Information Hiding: Steganography & Steganalysis

Dr. Zoran Duric
Department of Computer Science
George Mason University
Fairfax, VA 22030
http://www.cs.gmu.edu/∼zduric/

Steganography (“covered writing”)

- From Herodotus to Thatcher.
- Messages should be undetectable.
- Messages concealed in media files.
- Perceptually insignificant data is common in (uncompressed) media files.

Covered Writing Example

Sent by a German spy during WWI:

Apparently neutral’s protest is thouroughly discounted and ignored. Isman hard hit. Blockade issue affects pretext for embargo on byproducts, ejecting suets and vegetable oils.

Pershing sails from NY June 1!

Hiding in Images

- Idea: hide by modifying least significant bits (LSBs)
- Take an original image: rgb 410×614, 755k
- Convert to JPEG: 80% quality, 84k
- Insert the JPEG image into the original by replacing the LSBs by the bits of the JPEG file
- No noticable difference

Original/Uncompressed Image

80% JPEG Compression

JPEG Inserted into the Original Image

Steganography vs. Watermarking

Steganography	Watermarking
Goal:	**Goal:**
Hide existence of messages	Add “copyright” information
Hidden information “independent” of cover	Hidden information related to cover
Requirement:	**Requirement:**
Statistical undectability	Robustness
Successful attack:	**Successful attack:**
Detect hidden message	Render watermark unreadable

Steganography: Definition

♠ Simmons 1983: Prisoners problem

♠ USA – USSR non-proliferation treaty compliance checking

- Alice and Bob are prisoners, Wendy is a warden. Alice and Bob are allowed to exchange messages, say images, but Wendy checks all messages.
- Alice and Bob try to hide information in their messages so that Wendy cannot detect it.
- Wendy cannot arbitrarily suppress all messages; the prisoners' human rights cannot be violated without some proof of illegal activity.

Hiding by Matching Input

- LSB sequence $c_1, c_2, c_3, \ldots, c_n$.
- Message bits $m_1, m_2, m_3, \ldots, m_k$, $k < n$.
- Look for a good approximate match (e.g. N. Provos).
- **Theorem:** If the number of matching bits should exceed chance then the cover should be exponentially longer that the message.
- Hiding by matching is very wasteful (you can hide very few bits this way)

Secret Key Based Steganography

- If system depends on the secrecy of the method there is no key involved—pure steganography.
  - Not desirable — Kerkhoff's principle
- Compression + Encryption of the message
- Secret Key based staganography
- Public/Private Key Steganography

Lossy vs. Lossless Steganography

- Lossless steganography: modify lossless compression methods.
- An example would be modifying run length encoding process to embed messages.
  - During the encoding process the method checks all run lengths longer than one pixel.
  - Suppose that a run length of ten pixels is considered and that one bit needs to be embedded.
  - To embed a bit one the run length is split into two parts whose lengths add to ten, say nine and one; to embed a bit zero the run length is left unmodified.
  - The receivers check all run lengths. Two run lengths of the same color are decoded as a one.

  - A run length longer than one pixel, preceded and followed by run lengths of different colors, are decoded as a zero.

- Clearly, this technique relies on obscurity since detecting a file with information embedded by this technique is not hard.
- Lossy steganography: replace LSBs (least significant bits), modify PoVs (pairs of values)
- We are interested in lossy steganography.

LSB Methods (least significant bit)

- The given “cover” is an image.
- Image represented by pixel values
  - raw images: each pixel is a byte (gray value)
  - raw images: each pixel is a byte (color index in a palette)
  - raw images: each pixel is three bytes (r,g,b values)
- Image represented by a sequence of JPEG coefficients.
- LSBs of pixel values or JPEG coefficients can be altered freely.
- There are *many* LSBs in an image.

