TP I – Tatouage par quantification

Gaëtan Le Guelvouit

L'objectif de cette première session de travaux pratiques est de se familiariser avec la manipulation d'images monochromes et RVB, puis de développer votre premier algorithme de tatouage. Les sources sont en C++, placées dans le répertoire `src` (un fichier `Makefile` comprend tous les paramètres de compilation). Plusieurs classes – très simples – permettent de lire et d'écrire des images PGM et PPM (fichiers `image.*`), de faire une transformée DCT en blocs (fichiers `dct.*`), et de simuler des dégradations classiques (compression, ajout de bruit, etc.). Le répertoire `data` contient quelques images de test.

`https://tinyurl.com/cswt2023`

1 Tatouage par LSB

1.1 Compréhension du code

Nous utilisons des notations similaires à celles introduites dans le cours. Les fichiers `lsb_write.cpp` et `lsb_read.cpp` permettent respectivement de tatouer et de lire le tatouage via une algorithme de substitution des bits de poids faible.

1. Analyser le source `lsb_write.cpp`. Quel est le type du document hôte ?
2. Sous quelle forme est le message ?
3. Que ce passe-t-il si le message est trop court par rapport au document hôte ?

1.2 Lecture de la marque

À vous de compléter `lsb_read.cpp` pour être en mesure de lire les tatouages LSB produits par le programme précédent.

1. À quoi sert la ligne `mtsrand(xxx)` ? Compléter avec la bonne valeur.
2. Programmer la boucle de lecture.
3. Tester votre proposition sur l'image `data/lena_lsb.ppm`.

2 Tatouage par quantification

On passe maintenant au tatouage par quantification, dont les squelettes sont les fichiers `qt_read.cpp` et `qt_write.cpp`.

2.1 Premier essai

1. Quel est le format du message et du document hôte ?
2. à quoi servent ces trois boucles imbriquées au début du programme ? Sur quelle structure et quelle variable allons-nous travailler pour tatouer ?
3. Compléter le programme de tatouage.
4. Compléter le programme de lecture et tester l'enchainement tatouage-lecture sur une image.

2.2 Améliorations

1. L'erreur quadratique moyenne introduite par la quantification scalaire d'un signal continu est estimée par EQM $\simeq \Delta^2/12$. À partir de cet indice, programmer le calcul de Δ en fonction d'une valeur de PSNR quelconque.
2. Que doit-on modifier dans le programme de lecture pour que tout fonctionne ?
3. Tester pour 40 dB sur une image. Contrôler la distortion avec le binaire `./psnr` que vous avez compilé.