Multimedia Security and Privacy

TP2: Basic Cryptography and Watermarking

Prof. Sviatoslav Voloshynovskiy,
Denis Ullmann <denis.ullmann@unige.ch>.

Stochastic Information Processing Group

March 5, 2020

UNIVERSITÉ DE GENÈVE
FACULTÉ DES SCIENCES
Département d'informatique

SIP stochastic information processing

Submission

Please archive your report and codes in "Name_Surname.zip" (replace "Name" and "Surname" with your real name), and upload to "`Assignments/TP2: Basic Cryptography and Watermarking`" on `https://chamilo.unige.ch` before Wednesday, March 18 2020, 23:59 PM. Note, **the assessment is mainly based on your report, which should include your answers to all questions and the experimental results**.

1 Encryption

This exercise will show a fundamental difference between classical cryptography techniques and (robust) watermarking.

Exercise 1

1. Read in the image `liftingbody.png`
2. Make a permutation matrix that will map each pixel of the source image to a new 2D position. Store this matrix and use it to permute the input image to a new (permuted) output image. Show both on the screen.
3. Determine and show the histograms of both the original source image and the permuted image. Explain the results.

Exercise 2

Write a function that models block loss to an image in transmission. Your function should take an input image and set a random $[N \times M]$ block of the image to zero. See Figure **??**

(a) original **(b)** Block loss

Figure 1 – Block loss example

Exercise 3

1. Read in the source image `liftingbody.png`
2. Make a permuted version using your permutation matrix
3. Take the permuted image and distorted using the block-loss function.

4. Permute the image back to the original and explain the result.

Exercise 4

1. Read in the source image `liftingbody.png`
2. Generate a new noisy image whose values $\{-1, 1\}$ are uniformly distributed
3. Add the noisy image to the original source image.
4. Display both images and their histograms. Explain what you see.
5. Repeat the step with new noisy images with values $\{-5, 5\}$, $\{-10, 10\}$ and $\{-15, 15\}$
6. Add these 3 noisy images to the original image individually and determine the PSNR value each time. Plot all PSNR value vs. noise value and explain the result

2 Classical Cryptography

In this exercise you will expand a small existing demo to show that classical cryptography fails the moment the encrypted image suffers from any kind of noise. On `Chamilo` is a zip file called `AES.zip`. Download this file, expand it, and set the matlab working directory to the now expanded `AES` directory. These files in this directory together implement the `AES` cipher.

- Locate the function `aes_demo.m` and run it to confirm that everything works.
- Locate the function `aes_demo2.m` and run it. It reads in an image and then encrypts and decrypts it.
- Modify `aes_demo2.m` such that the encrypted image suffers from a single bit error, i.e., take a single pixel from the encrypted image and overwrite it with a random value. Explain the results.
- Modify `aes_demo2.m` such that the encrypted image suffers from Additive White Gaussian Noise. This can be done by generating an identically sized matrix with `randn` and adding it to the encrypted image. Explain the results.

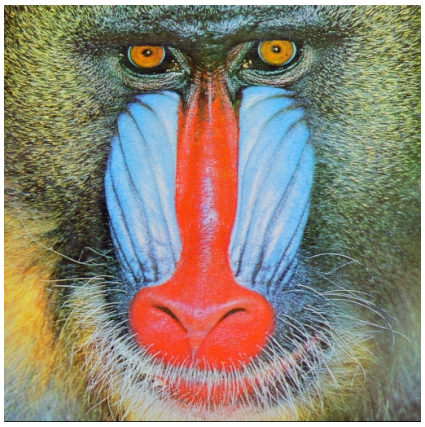
3 Basic Data Hiding

In this exercise you will implement a simple watermarking technique based on the most significant bits (MSB) and the least significant bits (LSB) of a data type. In this scheme, you will hide a *secret* image, the baboon in Figure **??** by writing it in to the so called *cover* image; Lena. The final combination is the so called *stego* image.

You will hide the baboon image into the cover image by only writing the most significant bits of the secret image into the least significant bit positions of the cover image forming the new stego image. Use the bit plane structure in Table **??**.

- The Lena and Baboon image can be found on `Chamilo`.
- C denotes the Cover image, S denotes the secret image
- $C_{r,8}$ denotes the value of Cover image C, in the red channel r, on bit position 8.
- Find out if the system you are working on is big endian or small endian.

(a) Lena



(b) Baboon

Figure 2 – Watermarking images. Lena will serve as a *cover* image to hide the *secret* Baboon image in.

Exercise 1

- Implement two functions. The first function should insert the secret image S in to cover image C following the scheme in Table **??** producing the stego image. The second function should recover the secret image S out of the stego image.
- Visualize the difference between the original cover image and the stego image. Explain the obtained results.

	MSB							**LSB**
Red	$C_{r,8}$	$C_{r,7}$	$C_{r,6}$	$C_{r,5}$	$C_{r,4}$	$S_{r,8}$	$S_{r,7}$	$S_{r,6}$
Green	$C_{g,8}$	$C_{g,7}$	$C_{g,6}$	$C_{g,5}$	$C_{g,4}$	$C_{g,3}$	$S_{b,8}$	$S_{b,7}$
Blue	$C_{b,8}$	$C_{b,7}$	$C_{b,6}$	$S_{g,8}$	$S_{g,7}$	$S_{g,6}$	$S_{b,6}$	$S_{b,5}$

Table 1 – Bit plane layout for the new composed stego image where secret image S is inserted in cover image C. $C_{r,8}$ denotes the value of Cover image C, in the red channel r, on bit position 8.

Hints

- Use bitmasks to access and work with bit values.