Embedding by Modifying Carrier Bits

- First approach identifies the carrier bits—i.e. the bits that will encode a message—and modifies them to encode the message.
- These carrier bits could be one or more LSBs of selected bytes of raster data—the selection process itself can use a key to select these bytes in pseudo-random order.
- Also, the raster data can be either raw image bytes (brightnesses and colors), or JPEG coefficients.
- Embedding is done by modifying the carrier bits suitably to encode the message.
- The message can be decoded from the carrier bits only—i.e., the receiver identifies the carrier bits and extracts the message using the key and the algorithm.

- These techniques can be compared using the following criteria (Westfeld, F5):
  - The *embedding rate* – the number of embedded bits per a carrier bit.
  - The *embedding efficiency* – the expected number of embedded message bits per modified carrier bit.
  - The *change rate* – the average percentage of modified carrier bits.

Message Embedding

- Compare the carrier bits and the message bits and change the carrier bits to match the message:
  - Changing the carrier bits to match the message bits.
  - Using bit parity of bit blocks to encode message bits.
  - Matrix encoding of message bits into carrier bits.

Changing the carrier bits to match the message bits

- Bit flipping: $0 \rightarrow 1$ or $1 \rightarrow 0$.
- Subtracting 1 from the byte value.

♠ For example, let the raster data bytes be

$$0100011\underline{1}\ \ 0011101\underline{0}\ \ 1001100\underline{0}\ \ 1010100\underline{1},$$

- ○ Using flipping to embed the message bits **0010** produces

$$0100011\underline{0}\ \ 0011101\underline{0}\ \ 1001100\underline{1}\ \ 1010100\underline{0},$$

- ○ Using subtraction to embed the message bits **0010** produces

$$0100011\underline{0}\ \ 0011101\underline{0}\ \ 1001\underline{0111}\ \ 1010100\underline{0},$$

- Subtraction produces more bit modifications, but the perceptual changes would be about the same as in the case of bit flipping.
- This technique has been used by various steganographic algorithms to embed messages in raw image data (gray and color images) and JPEG coefficients.
- The *embedding rate* is 1.
- The *embedding efficiency* is 2, since about 50% of carrier bits get modified.
- The *change rate* is 50%.

Block-Based Techniques

- Consider blocks of carrier and/or message bits at a time to embed a message into a cover
- *Bit parity* and *Matrix encoding*

Bit Parity

- #(available carrier bits) $\geq n\times$ #(message bits)
- Blocks of n carrier bits are considered and their parity compared to the corresponding message bits.
- If the parity matches the message bit nothing is done, otherwise any of the n bits in the current block can be modified to make the parity and the message bit match.
- The *embedding rate is* $1/n$
- The *embedding efficiency* is 2
- The *change rate* is $50\%/n$

Matrix Encoding

- Embeds k message bits using n cover bits, where $n = 2^k - 1$.
  $k = 2,\ n = 3$; $k = 3,\ n = 7$; $k = 7,\ n = 127$; $\ldots$
- Embed a k-bit code word $\mathbf{x}$ into an n-bit cover block $\mathbf{a}$.
- Let the bits of $\mathbf{x}$ be $x_i,\ i = 1 \ldots k$ and let the bits of $\mathbf{a}$ be $a_j,\ j = 1 \ldots n$.
- Let f be *xor* of carrier bit indexes weighted by the bit values, i.e.

$$f(\mathbf{a}) = \bigoplus_{j=1}^{n} a_j \cdot j$$

and let

$$s = \mathbf{x} \oplus f(\mathbf{a}).$$

- A modified cover block $\mathbf{a}'$ is then computed as

$$\mathbf{a}' = \begin{cases} \mathbf{a}, & s = 0 \ (\Leftrightarrow \mathbf{x} = f(\mathbf{a})) \\ a_1 a_2 \ldots \neg a_s \ldots a_n, & s \neq 0 \end{cases} .$$

- On the *decoder side* a k-bit message block $\mathbf{x}$ is obtained from an n-bit carrier block $\mathbf{a}'$ by computing

$$\mathbf{x} = f(\mathbf{a}').$$

- As an example let $\mathbf{x} = 101$ and let $\mathbf{a} = 1001101$. Therefore,

$$\begin{aligned} f(1001101) &= 001 \oplus 100 \oplus 101 \oplus 111 = 111 \quad \rightarrow \\ s &= 101 \oplus 111 = 010 \quad \rightarrow \quad a' = 1101101, \end{aligned}$$

i.e., the second bit was flipped to obtain $f(\mathbf{a}') = f(1101101) = 101$.

