Reversible Digital Watermarking

Chang-Tsun Li

Department of Computer Science
University of Warwick

Reversible Watermarking Based on Difference Expansion (DE)

- In some medical, legal and military applications, slight changes to content due to watermarking is not acceptable. So allowing the original content to be completely restored from the watermarked media is useful.
- Watermarking with this capability is called reversible or lossless watermarking.
- This work has inspired many others:

  J. Tian, "Reversible Data Embedding Using a Difference Expansion," IEEE TCSVT, 13(8), 2003

What is Difference Expansion (DE)

- Given two greyscales x and y & a bit b to be embedded
- Average $a = \left\lfloor \frac{x+y}{2} \right\rfloor$, Difference $d = x - y$ (1)
- If $x = 206, y = 201$ & $b = 1$

➔ $a = \left\lfloor \frac{x+y}{2} \right\rfloor = 203$ $d = x - y = 5 = (101)_2$

$$d' = d \| b = (101)_2 \| (1)_2 = (1011)_2 = 11$$

$$\equiv \boxed{d' = 2 \times d + b} = 2 \times 5 + 1 = 11$$

d (=5) has been expanded into d'(=11)

Embedding Data Bit in x & y

- x' & y': watermarked version of x and y

From $$a = \left\lfloor \frac{x+y}{2} \right\rfloor, \qquad d = x - y \tag{1}$$

we get $$x' = a + \left\lfloor \frac{d'+1}{2} \right\rfloor, \quad y' = a - \left\lfloor \frac{d'}{2} \right\rfloor \tag{2}$$

$$x' = a + \left\lfloor \frac{d'+1}{2} \right\rfloor = 203 + \left\lfloor \frac{11+1}{2} \right\rfloor = 209$$

$$y' = a - \left\lfloor \frac{d'}{2} \right\rfloor = 203 - \left\lfloor \frac{11}{2} \right\rfloor = 198$$

Extracting Data Bit and Recovering x & y

- From (1) $a' = \left\lfloor \frac{x'+y'}{2} \right\rfloor = \left\lfloor \frac{209+198}{2} \right\rfloor = 203 = a$ (Note a' = a)

$$d' = x' - y' = 209 - 198 = 11 = (1011)_2 = (101)_2 \| (1)_2 = d \| b$$

$d = \lfloor d'/2 \rfloor$ & $b = 1$ has been correctly extracted !

- From (2) $x = a + \left\lfloor \frac{d+1}{2} \right\rfloor = 203 + \left\lfloor \frac{5+1}{2} \right\rfloor = 209$

$$y = a - \left\lfloor \frac{d}{2} \right\rfloor = 203 - \left\lfloor \frac{5}{2} \right\rfloor = 201$$

$x = 206$ and $y = 201$ have been recovered!

Expandable Difference Values

- Overflow ($x' > 255$ or $y' > 255$) and underflow ($x' < 0$ or $y' < 0$) must be avoided when expanding their difference, i.e.,

$$0 \le x' = a + \left\lfloor \frac{d'+1}{2} \right\rfloor \le 255 \quad \text{and} \quad 0 \le y' = a - \left\lfloor \frac{d'}{2} \right\rfloor \le 255$$

$$\Rightarrow \quad |d'| \le 2(255-a) \quad \text{and} \quad |d'| \le 2a-1$$

- If $|d'| = |2 \times d + b| \le \min\{2(255-a), 2a-1\}, \ \forall b \in \{0,1\}$

  then d is expandable.

- Expandable difference values allow original (x, y) to be recovered without other arrangement.

Changeable Difference Values

$$d = 2 \times \left\lfloor \frac{d}{2} \right\rfloor + LSB(d)$$

- If $LSB(d)$ can be replaced by a data bit b and

$$\left| 2 \times \left\lfloor \frac{d}{2} \right\rfloor + b \right| \leq \min\{2(255-a), 2a-1\}, \quad \forall b \in \{0,1\}$$

  then d is changeable.

