Data hiding method using image interpolation

Ki-Hyun Jung a, Kee-Young Yoo b,*

a School of Computer Information, Yeungjin College, 218 Bokhyun-Dong, Buk-Gu, Daegu 702-721, Republic of Korea
b Department of Computer Engineering, Kyungpook National University, 1370 Sankyuk-Dong, Buk-Gu, Daegu 702-701, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 8 January 2008
Accepted 4 June 2008
Available online xxxx

Keywords:
Data hiding
Interpolation
Image processing

A B S T R A C T

Data hiding is to conceal the existence of secret data. A reversible data hiding method can extract the cover image without any distortion from the stego-image after the hidden data have been extracted. This paper proposes a new interpolation and a data hiding method. The proposed scaling-up neighbor mean interpolation method has a low-time complexity and high-computation speed. The proposed data hiding method is based on interpolation. Comparison on data hiding methods is divided into reversible or not. Our experimental results show the proposed method can embed a large amount of secret data while keeping a very high visual quality, the PSNR is guaranteed to be higher than 35 dB compared with other reversible data hiding methods. And also capacity is larger than any other reversible data hiding methods and comparable to other data hiding methods.

1. Introduction

Data hiding conceals the existence of secret messages while cryptography protects the content of messages. The word steganography is derived from Greek and it literally means “covered writing”, the art of hiding information in ways that prevent detection. It includes a vast array of secret communication methods that can conceal the very existence of messages [1–3]. Information can be hidden in many different ways. In order to hide information, straight message insertion may encode every bit of information in the cover data or it may selectively embed messages in noisy areas that draw less attention. Messages may also be scattered randomly throughout the cover data. There are a number of ways to hide information; the most common methods are the least significant bit (LSB) insertion, masking and filtering and algorithms and transformations [1].

Reversible data hiding methods enable the exact recovery of the original host signal upon extraction of the embedded information [4]. Celik et al. classify the reversible data hiding methods into two types: additive spread spectrum techniques and embedding by modifying selected features of the host signal.

Sampling and the interpolation of signals are important processes in signal theory, and they appear frequently in many applications [5]. There are a number of techniques that can be used to enlarge an image. Such methods generally have a trade-off between speed and the degree to which they reduce visual artifacts.

Image interpolation addresses the problem of generating high-resolution images from its low-resolution. The model that is employed to describe the relationship between high-resolution and low-resolution pixels plays a critical role in the performance of an interpolation algorithm. Conventional linear interpolation schemes, based on space-invariant models, fail to capture the fast-evolving statistics around edges and annoying artifacts. Linear interpolation is generally preferred, not for performance, but for computational simplicity [6].

This paper introduces a traditional scaling-up method and a new interpolation method is proposed; one that focuses on high speed and low complexity. The proposed data hiding method is demonstrated using the proposed neighbor mean interpolation method. The proposed neighbor mean interpolation method is more efficient, therefore the proposed data hiding method is also efficient.

This paper is organized as follows. Section 2 reviews traditional and well-known interpolation methods. The proposed interpolation method, which has low-time complexity and can operate at high speed, is outlined. In Section 3, the details of our newly-proposed data embedding scheme is described. In Section 4, the experimental results are presented and discussed. Finally, the concluding remarks are presented in Section 5.

2. Interpolation methods

In this section, commonly used interpolation methods are reviewed, and the proposed interpolation method, called neighbor mean interpolation, is introduced. This interpolation method focuses on high speed and low-time complexity. In Section 3, the secret data hiding method is demonstrated using the proposed neighbor mean interpolation method.
2.1. Traditional interpolation methods

The capability to interpolate images digitally to a higher resolution, with good image quality, is important in many applications of electronic imaging. Sharpness of edges and not being constrained to artifacts are two critical factors regarding the perceived quality of the interpolated images. Ease in computation is also an important factor [7]. Image interpolation methods are as old as computer graphics and image processing. Simple interpolation algorithms, such as the nearest neighbor and linear, have been used for resampling [8]. By the nearest neighbor method, one can find the closest corresponding pixel in the source image \( p(i,j) \) for each pixel in the destination image pixel \( p'(i,j) \). Nearest neighbor interpolation suffers from normally unacceptable aliasing effects with regard to enlarging and reducing images. Bilinear interpolation determines the grey level from the weighted average of the four closest pixels to the specified input coordinates, and it assigns a value to the output coordinates. This method generates an image that has a smoother appearance than nearest neighbor, but the grey level values are altered in the process, thereby resulting in blurring or loss of image resolution. Due to these changes in gray level values, image classification processes should be performed before interpolation. Bilinear interpolation requires 3 or 4 times higher computation time than that of the nearest neighbor method.