- The *embedding rate* of matrix encoding is

$$k/n \equiv k/(2^k - 1)$$

- The *embedding efficiency* is

$$k2^k/(2^k - 1)$$

- The *change rate* is

$$1/(n + 1) \equiv 2^{-k}$$

  (for any $(n = 2^k - 1)$-bit carrier block there are n matched k-bit code words and one that is mismatched).

- These numbers can change somewhat when JPEG coefficients are used to embed messages.

Embedding using Pairs of Values

- Utilizes perceptually similar *pairs of values* (*PoV*s) in raster data and modifies them to embed steganographic data.
- The *PoV*s are divided into *even* and *odd* elements.
- Embedding is done by modifying selected raster data to match the message.

- There are four cases:
  - The raster symbol is an *even* element (s_0) of some PoV (s_0, s_1) and the message bit is 0: leave s_0 unchanged.
  - The raster symbol is an *even* element (s_0) of some PoV (s_0, s_1) and the message bit is 1: replace s_0 by s_1.
  - The raster symbol is an *odd* element (s_1) of some PoV (s_0, s_1) and the message bit is 0: replace s_1 by s_0.
  - The raster symbol is an *odd* element (s_1) of some PoV (s_0, s_1) and the message bit is 1: leave s_1 unchanged.

Steganalysis

- Detecting the presence of a message.
- Statistically based.
- Extraction of message itself is secondary.