- A changeable (but not expandable) difference value does NOT allow original (x, y) to be recovered without other arrangement.
- An expandable integer is also changeable.

Embedding Algorithm

1. Form a set of m pixel pairs (x, y) and calculate their difference values $d = \in \{d_i \mid i = 1, \ldots, m\}$ using Eq. (1)
2. Partition d into 4 sets: EZ, EN $(=EN1 \cup EN2)$, CN and NC
3. Create a binary location map L such that $L_i = \begin{cases} 1, & \text{if } d_i \in \{EZ \cup EN1\} \\ 0, & \text{otherwise} \end{cases}$

   and perform lossless compression on L
4. Collecting LSBs of the d_i in $EN2 \cup CN$ to form C
5. Embed bit stream $L||C||P$ by
   - expanding d_i in $EZ \cup EN1$
   - changing d_i in $EN2 \cup CN$ (P is the actual payload or watermark)
6. Perform Eq. (2) $x' = a + \left\lfloor \frac{d'+1}{2} \right\rfloor, \quad y' = a - \left\lfloor \frac{d'}{2} \right\rfloor$

Step 1 - Embedding Algorithm

Step1. Form a set of m pixel pairs (x, y) and calculate their difference values

Each pair can be

- horizontally neighbouring pixels
- vertically neighbouring pixels
- formed in a pseudo random manner under the control of a secret key
- other ways

Step 2 - Embedding Algorithm

Step2. Partition d into 4 sets: EZ, EN (=$EN1$ U $EN2$), CN and NC

- EZ: all expandable $d_i = 0$ and $d_i = -1$

  EZ is separated from ZN because $d_i = 0$ and $d_i = -1$ together with $d_i = 1$ and $d_i = -2$ when d_i is in $EN2$ constitute 4 special cases which can increase embedding capacity (see description of Step 4).

- EN: all expandable d_i not in EZ
  - Expansion incurs more significant distortion, so depending on the payload, only a subset of EN is selected for expansion.
  - $EN1$: selected for expansion; $EN2$: not selected for expansion
  - $EN = EN1$ U $EN2$ (See Tian's paper for details)
- CN: all changeable d_i not in EZ U EN
- NC: all non-changeable d_i

Step 3 - Embedding Algorithm

Step3. Create a binary location map L such that

$$L_i = \begin{cases} 1, & \text{if } d_i \in \{EZ \cup EN1\} \\ 0, & \text{otherwise} \end{cases}$$

and perform lossless compression on L

- The location map L is required because the extraction side needs to know which d_i have been expanded.
- L needs to be compressed because it will have to be embedded and compression reduces the payload

Step 4 - Embedding Algorithm

Step 4. Collecting LSBs of the d_i in $EN2$ U CN to form C

- The LSB of d_i in $EN2$ U CN are changeable only, so their original LSB need to be saved, otherwise they cannot be recovered.
- We do not want to expand those d_i in $EN2$ in order to reduce distortion
- Special cases: for d_i = 1 and d_i = -2 in $EN2$ U CN, their LSBs do not have to be saved because they remain unchanged after embedding.

Step 5 - Embedding Algorithm

Step 5. Embed bit stream $L||C||P$ by expanding d_i in EZ U EN U CN (P is the actual payload or watermark)

$$\text{Let } B = L\|C\|P = \{b_1, b_1, b_1, ...\}$$

$$\text{if } d_i \in EZ \cup EN1 \qquad \text{(Expandable)}$$

$$d_i := 2 \times d_i + b_i$$

$$\text{elseif } d_i \in EN2 \cup CN \quad \text{(Changeable)}$$

$$d_i := 2 \times \left\lfloor \frac{d_i}{2} \right\rfloor + b_i$$

Step 6 - Embedding Algorithm

Step 6. Perform Eq. (2) $x' = a + \left\lfloor \frac{d'+1}{2} \right\rfloor, \quad y' = a - \left\lfloor \frac{d'}{2} \right\rfloor$ to get the watermarked image

