影像處理作業

A watermarking-based image ownership and tampering authentication scheme  
基於浮水印技術圖像認證方法

實作

系級:資工碩一

學號:107522099

姓名:詹振宗

2018年12月21日

作業報告內容

1. 封面 (題目, 系級, 學號, 姓名, & 日期)

2. 介紹 (例如，用什麼方法解決什麼問題)

3. 怎麼做 (how to do) (例如，提出或使用什麼方法或演算法)

4. 結果 (先文字敘述結果，再呈現影像輔助說明)

5. 討論 (discussion) 與 / 或 結論 (conclusion)

**A.介紹**

**摘要**

本作業依照論文[A watermarking-based image ownership and tampering authentication scheme](https://www.sciencedirect.com/science/article/abs/pii/S016786550500259X)內的提出的影像浮水印技術實作，此方法不同於以往的浮水印技術無法定位修改處的缺點，此論文題提出的方法可定位出影像被竄改之處，而為了使程式更容易實作，目前只考慮灰階圖片的浮水印認證方法，而且實作以下三個功能

1)將影像加上隱藏式浮水印

2)認證已加上浮水印的影像是否有被竄改

3)可以定位影像被修改之處的功能

**相關工作**

依照論文內所提及的方法，影像浮水印技術分為以下兩種

* fragile authentication

以加上浮水印的影像上的任何變動都是不允許的，否則此浮水印技術

會將影像視為已經被竄改，問文提出的方法以及本作業是屬於這種方法。

* robust authentication

可認證經過濾，壓縮，裁剪，銳化，模糊的影像，依舊可以使用浮水印認證

影像浮水印根據實作方式分為

* labeling-based schemes

需額外檔案作為認證，但是會增加記憶體與硬碟額外的負擔

* watermarking-based

浮水印直接嵌入在多媒體影像內，分為1)浮現式2)隱藏式兩種

又通常屬於fragile authentication

**B.實驗方法**

1. 將影像切為多個九宮格(9X9)，(圖1)

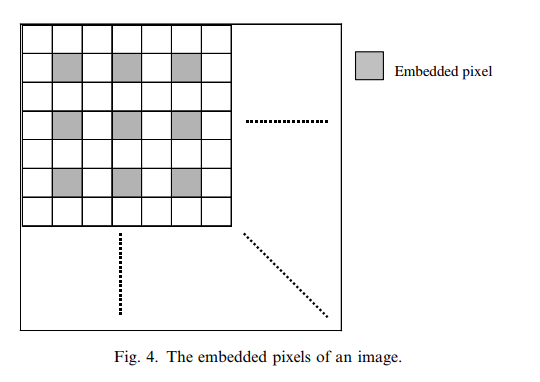


圖1

1. 將每個九宮格的ID(Index of this block )、九宮格的特徵 (P1~P8) (圖2) 、影像ID(Identification of the image)、加密者密碼(User’s secret key) 經過HASH成一個長度128 bits的特徵值

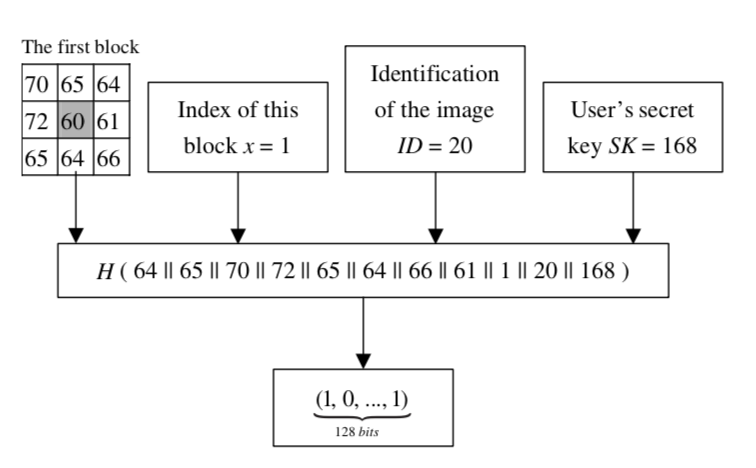
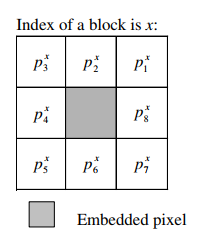