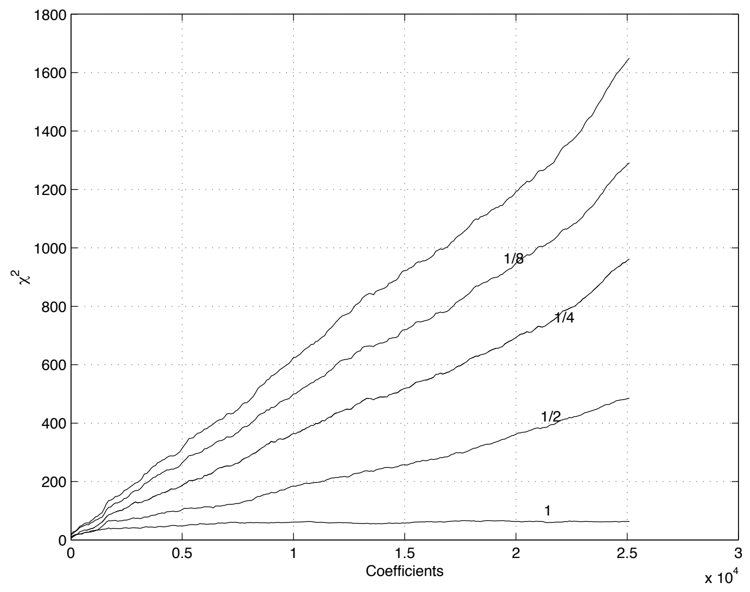
Image Statistics

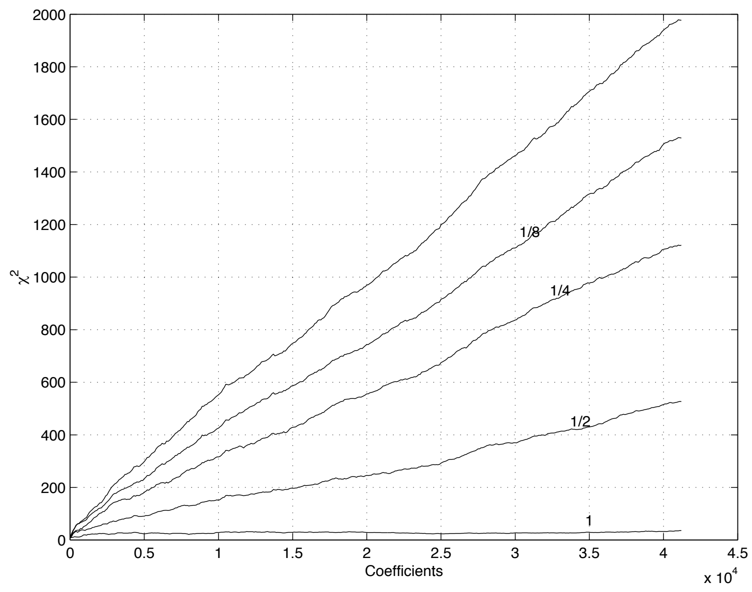
- Many are used in picture processing (e.g. entropy).
- Histogram-based statistics (Pfitzmann and Westfeld, IHW 99)
    - Coefficients come in pairs, differing by LSB; in JPEG their frequencies differ
    - In a modified image the 0s and 1s are equally probable; the distributions of odd and even coefficients become similar
    - h'_i and h''_i are histogram counts of a pair of coefficients
    - $\chi^2 = \frac{1}{2} \sum \frac{(h'_i - h''_i)^2}{h'_i + h''_i}$
    - Can be used to calculate the probabilty of a hidden message (integrating χ^2 distribution).

Defeating the χ^2 Test

- Image with message should have smaller χ^2 value.
- Method can be effective when most cover bits are involved.
- By using only some cover bits the published method fails.
- New χ^2 tests can still detect activity.

