TÉCNICO LISBOA

AUDIOVISUAL COMMUNICATION

Laboratory Session: Facsimile

Fernando Pereira

The objective of this lab session about the facsimile is to get the students familiar with the main aspects of a facsimile transmission which target the efficient transmission of bilevel images. For that purpose, the application "Audio and Video Communication Simulation System"[1] will be used which includes among others a module with a facsimile communication system.

This lab guide is organized in three parts corresponding to the three main modules in the facsimile communication system available (see Figure 1):

* Facsimile images encoder
* Transmission channel (with corruption of the coded bitstream)
* Facsimile bitstream decoder

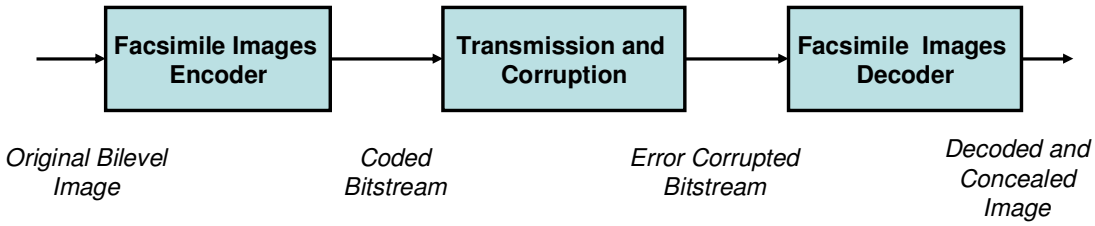

Figure 1 – Simplified architecture of the facsimile communication system.

1. FACSIMILE IMAGES ENCODER

This first module simulates the image encoding process for Group 3 facsimiles, targeting the telephone network, and Group 4 facsimiles, targeting data networks. The mandatory image codecs for each of the facsimile classes are different, considering the type of communication channels they target.

1.1 Select Image

This button allows selecting the image to encode from the eight CCITT (now ITU-T) test images. These test images include letters, electronic schemes, accounting documents, book pages, etc, all with

[1] The application "Audio and Video Communication Simulation System" has been developed by Pedro Fernandes in the context of his M.Sc. Thesis. 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.

rather different statistical characteristics regarding the black and while data present and their redundancy.

After selecting an image in the 'Fax images' directory, the image will appear in the screen. Using the mouse, the user may than control the size of the image in the screen, zoom in and zoom out parts of the image as well as define the zone of the image in visualization using the lateral slide bars.

1.2 Properties

This button allows selecting the coding method for the image just selected or already previously selected. The selected image is always mentioned in the encoder dialogue window.

The available coding methods are:

- **GROUP 3 (mandatory method)** – **Modified Huffman Method (MHM)** – This coding method exploits the horizontal redundancy (along the lines) but not the vertical redundancy (between the lines); this means one-dimensional coding is performed. Each line is ended with a *End Of Line* (EOL – 000 000 000 001) codeword; the end of the page is signaled with a *End Of Page* (EOP) codeword which corresponds to 6 EOLs.
- **GROUP 3 optional extensions**

  2.1) Modified READ Method (MRM) – The Modified READ (relative addressing) Method uses bidimensional coding since it exploits both the horizontal and vertical redundancies in the facsimile image. To control the propagation of errors between decoded lines, this method codes some lines one-dimensionally, periodically or not, using the MHM coding method. With this purpose, the application asks the user to introduce the value for the *k* parameter which determines the periodicity for the MHM coded lines among the MRM coded lines which will act as error propagation 'brakes'. A value *k* means here than periodically *k-1* (out of *k*) lines and coded exploiting both the horizontal and vertical redundancy while the remaining periodic line is codec only exploiting the horizontal redundancy. Usually, k=2 for low resolution (3.85 lines/mm) images and k=4 for high resolution (7.7 lines/mm) images. Each coded line ends with an EOL, followed by a *coding method label* bit which if a '1' indicates the next line will be MHM coded and if a '0'indicates the next line will be MRM coded.

  For the MRM, the coding is based on counting the number of successive white and black samples, the so-called *runs,* which are after entropy coded.

  2.2.) Modified Modified READ Method (MMRM) – The Modified Modified READ Method is similar to the MRM method using k=∞ which means there are only bidimensionally (MRM) coded lines with the exception of the first line. Since this method is typically used when it can be assumed that the channel is virtually error free, there is no need to 'take care' of errors and thus neither EOLs nor *coding method label* bits are used. EOP corresponds to 6 EOLs followed by the coding method label bit with '1'.
- **GROUP 4 (mandatory method) - Modified Modified READ Method (MMRM)** – For Group 4 facsimiles, the coding method is always the MMRM since it is assumed that there is no need to 'take care' of errors since the channel is virtually error free. The errors are dealt with in another way, e.g. in the lower OSI model layers of the Group 4 facsimile terminal using channel coding tools. In this case, the EOP corresponds to 2 EOLs without *coding method label* bits.