圖2 圖3

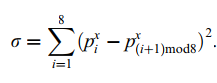
1. 將128bit特徵值壓縮至2~4 bits以便藏入圖片中

首先要決定要縮成多短(論文規定2~4個)，依照論文所提出的公式(圖4)，依照每個九宮格像素之間的差異，算出σ，而作者經實驗發現σ介於

0 ≤ σ < 8 壓縮成2個bit

8 ≤ σ <16 壓縮成3個bit

16 ≤ σ ≤ 255 壓縮成4個bit

圖4

1. 將HASH後的特徵值塞入九宮格中間的像素的LSB(Least Significant Bit)

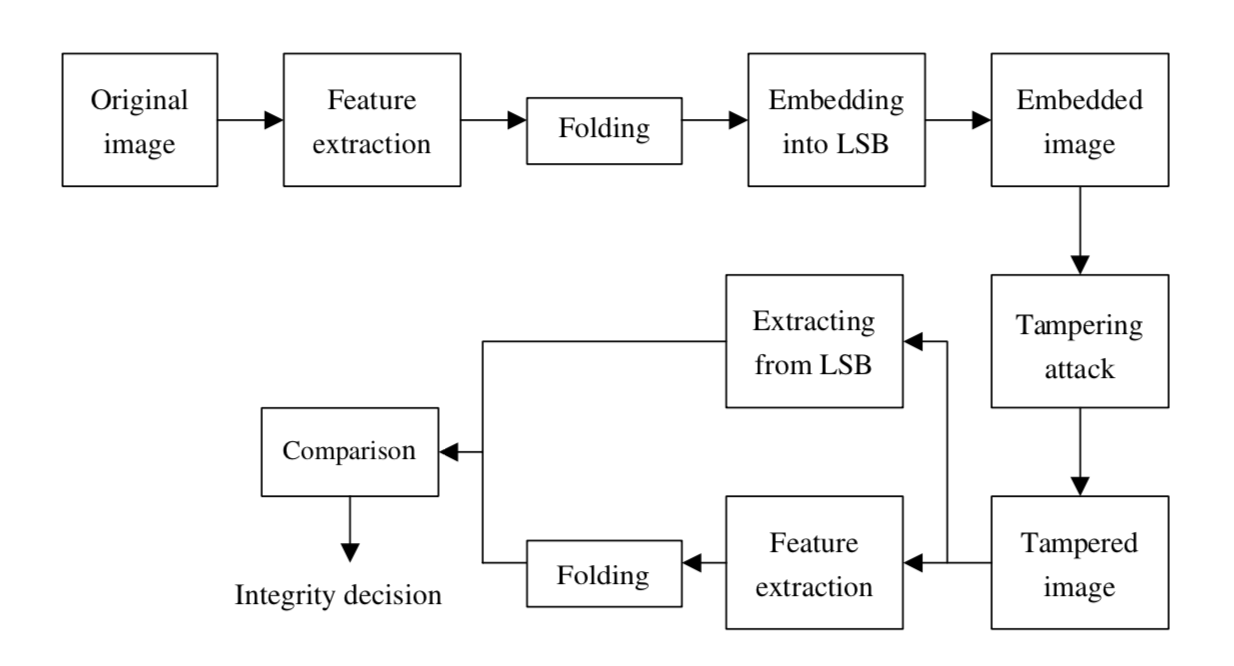
**認證影像步驟**

首先得到一張已作浮水印處理之影像，圖

1. 照上述方法每個九宮個的8個Pixel作運算得到特徵值
2. 特徵值與九宮格中間的嵌入值比較

相同=>驗證成功

不相同=>驗證失敗



**結果**

**實驗環境**

* 程式語言 python
* 實驗平台 windows 10
* 相依套件
  + OpenCV-python
  + Hashlib
  + numpy

本實作使用灰階圖片，因爲本方法使用fragile authentication

在此浮水印壓縮技術，任何變動都是不允許的，必須使用不壓縮或壓縮後不失真的影像格式，如bmp,png等影像格式



利用程式

Main.py

本程式提供兩個函式，分別為

* add\_waterMarking()
  + 傳入值為灰階影像像素的二維陣列
  + 回傳已做浮水印處理後的灰階像素的二維陣列
* validate\_waterMarking()
  + 傳入值為灰階影像像素的二維陣列
  + 回傳1)影像是否有被竄改2)被竄改的像素所在的九宮格位置

**結論**

依照論文使用的方法的確可以辦別影像是否遭到竄改，以及定位竄改之處，