χ^2 test for skipped LSBs



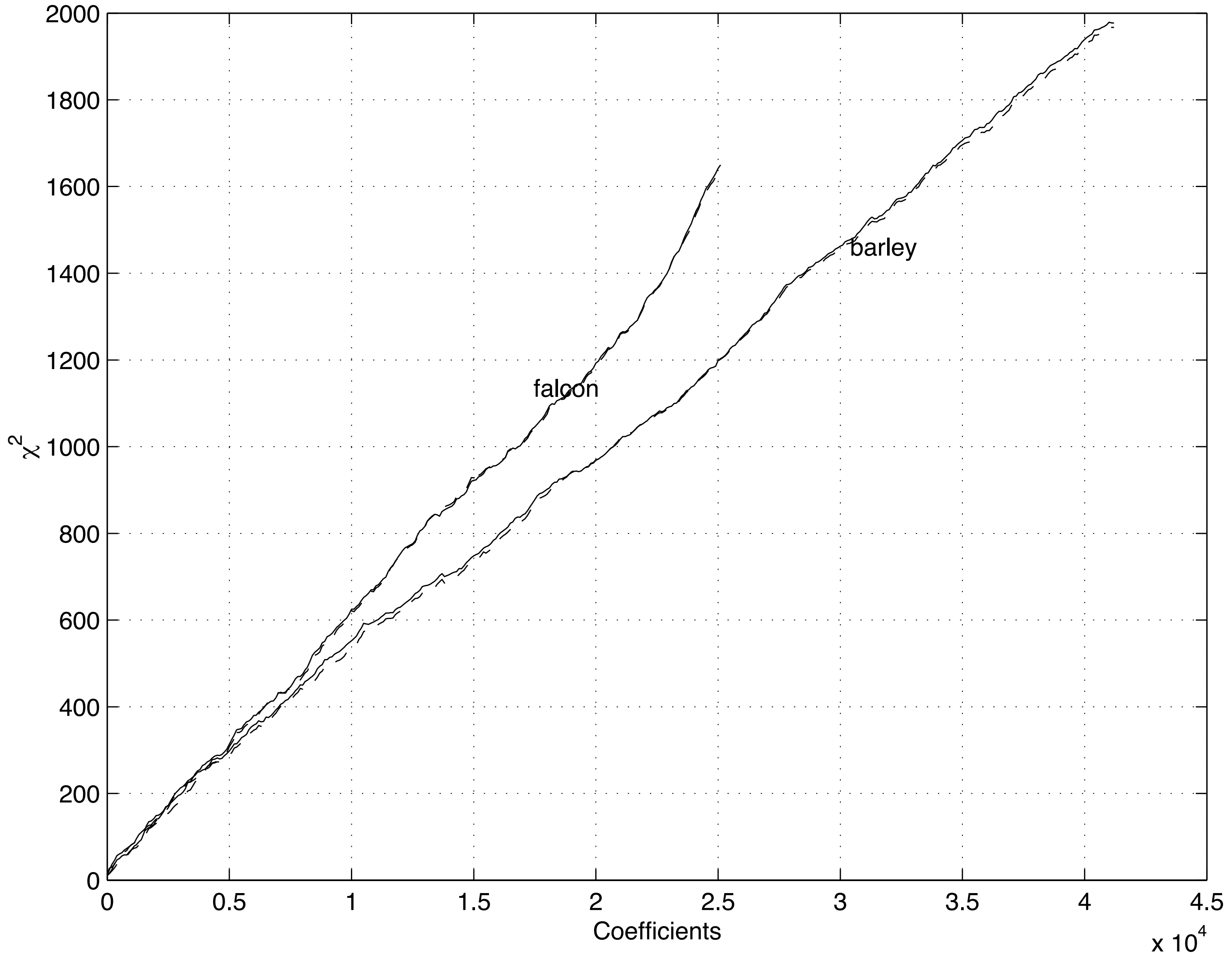


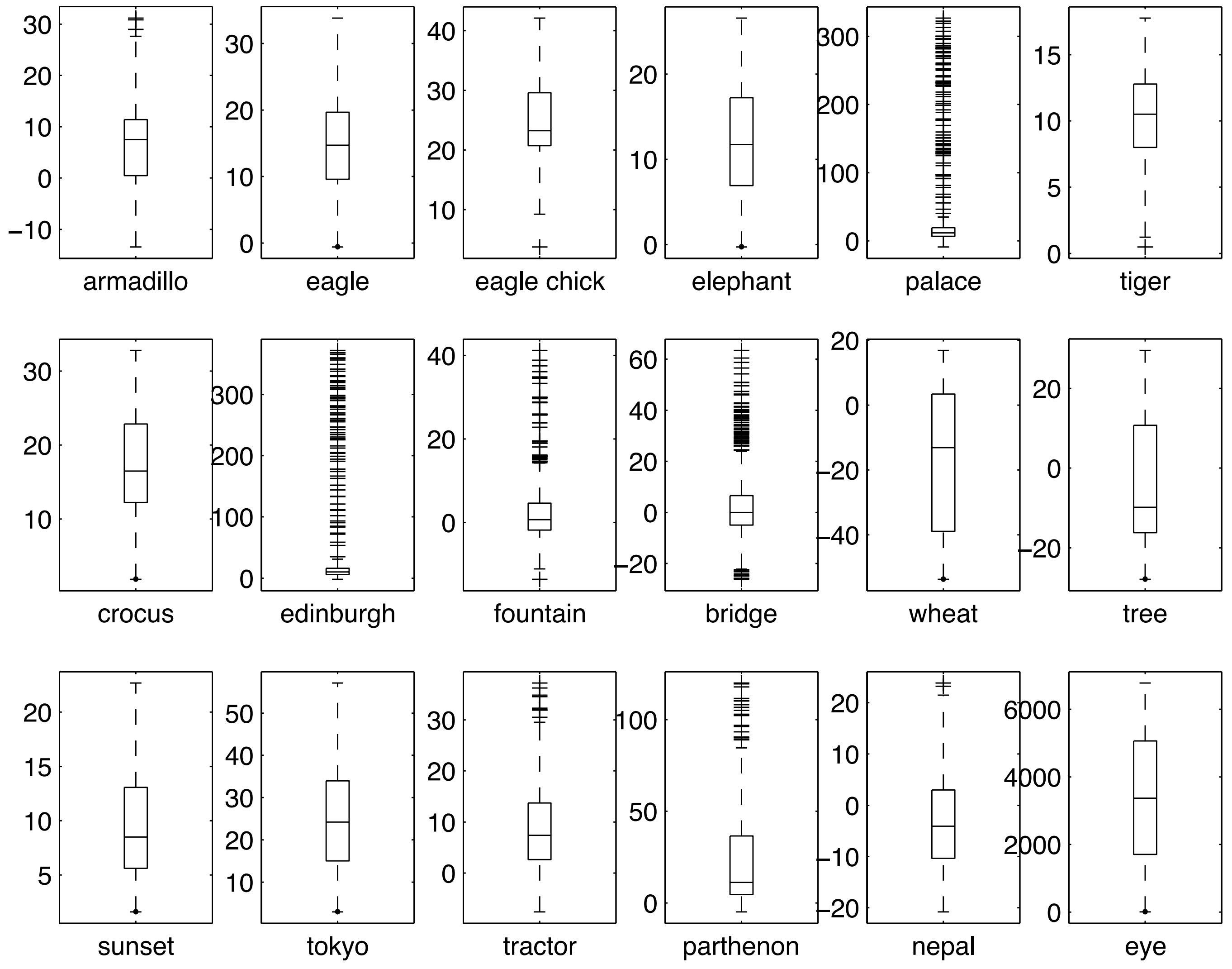
The “falcon” image (left) has 25179 coefficients available for embedding, altered: 12606, 6279, 3118, and 1569. The “barley” image (right) has 41224 coefficients available for embedding, altered: 20544, 10256, 5099, and 2545.

