Adaptive data hiding based on VQ compressed images

W.-C. Du and W.-J. Hsu

Vision, Image and Signal Processing, IEE Proceedings  
22 Aug. 2003  
Volume: 150, Issue: 4  
Page(s): 233-238

Presented by  
Benjamin Tripp  
November 27, 2006
Reasons

- Information is frequently transmitted in a range of digital forms including text, image, audio, video, and other media.
- Data hiding involves embedding secret data into other cover media with minimal perceivable degradation.
- Data hiding is an important tool for things such as protecting copyright (watermarking), communicating secretly, and embedding captions.
Data Hiding

• Data hiding has two fundamental requirements
  – The degrading of quality due to embedding should be minimized
  – The hiding capacity should suffice to store a reasonable amount of hidden data

• The goal of data hiding techniques is to cause a minimum of visual or auditory degradation for any reasonable quantity of hidden data
Vector Quantization
Pre-processing data

• Hidden data can be in any kind of format

• Secret data should be pre-processed before embedding for two reasons
  – Compressing data can reduce the amount of data to be hidden and therefore increase image quality
  – Encrypting data before hiding protects from illicit access and unscrambling
Fixed Embedding Method

- Assume original codebook is represented by $Y = (y_1, y_2, \ldots, y_n)$.
- Rearrange all codevectors in $Y$ such that $y_{2l-1}$ and $y_{2l}$ ($l = 1 - 2/n$), are as close as possible.
- Then partition $Y$ into two sub-codebooks $Y_0$ and $Y_1$, where $Y_0 = (y_1, y_3, \ldots, y_{n-1})$ and $Y_1 = (y_2, y_4, \ldots, y_n)$.
Fixed Embedding Method

• Assume that the hidden data $R$ is represented by a bit stream of length $w$

• $R = (b_1, b_2, \ldots, b_w)$ where $b_i \in \{0,1\}, \forall 1 \leq i \leq w$.

• For input vector $x$, use sub-codebook $Y_0$ if $b_i = 0$, and sub-codebook $Y_1$ is used if $b_i = 1$.

• Exchanging members of any pair $y_{2l-1}$ and $y_{2l}$ does not greatly reduce image quality since the members of the pair are so similar.
Fixed Embedding Method

Embedding Module
- Encoding
- Bit Stream: 011011001...
- Sub-codebooks: \( Y_0 \) and \( Y_1 \)

Extracting Module
- Decoding
- Codebook: \( Y \)
- Bit Stream: 011011001...

Figures represent the process flow of encoding and decoding.
Problems with Fixed Embedding Method

• Fixed Capacity
  – Each image block only embeds one bit, so that the total amount of data is limited to the number of image blocks

• Large Distortion
  – The distribution of all codevectors is not uniform, and therefore creating pairs of codevectors is not suitable for effective hiding
Adaptive Embedding Method

- Assume all codevectors are classified into $p$ disjoint clusters. All clusters can be represented by $S = \{S_1, S_2, \ldots, S_p\}$

- For each input vector $x_i$, the nearest codevector in codebook $Y$ can be found, and the cluster $S_{n(i)}$, where $n(i)$ is a function that ranges form 1 to $p$.

- Since all codevectors in cluster are similar, replacing one codevector with another codevector in the same cluster does not cause much distortion.
Adaptive Embedding Method

- This means that each input vector $x_i$ has $|S_{n(i)}|$ codevectors to choose from.
- When an image $I = \{x_1, x_2, \ldots, x_m\}$ is encoded by VQ, the list $L$ can be defined for all possible cases of selecting codevectors within a cluster.
- If $T$ is the length of ordered list $L$, then $T = \prod |S_{n(i)}|$
Adaptive Embedding Method

- A series number can be defined from 0 to T-1 for all members in the ordered list L.
- Since R is simply a bit stream, it can be transformed into an unsigned integer B.
Adaptive Embedding Method Algorithm

Input: Cover image $I$, codebook $Y = \{y_1, y_2, \ldots, y_n\}$, and secret bit stream $R = (b_1, b_2, \ldots, b_w)$

1. Transform $R$ into unsigned integer $B$.

2. Initialize $S = \{S_1, S_2, \ldots, S_n\}$, where $S_i = \{y_i\}$.

3. Combine two clusters in $S$ by using the nearest neighbor rule and update $S$.

4. Calculate the number $T$ of all possible assignments for $I$ based on cluster $S$.

5. If $2^w > T$, jump to step 3.

6. According to value of $B$, find the corresponding assignment of codevectors for all input value and output their indices.
Adaptive Embedding Method

encoding \rightarrow \text{adjustment} \rightarrow \text{decoding}

bit stream

011011001...

codebook

\begin{array}{c}
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx}
\end{array}

encoding module

extracting module

011011001...

codebook

\begin{array}{c}
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx} \\
\text{xxxx}
\end{array}
Adaptive Embedding Method Extraction

The merging number (number of clusters) needs to be passed to the receiver.

1. Extract the merging number and rebuild the clustering $S$.

2. Retrieve the matching for every index.

3. Calculate the series number $B$ from the selected codevectors based on clustering $S$.

4. Transform the unsigned integer $B$ to bit stream and output.
Experimental Results

- Compared their adaptive clustering embedding method (ACE) to fixed embedding methods.
- Fixed methods used two methods to group pairs:
  - mean gray-level embedding method (MGLE), which permutes codevectors according to their mean gray-scale values
  - Pairwise nearest-neighbor embedding method (PNNE), which just applies nearest neighbor until all codevectors are chosen.
Experimental Results

- **a** ‘Lena’
- **b** ‘F16’

---

- **a** Original image
- **b** VQ compressed image, PSNR = 31.24 dB
- **c** MGIE
- **d** PNNE
- **e** ACE
ACE Results

8 kbits 16 kbits 32 kbits 48 kbits 64 kbits 80 kbits
Embedding Distortion using Peak Signal-to-Noise Ration

<table>
<thead>
<tr>
<th>Capacity, Bits</th>
<th>MGLE Lena</th>
<th>MGLE F16</th>
<th>PNNE Lena</th>
<th>PNNE F16</th>
<th>ACE Lena</th>
<th>ACE F16</th>
</tr>
</thead>
<tbody>
<tr>
<td>8K</td>
<td>24.98</td>
<td>26.30</td>
<td>29.61</td>
<td>29.61</td>
<td>31.05</td>
<td>30.64</td>
</tr>
<tr>
<td>16K</td>
<td>22.49</td>
<td>22.79</td>
<td>28.35</td>
<td>27.98</td>
<td>30.78</td>
<td>30.59</td>
</tr>
<tr>
<td>32K</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>29.75</td>
<td>30.40</td>
</tr>
<tr>
<td>48K</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>28.16</td>
<td>29.90</td>
</tr>
<tr>
<td>64K</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>25.95</td>
<td>28.51</td>
</tr>
<tr>
<td>80K</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>22.91</td>
<td>26.38</td>
</tr>
</tbody>
</table>
Conclusion

- The adaptive embedding method is an improvement on previous VQ based data hiding techniques.
- The adaptive embedding algorithm has a higher hiding capacity and image quality than traditional techniques.