Image Processing – 67829 – Exercise 2

Due Date: 20.11 at 23:59

Version 1.1 - Last Update 6.11.2025

In this exercise, you will create and detect audio watermarks. An audio watermark is a unique electronic identifier embedded in an audio signal, typically used to indicate copyright ownership. Watermarks are usually inaudible to human listeners, ensuring the audio quality remains unaffected. Throughout the exercise, you are expected to analyze the audio files and listen to them.

1 Tasks

1.1 Creating watermarks

In Moodle, under "Exercise 2" -> "Exercise Inputs" -> "Task 1", you are provided with an audio file. Your task is create a "good" and "bad" watermark and add them to the audio file (i.e., add each one independently, not both at the same time).

Note: For this exercise, our definition of good and bad watermarks is very broad. In this context, a good watermark is one that is inaudible to the average human listener (not just to you, as hearing abilities vary) but can still be reliably detected by an algorithm. If the watermark is perceptible to the average human ear, it is not a good watermark!

Note: You are not expected to create complex watermarks but should explore different options for how, where, and what type of watermark to apply.

Note: For all audio files, make sure you use the correct sampling rate when loading them. Don't assume the default sampling rate of the library you use is the correct one.

1.2 Classifying audio based on watermarks

In Moodle, under "Exercise 2" -> "Exercise Inputs" -> "Task 2", you are provided with 9 audio files, each containing one of 3 possible watermarks. You may assume that each watermark is created by a very simple function. Your task is to group the audio files correctly according to their watermarks and identify the function describing each watermark.

Note: Your solution does not need to generalize to other audio files or other watermarks.

1.3 Determining audio speedup method

In Moodle, under ”Exercise 2” -> ”Exercise Inputs” -> ”Task 3” , you are given 2 additional audio files. The *same* speedup/slowdown factor of x was applied to both audio files. One of the files was sped up in the signal (time) domain and the other in the frequency domain. Your task is to determine which file was sped up using each method and find the speed up ratio (x).

Note: Both files have been watermarked with the same watermark. While you are not required to use the watermarks in your solution, they may be helpful.

Note: Don’t use the length of the audio to determine the speedup ratio, base your decision on actual empirical evidence.

2 Submission

Submission instructions are given in the ”Submissions Guidelines” document published on the course web page (here). Please read and follow them carefully. Any updates to those guidelines will be posted in the news forum, so be sure you follow the forum.

You must submit your code but there is no API you need to follow and there are no presubmissions.

2.1 Report Guidelines

In addition to the code, you should submit a report describing your solution. The report must follow the following structure and address the topics below. We provide an English and Hebrew Google Docs template (you need to copy it to use it). In case you choose not to use it, please maintain a similar structure (font size, same sections, same number of figures, same number of pages, etc.), in particular, the report should be no longer than 5 pages and include the following sections and topics:

1. Introduction

   (a) In your own words, state the goal of the exercise and what were the main techniques (i.e., an idea or concept you’ve learned in class, not a technical tool like numpy) you’ve used to solve it.

2. Adding Watermarks

   (a) Create 1 ”good” and 1 ”bad” watermarks.

   (b) Clearly describe your watermarks and the method used. Support your choices with figures.

   (c) Compare your design choices with other options you tried that were less effective.

   (d) Explain why the bad option was less successful and its impact on the audio signal.

(e) Explain how would you design a watermark for images and what things should be taken into account when designing such watermarks. **Note: You are not expected to implement this, just to explain.**

3. Classifying watermarks

    (a) Describe your overall approach and steps for detecting the watermarks.

    (b) Group and present the different audio signals, displaying their watermarks.

    (c) Use all 3 audio files of each group to extract and display the watermark function. **Note: You are not expected to recover the exact watermark, your solution may be noisy, but it should resemble the watermark.**

    (d) Implementation

    - Describe your implementation of the classification algorithm and how you identified each watermark's function.
    - Specify the parts that you implemented from scratch and those that you've used functionality from an existing library. What libraries did you use and why did you choose these?
    - Describe and justify any necessary hyper-parameters, thresholds, or other choices you made.
    - Discuss any challenges faced during implementation and how they were addressed.

    (e) Include figures and visualizations that support your choices and results.

    (f) Explain how would you remove the watermark and what needs to be taken into account when doing so in order to ensure minimal change to the audio. **Note: You are not expected to implement this, just to explain in words.**

4. Determining the speedup method

    (a) Clearly describe the difference between the two speedup methods, both from a perceptual perspective (i.e., how they sound) and a technical perspective (i.e., what each method does).

    (b) Identify the speedup method used for each signal and determine the speedup factor.

    (c) Explain your decision and support your claims with figures.

5. Conclusion

    (a) Summarize your key findings and insights.

Your final submission should be:

1. A PDF named ”ex2.pdf”.
2. A tar file a python file named ”ex2.py”, and a requirements.txt file with your dependencies. To create a tar file you can run the following command: `tar -cvf ex2.tar ex2.py requirements.txt`

Note: The PDF and tar should be submitted to the respective submission in the Moodle.

3 Grading

Your exercise will be graded based on a manual inspection of your report (and code). As mentioned above, there will be no presubmission tests and no automatic tests.