Using Codes that Mimic Statistics

- Use simple codes to modify and lengthen message.
  - If $m_i = 1$ replace with 00 or 11
  - If $m_i = 0$ replace with 01 or 10
- Use choices to create an encoded message that maintains χ^2 statistics.
- "Greedy algorithm": Each choice minimizes current deviation from original χ^2.
- This is remarkably good in practice.

Using Codes that Mimic Statistics: Examples


2000
1800
1600
1400
1200
1000
800
600
400
200
0
χ^2
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Coefficients
$x\ 10^4$
falcon
barley


armadillo
eagle
eagle chick
elephant
palace
tiger
crocus
edinburgh
fountain
bridge
wheat
tree
sunset
tokyo
tractor
parthenon
nepal
eye

Perfect Histogram Matching

- Mimic histogram directly.
- Stronger result than just χ^2.
- We first consider 2-bit codes.
- Can construct a graph G such that:

  Theorem: A perfect histogram matching exists if and only if there is a solution to the capacitated f-matching problem for G.
- Good algorithms exist for the capacitated f-matching problem.

Complexity of Perfect Histogram Matching

- What if b-bit codes are used?
- **Theorem:** Perfect histogram matching is NP-complete, for $b \geq 3$.
- If $b = 2$ it is easy; if $b = 3$ it is very hard.
- Importance of negative results.

Steganography & Steganalysis

- J. Fridrich (several papers): Stego images (containing embedded information) behave differently than clean images. Embed information in images and compute various features to detect stego content. Requires careful choice of features for each stego-insertion method.
- H. Farid: Collect large number of images $(10,000)$ and design a classifier (SVM) to differentiate clean and stego images using statistics of wavelet coefficients. Problem: Training and testing on the same image set.
- Information Theory (C. Cachin, P. Moulin): requires a good models for cover images. Problem: anybody can create their own images. Existing bounds are not tight enough.

Embedding using Pairs of Values

- Utilizes perceptually similar *pairs of values* (*PoV*s) in raster data and modifies them to embed steganographic data.
- The *PoV*s are divided into *even* and *odd* elements.
- Embedding is done by modifying selected raster data to match the message.

