projet

Stéganographie

T. NSI

Problématique

La stéganographie, ou art de la dissimulation, est exploitable dans de nombreux domaines, notamment dans le « watermarking », ensemble de techniques permettant de « tatouer » un fichier numérique.

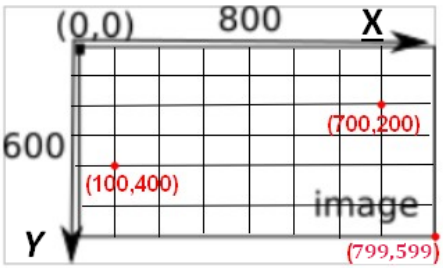
Un microstock souhaite introduire des informations utiles à la gestion des droits d'auteur dans les photos qu'il propose à la vente, à l'aide d'un tatouage invisible.

Votre projet

Votre chef de projet vous demande de :

1. présenter au client les concepts nécessaires à la compréhension de la méthode de tatouage.
2. développer un programme Python capable de coder et décoder un message invisible tatoué dans une image :
   - Eleve1 : codage d'un message (exemple message.txt) dans une image (james_bond.png)
   - Eleve2 : décodage du message et affichage du texte.

Tatouage



Les photos sont des images matricielles, des tableaux (ou matrices) de pixels, chacun étant représenté par plusieurs octets. Les pixels sont rangés de gauche à droite et de haut en bas. Le pixel en haut à gauche est l'origine. Dans une image 800x600, le pixel du coin en haut à droite a pour coordonnées (799,0) et (799,599) en bas à droite.

Les images seront stockées au format *png* avec couleurs vraies (ou 16 millions de couleurs). Chaque pixel comportera 3 octets servant à coder les 3 couleurs primaires : rouge, vert et bleu. Le constat de départ est qu'une couleur est très peu altérée lorsqu'on modifie le bit de poids faible de chaque couleur.

rgb : 103 100 238 01100111 01100100 11101110

rgb : 102 99 237 01100110 01100011 11101101

2 couleurs, une par ligne, avec leurs codages RGB décimaux et binaires

Votre chef de projet vous demande d'utiliser cette observation pour coder un message en binaire, en utilisant les bits de poids faible des 3 composantes couleur de chaque pixel. En effet, pour la couleur, peu importe la valeur du bit de poids faible (0 ou 1), nous l'écraserons donc allègrement avec notre codage.

Pour savoir à quel pixel commence et s'arrête le message, nous allons utiliser la méthode suivante :

- les 5 premiers pixels de la première colonne (0,0), (0,1) … (0,4) sont utilisés pour coder le nombre de caractères du message à coder.
- Le message débute dès le 6ème pixel (0,5)

exemple : voici les 3 couleurs des pixels (0,5) (0,6) et (0,7)

('01010101' , '11101111' , '00110011')
('01010101' , '11101110' , '00110011')
('01010101' , '11101110' , '00110011')

que nous utilisons pour écrire la lettre A codée 01000001 en *utf-8* (ou ASCII)

('0101010**0**' , '1110111**1**' , '0011001**0**')
('0101010**0**' , '1110111**0**' , '0011001**0**')
('0101010**0**' , '1110111**1**' , '00110011')

Stéganographie

Notation

Attendus techniques

Vous devrez rendre un code Python documenté avec des noms de variable et de fonction bien choisis.

Vous devrez présenter à l'oral :

1. L'interface de votre programme avec notamment :
   - Les préconditions : sous quel format sont attendues les données en entrée, quels sont les attendus implicites de votre programme (données vides acceptées?) etc...
   - Les postconditions : description précise de ce que renvoie votre programme (notamment lorsque les préconditions ne sont pas satisfaites) et sous quel format.
2. Un jeu de test permettant de tester les préconditions décrites ci-dessus.
3. Une discussion sur la complexité des algorithmes que vous avez implémenté.

Attendus généraux

Les points suivants sont pris en compte pour la notation :

- Capacité à rechercher des informations, autonomie, initiative et coopération au sein de l'équipe.
- Capacité à présenter la sténographie en général et votre méthode en particulier de manière claire et synthétique.
- Analyse du problème, capacité à argumenter vos choix :
  - des algorithmes (sous fonction(-alité), complexité…)
  - de validation et de documentation (interface, tests,…)
- La qualité du support (diapositives…) et de la présentation orale.