Extraction & Recovering Algorithm

1. Form a set of m pixel pairs (x', y') and calculate their difference values $d' = \{d'_i \mid i = 1, \ldots, m\}$ using Eq. (1)
2. Partition d' into 2 sets: CH (CHangeable) and NC (Non-Changeable)
3. Collecting LSBs of the d'_i in CH to form $B = L \| C \| P = \{b_1, b_2, b_3, \ldots\ldots\}$
4. Decompress location map L
5. Restore original difference values d_i (to be explained later)
6. Authenticate content by comparing the extracted P against its original version
7. Recover (x_i , y_i) based on d_i using Eq. (3)

$$x_i = a_i + \left\lfloor \frac{d_i + 1}{2} \right\rfloor, \qquad y_i = a_i - \left\lfloor \frac{d_i}{2} \right\rfloor \tag{3}$$

Note a_i remains unchanged after embedding, so $a_i = \left\lfloor \frac{x'_i + y'_i}{2} \right\rfloor$

Step 5 - Extraction Algorithm

Step 5. Restore original difference values d_i

Let $B = L \| C \| P = \{b_1, b_2, b_3, ...\}$

if $d'_i \in CH$

  if $L_i = 1$ (ith bit of location map L)

    $d_i = \lfloor d'_i / 2 \rfloor$ ($\because d_i$ is expanded)

  else

    if $d'_i = 1$ or $d'_i = -2$

      $d_i = d'_i$

    else

      $d_i = 2 \times \lfloor d'_i / 2 \rfloor + b_i$

Conclusions

- Each difference value can be allowed to carry more than one bit.

  If $k(k \in Z^+)$ is the largest integer that satisfies

  $$\left|2^k \times d + b\right| \leq \min\{2(255-a), 2a-1\} \quad \forall\, b \in \{0,1,2,...,2^k-1\}$$

  then the *hiding capacity* of d is k.

  $k = 1$ is the special case that we have discussed

- To minimise embedding distortion
  - d with small magnitude is preferred $\because (x-x')^2 + (y-y')^2 \approx \frac{h^2}{2}$

    (see Tian's paper for details)
  - d with greater hiding capacity is preferred when full capacity is not needed
- Multi-layer embedding is possible
  - Embedded image can be embedded again
  - The overall hiding capacity of each layer decrease as the number of expandable difference values is inversely proportional to the number of layers.

Steganography and Steganalysis

Chang-Tsun Li

Department of Computer Science
University of Warwick

Steganography

- Steganography:
  - is the act of **covert communications** with the aim of preventing the third party from knowing that secret communication is taking place
  - is about hiding secret message in the **cover media** such that the **stego media** remains **innocuous**

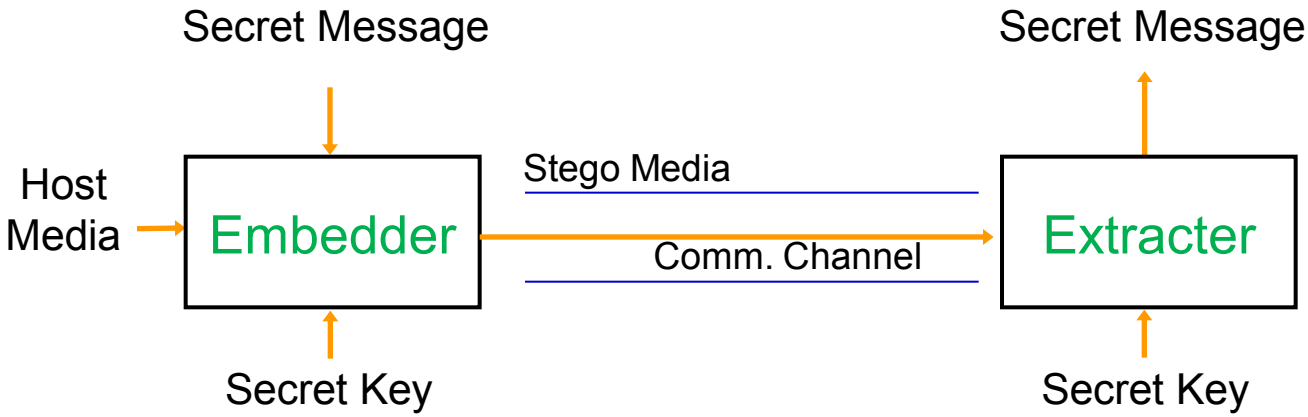
- Main requirement:

