Introduction

The papers that we use as a main reference for this topic are “Data hiding in image and video – Part I: fundamental issues and solutions” and a dissertation on “Multimedia data hiding”. The authors of the first paper are Bede Liu and Min Wu, who is also the author of the other one. In this part I of two-part paper, we introduce brief backgrounds and fundamental issues of data hiding in image and video as well as general solutions. We start with a review of two types of embedding scheme and compare their performance in various aspects. After that, we study multi-level embedding system and how it handles a different Watermark-to-noise ratio (WNR). Finally, the non-stationary nature of visual signals lead us to another important issue called uneven distribution of embedding capacity which causes difficulty in data hiding. We will demonstrate the solution to this problem by using the combination of constant embedding rate (CER) with shuffling and variable embedding rate (VER) with control bits. The part II will discusses about applying these solutions to specific design problems for embedding data in gray scale and color image and video.

General background

We can consider data hiding or digital watermarking as a scheme to embed secondary data in digital media. There are many applications related to this such as ownership protection, access control, authentication, etc. Each of these applications may have different requirements. For example, it is necessary for ownership protection application to use a watermark that is very difficult to remove and robust against common
processing and attack while embedding movie sub-title focuses more on embedding capacity and perceptual quality of marked media. In general, data hiding application always involves three requirements that are perceptual quality, embedding capacity and robustness. A typical data hiding framework is illustrated in figure 1. Watermarking technique can be divided into 2 categories. One is to embed watermark in spatial domain and the other is to embed watermark in transformed domain such as DCT, DFT and DWT domain. In addition, we can also classify them into 2 groups that are Type I (Additive embedding) and Type II (Relationship enforcement embedding) which will be discussed in next section.

![General framework of data hiding systems](image)

**Fig. 1.** General framework of data hiding systems

**Two types of data embedding**

In first category (Type I – Additive embedding), the secondary data is added to host signal as illustrated in figure 2. We can derive the relation of original media signal ($I_0$), marked media signal ($I_1$) and the embedded bit ($b$) as $I_1 - I_0 = f(b)$ and $I_0$ can be regarded as major noise source (host interference). Therefore, the knowledge of $I_0$ will greatly enhance detection performance by eliminating the interference.
In second category (Type II – Relationship enforcement embedding), a original media signal ($I_0$) is partitioned into subsets which are mapped by function $g(.)$ to the set of value taken by the secondary data as illustrated in figure 3 and the relation between marked media signal ($I_1$) and the embedded bit ($b$) is $b = g(I_1)$. Hence, in this case, we don’t need the knowledge to $I_0$ because the information about $b$ is carried in $I_1$. Throughout this paper, Odd-Even embedding will be used as example of type II. In this
embedding scheme, we simply choose an even number as \( I_1 \) to embed a “0” and odd number to embed “1” as shown if figure 4.

For type I embedding, it has excellent robustness and invisibility when the original host media is available but under blind detection, we have to trade off perceptual quality and embedding capacity for robustness by using higher WNR and longer watermarking signal for representing one bit, respectively. For type II, there is no interference from host media. Therefore, we can encode one bit with smaller number of host components and this results in better embedding capacity. To demonstrate the trade-off between perceptibility and robustness of type II, let’s consider the odd-even embedding model. First of all, MSE of odd-even embedding is equal to \( Q^2/3 \) where \( Q \) is quantization size. When we increase \( Q \), we actually introduce more distortions to media but we also obtain a better robustness because of a larger tolerance zone \((-Q/2,Q/2)\). In general, type II is suitable for high data-rate data hiding application that do not have to survive noise.

**Quantified capacity study**

The channel model for type I is continuous input and continuous output (CICO). The additive noise comes from interference from host signal and noise due to processing and distortion. Using Shannon channel capacity and assuming that host signal and processing noise are independent and both are i.i.d Gaussian distributed, we obtain the embedding capacity as follow:

\[
C_{\text{CICO}} = \frac{1}{2} \log_2 \left( 1 + \frac{A^2}{\sigma_I^2 + \sigma^2} \right)
\]

where \( \sigma_I^2 \) is the power of the original host signal, \( A^2 \) is the power of the embedded signal and \( \sigma^2 \) is the power of additive processing noise (In general, \( \sigma_I^2 \gg \sigma^2 \))

The channel model for type II can be considered as a discrete input and discrete output (DIDO) binary symmetric channel (BSC). The capacity of this type of channel is given by

\[
C_{\text{DIDO}} = 1 - h_p \quad \text{and the binary entropy is} \quad h_p = p \log \left( \frac{1}{p} \right) + (1 - p) \log \left( \frac{1}{1 - p} \right)
\]

where \( p \) is the probability of bit error for additive white Gaussian noise (AWGN)
\[ p = \min \left\{ \frac{1}{2}, 2\sum_{k=0}^{\infty} Q\left(\frac{4k+1}{2\sigma}Q\right) - Q\left(\frac{4k+3}{2\sigma}Q\right) \right\} \]

Now, we will compare the embedding capacity of type I and II by fixing MSE introduced by embedding process to \( E^2 \) – (fix the perceptual quality of media). Then, we have these parameters for testing and the result is shown in figure 5.

