Errors –** For bursty bit errors, the user must insert the following parameters to characterize the desired errors:

  ◊ **Probability of Occurrence of the Start of an Error Burst** – Probability that an error burst starts at a certain bit.

  ◊ **Minimum Length of an Error Burst –** Minimum length of the bit sequence affected by a burst of errors.

  ◊ **Maximum Length of an Error Burst -** Maximum length of the bit sequence affected by a burst of errors.

  The application computed and shows the bit error probability for the bursts understood as the faction of bits corrupted (see note below).

- **Type of Errors in the Corrupted Bitstream**:

  ◊ **Uniform Errors** – The user must indicate in the corresponding check box if uniform errors must be inserted in the next bitstream 'transmission'.

  ◊ **Error Bursts** - The user must indicate in the corresponding check box if bursty errors must be inserted in the next bitstream 'transmission'

  ◊ ***Skip*** – This parameter allows skipping some initial coded lines in the corruption process; this possibility allows visualizing in the same decoded and concealed image, parts which have been corrupted and not corrupted.

Note: *The application computes the bit error probability for bursty errors (understood as the fraction of bits that have been corrupted) assuming that:*

- *After a burst starts, no other burst may start before the previous burst ends.*
- *For each bit within the burst, the bit error probability is 0.5.*

Than, using representative values, it comes:

- *If the minimum and maximum lengths of a burst is 10 and 30 bits, respectively, than the average length of a burst is 20 bits (average number of bits impacted by the burst) assuming all burst lengths are equally probable.*
- *If the probability of a bit is corrupted within a burst is 0.5, than, on average, there are 10 bits corrupted in a burst.*
- *If the probability of starting a burst in a certain bit is 0.0001 and that burst as 10 bits, on average, than there is a probability of 0.0001 of having 10 bits corrupted, this means there is a probability of 0.0001 $\times$ 10 = 0.001 that any bit is corrupted due to a burst of errors.*

2.3 Transmit Bits

This button implements the error corruption with the selected characteristics on the selected file. Before, the user has to define the name of the file where the corrupted bitstream is to be stored. The application always suggests giving the corrupted bitstream file the same name as the non-corrupted bitstream file with the addition of a letter in the file extension signaling the type of errors introduced: "b" for burst errors, "u" for uniform errors and "a" for all, meaning both burst and uniform errors.

2.4 Statistics

After the transmission with corruption, the *Statistics* button allows to obtain much statistical data about that process, notably:

i) Number of uniform errors introduced in the bitstream.

ii) Bit error probability corresponding to the uniform errors (burst errors are excluded).

iii) Number of error bursts introduced in the bitstream.

iv) Average length of the error bursts introduced in the bitstream.

v) Number of error bits introduced with the error bursts.

vi) Bit error probability corresponding to the error bursts (in relation to the total number of bits).

3. JPEG DECODER

This module simulates the decoding process of an image coded with the JPEG standard. The application receives as input a file with the JPEG bitstream and generates the corresponding decoded image.

3.1 Select Stream

This button allows selecting the file with the JPEG bitstream to decode, corrupted or not. Remind the name of the files indicates the name of the image coded while the extension indicates the coding standard used and the type of error corruption inserted.

3.2 Properties

This button allows specifying the decoding and post-processing conditions to use. The selected bitstream is always shown in the decoder module window.

* ***Smoothing filter*** – Allows controlling the usage of the low-pass filter used to reduce the block effect typical of block-based coding. The filter bandwidth is controlled using a slide bar.
* ***Display mode*** – Allows controlling the visualization sequence for the decoded image: i) the decoded image is visualized along the decoding process; ii) the decoded image is visualized all at the same time after the end of the decoding process.

3.3 Decode Stream

This button actions the decoding process for the selected JPEG bitstream. During the decoding process, the decoded image and some error messages may appear in the screen.

3.4 Statistics

The application indicates the number of errors detected along the decoding process.