  Undetectability of secret communications

Steganography & Watermarking

- Both are forms of data hiding
- Different purposes:
  - Watermarking: protecting the **cover/host media** or authenticating the **cover/host media**. It is all about the **cover/host media**.
  - Steganography: It is all about the **hidden message**, not the cover/host media. The user can use any suitable cover media from a large space of cover media.
- Different goals:
  - Watermarking: Make the act of data hiding known
  - Steganography: Conceal the act of data hiding
- Different payloads (sizes of secret messages):
  - Watermarking: small
  - Steganography: large

Steganographic Model



Where to Hide

- Sequential: e.g. sequentially replacing the LSBs of each image pixel with message bits starting from the upper left corner
  - Easy to implement,
  - But also easy to detect. E.g., analysing statistical properties of pixels in the same order and looking for sudden changes of statistical properties can detect covert comm if not all pixels are carrying secret data.
- Random: select elements of the cover media according to a secret key. E.g., use pseudo-random number generator (PRNG) and a secret key to generate a random walk through the cover media.
  - Greater security

Where to Hide

- Adaptive: Select elements of the cover media based on the content of the cover media
  - Why: Statistical detectability of data hiding is likely to be **content based**.
  - E.g. data hidden in noisy images or highly textured areas of an image is more undetectable than the same data hidden in smooth areas. ➔ Hide more in textured areas and less in smooth areas

Statistics/Model Preserving Steganography

- Modulating the cover media so that statistical properties, such as histogram, or models of the cover media, are preserved in the stego media.
- Example: LSB Embedding – replacing the LSBs of pixels with message bits
- Let the original pixel value be $(72)_{10} = (0100\ 1000)_2$
  - If Message bit = 0 ==> stego pixle = $(0100\ 1000)_2 = (72)_{10}$
  - If Message bit = 1 ==> stego pixle = $(0100\ 1001)_2 = (73)_{10}$

Detectability of Statistics /Model Preserving Steganography

- Tougher for the embedder: The embedder carries the burden of ensuring the preservation of as many statistics as possible.
- Easier for the adversary / steganalyst: The steganalyst's detection of one single statistics unpreserved by the embedder will jeopardise the covert communication.