1.3 Code Image (button ►)

This button allows coding the selected image with the selected bilevel image coding method. A file name for the coded bitstream is asked to the user to store the coded bits. The application always suggests a name for this file corresponding to the image name with an extension corresponding to the coding method:

- '3h' for Group 3 MHM
- '3r' for Group 3 MRM
- '3m' for Group 3 MMRM
- '4m' for Group 4

During the coding process, the application will show in the right of the screen two charts with two types of compression factors[2] for the white, the black and the full set of pixels in the image (see Figure 2):

- **Evolution of a 'local compression factor'** corresponding to the compression factor for a 8 lines sliding window along the image lines (from top to down); this local compression factor expresses the variation of the redundancy along the image, with higher compression factors for the more redundant areas, such as uniform areas, and vice-versa.
- **Evolution of a 'global compression factor'** corresponding to a growing window with one side fixed at the first line and the other side varying from the first line to the last line in the image; this global compression factor expresses the accumulated variation of the redundancy along the image until finally covering the full image.

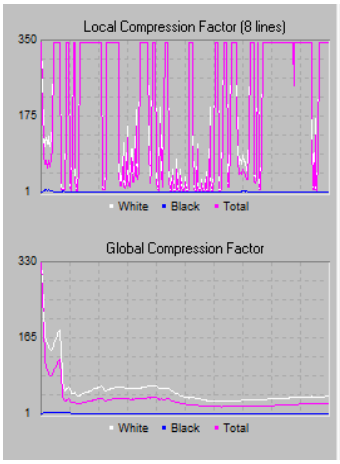

Figure 2 – Local and global compression factor charts.

1.4 Statistics

After the coding process, the *Statistics* button allows to get much statistical data about the performed coding process, notably:

- For each tone, black and white:

[2] The compression factor is always a ratio between the number of bits spent with PCM coding and the number of bits spent with a selected source coding method, thus expressing the compression power of the coding method; in the case of case of facsimile, the codecs are always lossless which means there is no quality loss associated to the coding process.

i) Total number of samples/pixels for each tone.

ii) Total number of bits for the codewords used to code the samples of a certain tone.

iii) Compression factors for the coding of the samples of a certain tone (ratio between the two previous values).

iv) Average length of the runs for each tone (without considering the division of the runs in *make-up* and *terminating* codewords and average length of the partial runs for each tone (considering now the division in *make-up* and *terminating* codewords).

v) Number of samples/pixels coded with *make-up* and *terminating* codewords for the MHM. These samples correspond to MHM coding or MRM horizontal coding mode.

vi) Number of *make-up* and *terminating* codewords for the samples in v).

vii) Number of samples coded with the MRM vertical and pass coding modes when MRM is used.

viii) Number of coding symbols corresponding to the samples in vii).

- For the overall image:

i) Total number of samples/pixels in the image.

ii) Total number of code bits; this total is higher than the sum of the total bits used to code the black and white runs because it also includes the bits corresponding to the EOLs, EOP and *coding method label* bits.

iii) Global compression factor corresponding to ratio between the total number of samples and the total number of code bits.

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 be corrupted. As mentioned above, the application suggested file name indicates the name of the coded image 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 input file/stream is always indicated in the dialogue window of the transmission channel module.

- **Uniform Errors** – For uniform bit errors, the user must insert the desired bit error probability corresponding to the probability that 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 computes and shows the bit error probability for the bursts understood as the fraction 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 a certain number of initial image 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.*

Then, 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 that all burst lengths are equally probable.*
- *If the probability of a bit is corrupted within a burst is 0.5, then, 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, there is then, on average, 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 error corruption of the selected bitstream with the selected channel characteristics. For this, 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 (output) bitstream file the same name as the non-corrupted (input) 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
- 'a' for both burst and uniform errors

2.4 Statistics

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

1. Number of uniform errors introduced in the bitstream.
2. Bit error probability corresponding to the uniform errors (burst errors are excluded).
3. Number of error bursts introduced in the bitstream.
4. Average length of the error bursts introduced in the bitstream.
5. Number of error bits introduced with the error bursts.
6. Bit error probability corresponding to the error bursts (in relation to the total number of bits).

3. FACSIMILE IMAGES DECODER

The last facsimile communication module simulates the decoding and error concealment processes for Group 3 and Group 4 facsimiles.

3.1 Select Stream

This button allows selecting the file with the corrupted bitstream to be decoded and error concealed. If the application suggestions in terms of file names have been accepted by the user, the file name indicates the name of the coded image while the file extension indicates the coding method and the type of channel errors inserted.

