ELEN-0060: Information and Coding Theory

Project 2 - Source coding, data compression and channel coding

Maxime Goffart
180521

Olivier Joris
182113

Academic year 2021 - 2022

1 Channel coding

1.1 Question 16

In order to implement a function to read and display the given image, we used the methods `imread` and `imshow` provided by OpenCV.

1.2 Question 17

To encode the image signal, we used a fixed-length binary code of 8 bits. We have chosen 8 bits because there are 256 (from 0 to 255) possible values, so we need $\lceil log_2(256) \rceil = 8$.
The code is the binary representation of the grayscale value of each pixel.

1.3 Question 18

By simulating the channel effect on the binary signal of the image, we get the following image:

Figure 1: Image after simulating the channel effect

As we can see in the picture, after simulating the channel effect, there are a lot of small dots[1] that are pixels with different grayscale values compared to their very close neighbors.
This is due to the fact that we are simulating a potential loss bit by bit and we are not using any sort of redundancy. Thus, if one of the most significant bits is modified, it completely changes the grayscale value for the pixel.

1.4 Question 19

In order to compute the Hamming(7,4) code for the binary image signal, we need to add 3 redundancy bits for every 4 bits. The 3 redundancy bits are:

- Bit 1 = $(bit0 + bit1 + bit2)\ mod\ 2$
- Bit 2 = $(bit1 + bit2 + bit3)\ mod\ 2$

[1] Zoom in the image to see them better.

- Bit 3 = $(bit0 + bit2 + bit3)\ mod\ 2$

where bit 0, bit 1, bit 2, and bit 3 are, respectively, the first, second, third, and fourth bits for which we want to add redundancy.
By applying this principle on each block of 4 bits from the binary image signal, we get the Hamming(7,4) code for the entire binary image signal.

1.5 Question 20

If we decode the binary image signal with redundancy, we get the following image:

Figure 2: Image after simulating the channel effect with redundancy

As we can see in the picture, we observe less dots compared to *question 18*. This is beacause we used redundancy with the Hamming(7,4) code. Yet, we can still observe some dots. This is because Hamming(7,4) code can correct only one error per chunk of bits. In some cases, we might have more than one error per chunk thus the original source is not recoverable.

In order to decode the binary signal with redundancy, we used the syndrom decoding technique, as in the exercise sessions.
To apply this technique, we consider that the source bits were correctly transmitted through the channel. Then, we recompute the 3 parity bits based on the received 4 source bits. Afterwards, we apply a bitwise and between the received parity bits and the re-computed parity bits.
Based on the result of the bitwise operation, multiple cases are possible:

- If 2 or 3 bits are 1 in the result of the bitwise operation, we can deduce that one of the source bit has been incorrectly transmitted. To recover from the error, we can procede the following way:
  - If bits 0 and 1 of the bitwise and are 1, we need to flip the second source bit.
  - If bits 1 and 2 of the bitwise and are 1, we need to flip the fourth source bit.
  - If bits 0 and 2 of the bitwise and are 1, we need to flip the first source bit.
  - If all the bits of the bitwise and are 1, we need to flip the third source bit.

- If 1 bit is 1 in the result of the bitwise operation, we can deduce that one of the parity bit has been incorrectly transmitted.
- It is possible that 2 or more bits were transmitted incorrectly. In that case, we cannot recover from the error.