Type I: Power of embedded signal \( \Rightarrow E^2 \)
- Host interference \( \Rightarrow \sigma_1^2 = 10E \)
- Gaussian processing noise \( \Rightarrow \sigma^2 (\sigma_1^2 >> \sigma^2) \)

Type II: MSE = \( E^2 = \frac{Q^2}{3} \Rightarrow Q = \sqrt{3}E \)

![Figure 5: Embedding capacity of type I and II](image)

As we can see from figure 5, we conclude that type I is suitable for strong noise condition while type II is useful under low noise condition.
**Multi-level embedding**

An embedding scheme is usually designed to target specific WNR. This might lead to no extractable data when actual noise condition is much stronger than design value (case: $C_{1,2}$ in figure 6) or we might simply waste embedding capacity on robustness when actual noise condition is lower than what we expect (case: $C_{1,1}$ in figure 6). In order to overcome this problem, we introduce multi-level embedding technique.

![Figure 6](image1.png)  
**Figure 6:** Left: Embedding capacity versus WNR and Right: Embedding system ($C_{1,1}$ and $C_{1,2}$) designed to target specific WNR ($X_1$ and $X_2$, respectively)

![Figure 7](image2.png)  
**Figure 7:** Left: Two-level embedding scheme ($C_{II}$) and Right: $\infty$-level embedding scheme ($C_{\infty}$)
First, let’s consider two-level embedding scheme. The main concept is to allow fraction $\alpha_1$ of embedded data to survive WNR of $X_1$ while all embedded data survive a higher WNR of $X_2$ as shown in figure 7. As we can see, $C_{II}$ achieve the better embedding capacity than $C_{I,1}$ when WNR is more than $X_2$. When WNR is less than $X_1$, it also has a better embedding capacity than $C_{I,2}$. This ideal can be applied to create any N-level embedding system but the degradation will occur when we use too many levels as shown in figure 7.

**Uneven embedding capacity**

The unevenly distributed embedding capacity comes from the non-stationary nature of perceptual sources. For example, changes made in smooth area are easier to be perceived than those in texture area. Our goal is to obtain the highest embedding capacity by embedding as many bits as possible in each region (host media is segmented into many regions). By doing so, we need to convey size information about how we distribute embedded data. This overhead information might be very large and in turn, reduce the actual embedding capacity. We can overcome large overhead problem by using constant embedding rate (CER) to embed a fix number of bits in each region but we have to keep in mind that this fix number must be small while the size of each region must be large enough to allow all segments to have enough embeddable coefficients. Nevertheless, increasing block size alone is usually not enough to satisfy this condition. Let’s consider image in figure 8. By using blockwise DCT transform of size 8 x 8 and applying perceptual threshold to AC coefficients, we can determine all locations of smooth block, segment with no embeddable coefficient. These smooth blocks are labeled by black pixel in figure 8 and they correspond to a very smooth area like sky and shadow. We found that around 20% of all blocks are smooth block. Now, let’s increase the block size to 16 x 16. It turns out that 15% of all blocks are smooth block. This is due to the fact that smooth blocks tend to be together and result is shown in figure 9. Therefore, we can’t simply increase the block size to solve the problem. We need to equalize the embedding capacity using shuffling.
Shuffling is a technique to equalize the number of embeddable coefficients in each block and it can be done as shown in figure 10. There are many kinds of shuffling based on permutation function but we will focus our interest on complete random permutation case which all permutations are equiprobable. For this case, we have

\[
E \left[ \frac{m_r}{N} \right] = \frac{\binom{s}{r} \binom{s-q}{n-r}}{\binom{s}{n}}
\]

\[
Var \left[ \frac{m_r}{N} \right] = \frac{1}{N} \cdot \frac{\binom{s}{r} \binom{s-q}{n-r}}{\binom{s}{n}} + \left( 1 - \frac{1}{N} \right) \frac{\binom{q}{r} \binom{q-2q}{n-2r}}{\binom{s}{n}} - \left[ \frac{\binom{q}{r} \binom{s-q}{n-r}}{\binom{s}{n}} \right]^2
\]
where \( S \) is total number of coefficients, \( N \) is total number of block, \( q = \frac{S}{N} \) is number of coefficients in each block, \( m_r \) is the number of block that has \( r \) embeddable coefficients and \( n \) is total number of embeddable coefficients. For testing image in figure 8, we have 

\[
E\left[\frac{m_r}{N}\right] = 0.002\% \approx 0.086 \quad \text{blocks and most segments have 5-15 embeddable coefficients.}
\]

The simulation result is shown in figure 11.

Figure 10: shuffling technique

Figure 11: Histogram of embeddable coefficients (case: normal and shuffling)
What shuffling really does is to allocate embeddable coefficients from texture area to smooth region. Thus, we unintentionally increase the sensitivity of those texture areas but this is acceptable for some applications that users benefit from data hiding like multi-language sub-title (in such application, intentional attack is irrelevant). We have already discussed about CER which is suitable when the overhead information is large. For the case that overhead data is small, variable embedding rate (VER) is more suitable. The main issue for VER is how we convey this side information to detector. There are several methods to do this. For instance, we can design some parts of embedded data to be pre-determine, i.e., decoder decode this pre-determine data using all possible mechanisms and decide which one was used in encoding according to the result. This very same idea can be applied to determine the block size that was used in encoding.