- There are four cases:
  - The raster symbol is an *even* element (s_0) of some PoV (s_0, s_1) and the message bit is 0: leave s_0 unchanged.
  - The raster symbol is an *even* element (s_0) of some PoV (s_0, s_1) and the message bit is 1: replace s_0 by s_1.
  - The raster symbol is an *odd* element (s_1) of some PoV (s_0, s_1) and the message bit is 0: replace s_1 by s_0.
  - The raster symbol is an *odd* element (s_1) of some PoV (s_0, s_1) and the message bit is 1: leave s_1 unchanged.

- If the message bits and raster data are uncorrelated, and the proportion of ones and zeros in the message is equal approximately half of the raster data need to be modified to embed a message.
- On the *receiver (decoder)* side the raster data are examined:
  - Each raster symbol is interpreted as either even or odd element of some *PoV*.
  - Even elements are decoded as zeros, odd elements are decoded as ones.

- An example of a steganographic technique that uses *PoV*s to embed messages is *EzStego* (by Romana Machado) used in reduced-color-set images.
- In a full-color-set (RGB) image each color is represented by three values corresponding to the red, green, and blue intensities.
- In a reduced color set the colors are sorted in lexicographic order; the sorted list of colors is called a *palette*.
- The palette is stored in the image header and the raster data are formed by replacing the colors by the corresponding indexes in the palette.
- If the palette has less than 256 colors the three-bytes per pixel full-color image can be represented using just one byte per pixel.
- To recover actual colors both the raster data and the palette are needed. Each raster data value is replaced by the corresponding color.

- Note that colors that are neighbors in the palette, and therefore are assigned indexes that differ by one, can correspond to colors that look very different.
- For example, it is possible that the palette colors with indexes 0, 100, and 101 correspond to RGB colors $(5, 5, 5)$, $(255, 5, 0)$, and $(10, 10, 10)$, respectively.
- Thus, flipping a bit and changing a color from, say, 100 to 101 could create a visible artifact in the image.

Sorting the Palette in *EzStego*

- Let the original palette be $C_{old} = \{c_i,\ i = 0, ..., n-1\}$, let $I(c_i, C_{old}) \equiv i$ be index of c_i in C_{old} and let $\delta(a, b)$ be the distance between colors a and b.
- Sorting is done using this algorithm:

```
1. $D \leftarrow \{c_0\}, \quad C \leftarrow C_{old} \backslash \{c_0\}; \quad c \leftarrow c_0,$
2. Find color $d \in C$ that is most distant from $c$
3. $D \leftarrow \{c_0, d\} \equiv \{d_0, d_1\}, \quad C \leftarrow C \backslash \{d\}$
4. *while* $C \neq \emptyset$ *do*
5.    Find color $d \in C$ that is most distant from $c$
6.    Find 2 colors $\{d_i, d_{i+1}\} \in D$ so that $\delta\{d_i, d\} + \delta\{d, d_{i+1}\}$ is minimal
7.    $D \leftarrow \{d_0, \ldots, d_i, d, d_{i+1}, \ldots\}, \quad C \leftarrow C \backslash\, d, \quad c \leftarrow d$
8. *endwhile*
```

- Note that this algorithm finds an approximation to the Traveling Salesperson problem in the color palette C_{old}, where colors correspond to cities.
- The *PoV*s that the algorithm uses correspond to the indexes of sorted colors in the original palette.
- The *PoV*s are

$$(I(d_{2k}, C_{old}), I(d_{2k+1}, C_{old})),\ k = 0, \ldots, n/2,$$

where $I(d_i, C_{old})$ is the index of color d_i in C_{old} and $D = \{d_0, d_1, \ldots, d_{n-1}\}$ is the sorted palette.

Conclusions

- Many steganographic algorithms have been published (Internet)
- Few formal results regarding limits on statistical detection of stego content
- Demonstrated that finding images that match long messages is hard. Designing codes to match χ^2 image statistics is not hard.
- Possible to design codes to match other statistics.
- Tight bounds on steganography and steganalysis are not known.