Assignment 01: Image Encryption by 2D EAT and RP

授課教師：王宗銘

2023/09/05

1. 請撰寫 2 個 python 程式。

第一個程式練習利用 2D-EAT+Durstenfeld 的 Random Permutation (RP)對影像作加密處理。

第二個程式練習利用 2D 2D-EAT 的 Reverse Matrix 及 Durstenfeld 的 Reverse Random Permutation (RRP)對影像作解密處理。

(1) 加密程式：

程式名稱 學號-01-2D-EAT-RP_enc.py。

Step 1: 請使用以下矩陣做 EAT 轉換，並在程式中給定參數(a, b)之數值。只要更改(a, b)數值，即可重新購建不同的矩陣，作 EAT 轉換。例如設定(a, b) = (1, 1)，則轉換矩陣為 $[A]=\begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix}$，代表則座標(x, y)的像素值會被轉換至座標$(x', y')$，如下式所示。請注意：

$$\begin{bmatrix}x'\\ y'\end{bmatrix}=\begin{bmatrix}1 & a\\ b & ab+1\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix} \ mod\ N,$$ 其中 N 代表影像解析度。

請在程式內設定一個變數 G，5≤G≤ 300，代表做 2D-EAT 的次數。若 G=89，代表整張影像會被做 89 次的 EAT 轉換。只要更改 G，即可對影像重新執行不同次數的加密。

請注意：設定之 G 值不能是 2D EAT cycle 的數值，否則影像不能顯示加密效果。各個影像解析度 2D EAT cycle 的數值，請參見投影片。例如，N=512 時，G≠384。

舉例： 假設 N=512，設定 G=51。Host 灰階影像(x, y)=(3, 7)位置之像素 P=19，經過 G=51 轉換後，轉換到(x', y')=(7, 179)。此時，(x', y')=(7, 179)之像素仍維持 P=21。

Step 2-1:

(1) 將 P=19 轉成 2 進制 8 個 bits 表示,“0001 0011”，並儲存在 array。下表為原索引{0, 1, 2, 3, 4, 5, 6, 7}對應之各位元。

Position	0	1	2	3	4	5	6	7
Bit value	0	0	0	1	0	0	1	1

(2) 使用自定的 seed，例如 seed=100，利用 mod 函數，來取 7 個隨機整數，分別對應 8 進制、7 進制…、2 進制之整數，並根據 Durstenfeld 的 RP 演算法，將原索引{0, 1, 2, 3, 4, 5, 6, 7}依據 7 個隨機整數做排列。假設產出之排列為{2, 1, 3, 5, 6, 4, 7, 0}，如下表所示

Permutation	2	1	3	5	6	4	7	0
Bit value	0	0	1	0	1	0	1	0

(3)將對應的 2 進制 8 個 bits 表示,“0010 1010” 轉成 10 進制之 42,並取代(x', y')=(7, 179)之像素。經過 2D-EAT+RP 後,Host 灰階影像(x, y)=(3, 7)位置之像素已被轉換至(x', y')=(7, 179),且原始像素值 P=19,已經被加密成 P=42。對所有 Host 影像之像素做類似的處理,即可產出一張利用 2D-EAT+RP 之加密影像。

Step 2-2:

若像素為彩色影像,則請依照 R, G, B 排列,並將之轉成 24-bit 位元。建議直接將 24-bit 做 Random Permutation,而非依照頻道(此與課堂所言不同!!),每 8-bit 做 Random Permutation 處理。例如(R, G, B)=(12, 122, 189),則對應的 24-bit 為 {RGB}=00001100 01111010 10111101。Random Permutation 做 23 次即可根據產出的 23 個隨機數,將此序列作排列成為{RGB}′.。

(2) 解密程式:

程式名稱 學號-01-2D-EAT-RRP_dec.py。

Step 1. 使用與加密影像一致的參數 (a, b, G),對加密影像作解密處理。請注意,解密時,需要用加密處理之反矩陣$\begin{bmatrix} ab+1 & -a \\ -b & 1 \end{bmatrix}$,同樣是做 G 次。如此,座標$(x', y')$的像素會被轉換至座標$(x, y)$,如下式所示。若 a=b=1,則 2D-EAT 之逆轉換為

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \ mod\ N$$

舉例:(x', y')=(7, 179)經過 G=51 次的逆轉換後,產出(x, y)=(3, 7)像素。此時像素值仍維持加密之數值,P=42。對所有 Encrypted 影像之像素做類似的處理,即可產出一張利用 2D-EAT+RRP 之解密影像。

Step 2:

(1) 將 P=42 轉成 2 進制 8 個 bits 表示, “0010 1010”,並儲存在 array。下表為原索引{0, 1, 2, 3, 4, 5, 6, 7}對應之各位元。

Permutation	2	1	3	5	6	4	7	0
Bit value	0	0	1	0	1	0	1	0

(2)根據相同的 seed,對(x, y)=(3, 7)產生 7 個隨機整數,分別為 8 進制、7 進制…, 2 進制。根據 Reverse Random Permutation (RRP) 演算法,依照 2 進制、3 進制…、8 進制之順序,產出原先的二進制序列 “0001 0011”。

(3) 將對應的 2 進制 8 個 bits 表示, “0001 0011” 轉成 10 進制之 19,如此可順利解密(x, y)=(3, 7)之像素 P=19。

position	0	1	2	3	4	5	6	7
Bit value	0	0	0	1	0	0	1	1

(3)請設定一個目錄，名稱為 source，儲存加密的影像。
請設定一個目錄，名稱為 encryp，儲存加密影像，並請在檔案名稱後加入 enc。
請設定一個目錄，名稱為 decryp，儲存解密影像，並請將檔案名稱後加入 dec。
例如：欲加密影像為 Lena.png，存在 source image 內；加密後之影像為 Lena_enc.png，存在 encryp 目錄內；解密後之影像為 Lena_dec.png，存在 decryp 目錄內。

4. 撰寫之程式：
(1)可以使用 openCV 套件。
(2)請注意 python openCV 之頻道排列是 blue, green, red，非為 red, green, blue。請做向量處理。
(3) python 版本>=3.10，請確認程式在 IDLE python 64 bit 是可執行的。
(4) 請遵守檔案編號原則，以免助教判定繳交格式錯誤，導致錯誤執行，不予評分。
(5) 提供 standard USC-SIPI 測試 png 影像。請將這些影像放在 source 目錄內。加密程式逐一將上述影像作加密處理。加密後，解密影像逐一將加密影像做解密處理。
1. Aerial.png, 2. Babara.png, 3. Baboon.png, 4. Boat.png, 5. House.png, 6. Lena.png, 7. Peppers.png, 8. Tank.png, 9. Truck.png。

5. 繳交: 請繳交壓縮檔案，壓縮方式請選 zip 或 rar。
壓縮檔案名稱: 學號-ass01.rar，包含下列 3 目錄
(1) 2 個 Python 程式，請放在與 source, encrypt, decryp 同層
加密程式：學號-01-2D-EAT-RP_enc.py
解密程式：學號-01-2D-EAT-RRP_dec.py
(2) source 目錄：內含原始 9 張影像
(3) encryp 目錄：內含已加密之 9 張影像
(4) decryp 目錄：內含已解密之 9 張影像
b. readme.txt，請放在與 source, encrypt, decryp 同層，敘述如何執行 python 程式，載明是否需要額外的套件。