3.2 Properties

Considering that decoding bitstreams which have been channel corrupted leads inevitably to information losses, and thus to a reduction of the image quality, it is very important to minimize the negative subjective impact associated to the errors in the decoded and error concealed image.

The maximization of the decoded image quality, through the minimization of the negative impact of the error corruption, is a decoder task which may be performed in three main ways:

1. **Retransmission** - Asking the encoder to retransmit at least some of the coded data; this solution implies delay and having a feedback channel available.
2. **Channel coding** – Performing channel decoding to correct at least some of the bitstream errors; this solution implies spending bits of channel coding, thus reducing the overall compression factor.
3. **Error Concealment** – Post-processing the decoded image to 'hide' in the best way the error corrupted image areas; this only implies some decoded complexity to reduce the quality impact of the errors, never correcting the bitstream errors themselves.

In a situation where it is not possible to request the sender to send again the coded data and no channel coding is used no avoid spending bits, the decoder is not able to correct the corrupted bits and should than exploit error concealment methods which are applied after the decoding of the image. These methods are not standard, do not require the transmission of any additional data; moreover they may be more or less intelligent and complex depending on the manufacturers which means that a higher 'complexity investment' typically results in a 'higher concealed quality'. This type of error

processing where the decoder tries to 'hide' the effect of channel errors based on the decoded image is more important for the MRM coding method since it suffers from error propagation along the page due to the exploitation of the vertical redundancy (in addition to the horizontal redundancy).

The error concealment options available in the application at hand to minimize the negative subjective impact of the transmission errors are:

* ***Print White*** **(PW)** – Whenever a line has a decoding error, that line is fully set to white in the concealed image as well as all the next lines until a line is decoded in good conditions, this means a one-dimensionally coded line is decoded without errors.

* ***Print Previous Line*** **(PPL)** - Whenever a line has a decoding error, that line and all the next lines until a line is decoded in good conditions is set the same as the previous line.

* ***Print Previous Line/White*** **(PLW)** - Whenever a line has a decoding error, that line is set the same as the previous line and the following lines are fully set to white until a line is decoded in good conditions, this means a one-dimensionally coded line is decoded without errors. For k=1, PPL and PLW give the same results.

* ***Normal Decode/Previous Line*** **(NDPL)** – Whenever a line has a decoding error, the well decoded part of the line is used and the remaining of the line is set the same as the previous line. The process proceeds with this line becoming the reference for the decoding of the next line until a line is decoded in good conditions, this means a one-dimensionally coded line is decoded without errors.

Before decoding a corrupted bitstream in this application, the user has to select one of the error concealment methods above in order the decoder knows what method to use to 'hide' the effect of the transmission errors. All the methods above can only recover the decoding synchronism when a one-dimensionally decoded line is received without errors (which may be next one-dimensionally coded line or not depending on where the errors lay). For this process to work, the bitstream has to include *End of Line* (EOLs) codewords for the Group 3 facsimiles. For k=∞, this means when no one-dimensionally coded lines are present beside the first one, these methods are rather useless since recovering the decoding synchronism does not happen before the end of the page. In these conditions, the facsimile system must be relying on some alternative solution to deal with the errors, notably by data retransmission using an adequate protocol or channel coding (which may still leave residual errors which have to be dealt with by the source decoder). In that case, the image degradation is reduced not by using the 'hiding' methods presented above but rather by correcting the errors themselves and decoding with correct data.

3.3 Decode Stream (button ►)

This button implements the decoding of the selected corrupted bitstream using the error concealment method selected by the user. The application always suggests a file name for the decoded and concealed image, in this case adding to the file extension some letters identifying the error concealment method used, e.g. PPL for the Print Previous Line method.

During the decoding process, the application shows in the screen the detected decoding problems as well as some warnings, notably corresponding to:

* **Syntactic Errors –** The received codeword does not exist in the code tables, meaning that an 'impossible' codeword has been received and, thus, an error must have happened; in this case, the decoder has a location for the syntactic error, although the erred bit typically happens somewhere before the decoder gets 'syntactically lost'.

* **Semantic Errors –** The decoded line has more or less than the expected number of samples, e.g. 1728 for an A4 page, meaning that a semantic error has occurred. In this case, there are no syntactic errors since the initially 'correct' codewords got transformed into other 'existing' codewords, thus syntactically valid, but semantically (this means in terms of meaning) the bitstream is not valid anymore. In this case, it is basically impossible to precisely know where the error is located using only the information from the line under decoding; maybe it is possible to know a bit more by comparing this line with the previously decoded line, if they are very similar.

* **Warning –** Indicates the correctly decoded lines as well as the lines which *coding method* label bit have a different value from the one expected based on the parameter *k*; this situation does not imply any errors, since the decoder always uses the label bit as, for some reason, the strict MHM periodicity may not have been followed.