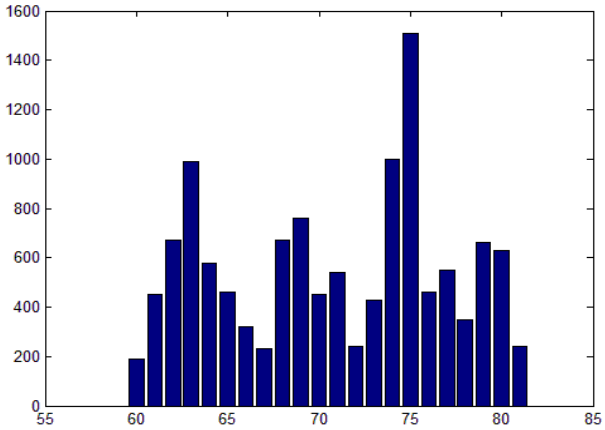
Problem with LSB Embedding

E.g.The original pixel value be $(72)_{10} = (0100\ 1000)_2$

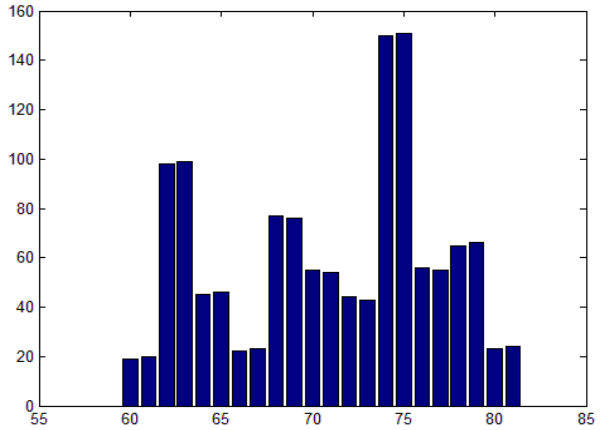
If Message bit = 0 ==> stego pixle = $(0100\ 1000)_2 = (72)_{10}$

If Message bit = 1 ==> stego pixle = $(0100\ 1001)_2 = (73)_{10}$

For any original pixel with gray level $2g$, $g = 0, \ldots, 127$, the probability that the stego pixel's gray level remains the same ($2g$) and becomes ($2g + 1$) are both equal to 0.5. This create an unusual histogram like the second plot.



cover image



stego image

Masking Embedding as Natural Processing

- Many types of device-dependent noise remain in images: e.g.
  - **Photo Response Non-Uniformity** due to sensor imperfection
  - **Dark current** produced when the sensor is not exposed to light
  - **Colour demosiacking errors** due to colour interpolation

- Masking embedding distortion as device noise to make it difficult to tell whether the slightly increased noise level of the stego image is due to **data hiding** or to the **device**

Steganalysis

Steganalysis is about

- **Detecting the presence of a secret message** given a stego media
- **Recovering message attributes** such as message length or content (i.e., forensic staeganalysis)

Categories of Steganalysis

- Categories of steganalytical methods
  - Targeted steganalysis: focusing on specific steganagraphic algorithms
  - Blind steganalysis: Aiming at all types of steganagraphic algorithms
- Both categories can be seen as **classification problem** so pattern recognition and machine learning techniques are applicable in steganalysis

Steganalysis as Classification Problem

- It is a two-class classification problem - **cover** media or **stego** media (i.e., **absence** or **presence** of secret message)
- The dimensionality of the space of all media is too high
  ➔ ***features*** that characterise media are used instead
- **Fourier Transform of the intensity of histogram** of an image is a good example, each component representing a feature.
- The **boundary** between the clusters of *cover* and *stego* media can be learnt through a training phase.
- The feasibility of the boundary
  - determines false positive & false negative rates
  - depends on the **discriminating power of the feature set**.

Targeted Steganalysis

- The steganalyst knows the steganographic algorithm or assumes that it is used
- The knowledge about the steganographic algorithm can be turned into useful **features**. E.g., if LSB embedding is used, then we know
  - LSB embedding adds noise to stego images
  - It increases difference between neighbouring pixels
  - Large ***sum of absolute difference*** between neighbouring pixels suggests true positive, while small sum indicates low positive
- Is ***sum of absolute difference*** a good feature? Probably not !
  - the *intra-class* variation may be greater than the *inter-class* variation due to the diversity of the cover media.
  - **feature selection** is a major research area.

Blind Steganalysis

- No prior knowledge about the steganographic algorithm is available ➔ cannot create stego media for training purpose.
- Two options
  - Generate stego media with a wide variety of known steganographic algorithms
  - One-class learning: the classifier learns knowledge about cover media in the feature space and labels a piece of media as
    - » **Cover media** if the feature set of the media in question is close to the centroid of the cluster of cover media,
    - » **Stego media** if not close enough

Conclusions

The choice of the cover media plays a key factor in determining the security of steganographic algorithm

- It is more difficult to detect hidden message in noisy or highly textured images than in images with large smooth areas. This is because the intra-class variation of the clusters of cover and stego images of the former type are greater.
- It is more difficult to detect messages of the same length hidden in smaller images than in larger images because features computed from a smaller sample space are more noisy.
- Colour images are poorer cover media for data hiding than greyscale images because colour images provide more data for statistical analysis.