Audiovisual Communications Course Laboratories

Laboratory Session on the JPEG Standard

Session _____

Nº _____ **Name _____**

Nº _____ **Name _____**

Nº _____ **Name _____**

Try to address the following questions in a way speedy way but always fully understand what you are doing; if you don't understand, ask for help. The most important is to learn …

Whenever configuration details are not provided, use the default parameters available. For each case, analyze the statistics available by pressing the *Statistics* buttons.

1. **Compression Factor for Various Images:** Code the images indicated in the table below with the default parameters and register the compression factor (CF) and the PSNR for the luminance (Y) and chrominances (C_R and C_B). Comment on the compression factors and subjective image quality achieved for the various images considering their content. Do you think the PSNR is a very reliable image quality metric ? Why ?

	CF	PSNR Y	PSNR C_R	PSNR C_B
BALLOON				
BARBARA1				
BARBARA2				
BOARD				
BOATS				
HILL				
HOTEL				

2. **Compression Factor versus Number of DCT Coefficients:** Code the images indicated in the table below **using the DCT coefficients in the first X positions in the zig-zag scanning**. Comment on the variation of the compression factor and the subjective image quality for the various cases and images.

	DCT COEF WITH X=5		DCT COEF WITH X=15		DCT COEF WITH X=30	
	CF	**PSNR Y**	**CF**	**PSNR Y**	**CF**	**PSNR Y**
BALLOON						
BARBARA2						
BOATS						
HOTEL						

3. **Compression Factor versus Quantization Scale Factor:** Code the images indicated in the table below varying the Quantization Scale (QS) factor applied to the quantization matrices. Comment on the variation of the compression factor and the subjective image quality for the various images as a function of QS.

	QS = 50%		QS = 200%		QS = 400%	
	CF	**PSNR Y**	**CF**	**PSNR Y**	**CF**	**PSNR Y**
BALLOON						
BARBARA2						
BOATS						
HOTEL						

4. **Compression Factor versus Synchronization Markers:** Code the images indicated in the table below varying the periodicity of the insertion of synchronization markers. Comment on the variation of the compression factor depending on this periodicity.

	QS = 50%		QS = 400%	
	CF (1 MARKER/MCU ROW)	CF (1 MARKER/MCU)	CF (1 MARKER/MCU ROW)	CF (1 MARKER/MCU)
BALLOON				
BARBARA2				
BOATS				
HOTEL				

5. **Progressive Mode:** Code the images indicated in the table above using the possibilities available in terms of progressive (quality scalability) coding, notably spectral selection, and growing precision. Comment on the evolution of the image quality along the full decoding process for the various progressive modes; visualize the decoding process using the option *Show scan lines while decoding* in the *Display Mode* menu at decoder.

6. **Low Pass Post-Process Filtering:** Code the images BALLOON and BARBARA1 with QS = 500% and decode them varying the bandwidth of the post-processing low pass filter. Comment on the variation of the decoded images subjective image quality depending on the used low pass bandwidth.

7. **Decoding after Corruption with Uniform Errors:** Code the image BALLOON with the default coding parameters, varying the insertion periodicity of the synchronization markers. Simulate the transmission of the bitstreams through a channel only with uniform errors corruption, with different bit error probabilities, e.g., $P_b = 0.0001$ and $P_b = 0.001$. Decode the corrupted bitstreams and comment the subjective image quality of the decoded images; try to insert again the errors if the application goes in 'error'.

8. **Decoding after Corruption with Burst Errors:** Repeat the previous question, now for burst errors, e.g., $P_B = 0.0001$, $L_{min} = 10$ and $L_{max} = 50$. Decode the corrupted bitstreams and comment on the subjective impact of the errors on the decoded images, comparing the uniform and burst error cases; take into account the total number of erred bits for each case (use the *Statistics* button in the Transmission Channel module). Comment on the influence on the subjective quality of the insertion periodicity of the synchronization markers when the transmission channel is not perfect.