After decoding and error concealment, the image is shown on the screen. Moreover, the user is given the possibility to compare the final (decoded and concealed) image with the original image which has been initially coded, getting the percentage of samples/pixels different (and thus wrong) in the two images. This objective quality metric may be rather misleading since the subjective quality differences between the several error concealment methods are typically much higher than this metric may indicate.

3.4 Statistics

After decoding and concealment, the *Statistics* button allows obtaining much statistical data about the decoding process, notably:

i) Number of lines correctly decoded.

ii) Number of *coding method label* bits different from what was expected (considering the parameter *k*).

iii) Number of synchronism losses due to syntactic errors ('impossible codewords').

iv) Number of lines decoded with more or less samples than required, typically 1728 for an A4 page.

v) Number of incorrect samples in the final (decoded and concealed) image, this means white samples than became black and vice-versa.

vi) Percentage of the decoded and concealed image with incorrect pixels, this means white samples than became black and vice-versa.

Audiovisual Communication Course Laboratories

Laboratory Session: Facsimile

Date _____ Day of the week

_____

Nº _____ Name

_____

Nº _____ Name

_____

Nº _____ Name

_____

Try addressing the following questions in a speedy way but while always fully understanding 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* button.

1. **Global Compression Factors:** Code the images in the following table with the Group 3 MHM mandatory coding method, the Group 3 MRM optional coding method and the Group 4 MMRM mandatory coding method and report the global, partial white and partial black pixels compression factors. ***Comment on*** i) the variation of the global compression factors for each image depending on the used coding method; and ii) the variation of the global compression factors for each coding method depending on the image content.

IMAGE	GROUP 3 (MHM)			GROUP 3 EXTENSION (MRM, K=4)			GROUP 4 (MMRM)		
	WHITE	BLACK	GLOBAL	WHITE	BLACK	GLOBAL	WHITE	BLACK	GLOBAL
CCITT 1									
CCITT 4									
CCITT 6									
CCITT 8									

_____

_____

_____

_____

_____

_____

_____

_____

_____

2. **Local Compression Factors:** Regarding the CCITT 8 image, coded with the Group 3 and Group 4 mandatory coding methods (MHM and MMRM): i) ***comment*** the evolution (along the page) of the local compression factor as a function of the image content; and ii) ***identify and explain*** some differences between the local compression factors curves for the MHM and MMRM methods.

3. **MRM Coding with Periodic MHM Coding:** Code the images in the following table with Group 3 optional coding method (MRM) for the *k* values indicated and report the obtained global compression factors. Based on the results, ***explain*** i) the influence of the *k* parameter on the compression factor; and ii) the benefits or not of using higher *k* values, considering the image content.

IMAGE	GROUP 3 EXTENSION								
	K = 2			*K* = 4			*K* = 16		
	WHITE	BLACK	GLOBAL	WHITE	BLACK	GLOBAL	WHITE	BLACK	GLOBAL
CCITT 1									
CCITT 7									

4. **MHM Decoding with Uniform Transmission Errors:** Code the CCITT 5 and CCITT 8 images with the MHM coding method and transmit the resulting bitstream through a channel with **uniform error corruption** with bit error probability $P_b = 0,001$. Decode the corrupted bitstream using all the available error concealment methods, this means *Print White* (PW), *Print Previous Line/White* (PLW), *Print Previous Line* (PPL) and *Normal Decode/Previous Line* (NDPL). Observe the error concealed images and ***comment on*** their subjective impact (cleanness and readability) also considering the computed percentage of incorrect pixels (whites becoming blacks and vice-versa), the image content characteristics and the error concealment method used. Which main conclusions do you take ?

5. **MRM Decoding with Uniform Transmission Errors:** Repeat the previous question, using now MRM coding with *k=4*. ***Explain*** the benefits and disadvantages of using each of the available error concealment techniques in this situation; if needed, also test with other images.

6. **Uniform versus Bursty Error Patterns:** Code the CCITT 3 image with the MRM coding method with k=4 and transmit the bitstream in two different channels: i) with **uniform errors** with $P_b = 0,001$; and ii) with **bursty errors** with $P_B = 0,0001$, $L_{\min} = 10$ e $L_{\max} = 50$. After, decode the two bitstreams with the *Normal Decode/Previous Line* (NDPL) error concealment method. Considering the total number of corrupted bits introduced in each case (use the *Statistics* button to know the number of corrupted bits), ***comment on*** the subjective impact (cleanness and readability) on the concealed images of the two types of error distributions. *If needed, change the channel corruption parameters to make the total number of corrupted bits for the two corruption cases (uniform and burst) the same, meaning that only the distribution of the errors in the page is different.*

7. **Lossless Coding:** ***Describe a simple experiment*** in the application at hand which is able to demonstrate that all facsimile coding methods tested implement *lossless coding*.