2.2. The proposed neighbor mean interpolation

The proposed neighbor mean interpolation (NMI) method uses neighboring pixel values to calculate the mean, and then the calculated mean value is inserted into a pixel that has not been allocated yet. In general, we can get more high-resolution pixels when a neighboring pixel value is referenced in order to calculate a value that is to be allocated, but time complexity is higher when the number of referenced pixel is higher. The scaling-up method decides what application to which it should be applied. In this paper, a new interpolation method called neighbor mean interpolation is proposed. On the pixel \( p(i,j) \), the output pixel, \( p'(i,j) \) is defined as followed, where \( 0 \leq i \leq m \) and \( 0 \leq j \leq n \).

\[
p'(i,j) = \begin{cases} 
\frac{p(i,j) + p(i,j+1)}{K} & \text{if } i < K \text{ and } j < K \text{ and } i+j < K \\
\frac{p(i-1,j) + p(i+1,j)}{K} & \text{if } i-K < 0 \text{ or } j-K < 0 \\
\frac{p'(i-1,j) + p'(i+1,j) + p'(i,j+1) + p'(i,j-1)}{K+1} & \text{otherwise} 
\end{cases}
\]

Fig. 1 shows an example of how to process the neighbor mean interpolation. The resulting image is scaled two times more. The pixel \( p'(0,0) \) and \( p'(2,2) \) are the same value with \( p(0,0) \) and \( p(2,2) \), respectively. In the case of \( i < j \), \( p'(0,1) \) is calculated from \( p(0,0) + p(0,2)/2 \) operation. When \( i < j \), \( p'(1,0) \) is the result of \( p(0,0) + p(2,0)/2 \). Finally, \( p'(1,1) \) is derived from \( p(0,0) + p'(0,1) + p'(1,0)/3 \).

The neighbor mean interpolation is similar to bilinear interpolation, but this method has less blurring and greater image resolution.

In Section 4, the experimental results demonstrate that the PSNR of the proposed neighbor mean interpolation method is more efficient than nearest neighbor interpolation and bilinear interpolation.
methods, so we use the resulting images that are produced by the NMI method with a cover image.

3. The proposed data hiding method

In this section, we propose a data hiding method using the neighbor mean interpolation method. Before hiding secret data, the proposed method utilizes the resulting images of the neighbor mean interpolation method.

The sequence of data hiding can be in zig-zag, left-to-right and upper-to-down directions. Before secret data are embedded, the host image is partitioned into four-pixel, non-overlapping, consecutive blocks by zig-zag scanning as shown in Fig. 3.
For every four non-overlapping consecutive pixel values, i.e., \( p(i,j), (i+1,j) \), \( p(i+1,j+1) \), \( p(i+j+1) \), and \( p(i+j+1) \), the corresponding stego-image pixel values are \( p'(i,j) \), \( p'(i+1,j) \), \( p'(i+1,j+1) \), and \( p'(i+j+1) \). In here, we embed secret data into three pixels except for \( p(i,j) \) pixel.

The PSNR value is dependent on what scaling-up method is used. Instead of using the proposed neighbor mean interpolation, other methods could be applied to make a cover image. This method is used for a comparison of PSNR and capacity.

This new proposed method does not require a cover image. The cover image and secret data can be extracted by stego-image. Also, these can be calculated by simple operations. Since the proposed method uses a cover image that produced by the scaling-up interpolation method, more secret data can be hidden that of the other data hiding methods. First, we get a scaling-up image by using the neighbor mean interpolation method. Then, for every four non-overlapping consecutive pixel values, a difference value \( d \) is computed as Eq. (2), where \( 0 \leq x \leq 127 \) and \( \beta \), \( \delta \) value is 0 or 1, respectively. Herein, \( d \) is calculated as zero when \( \beta = \delta = 0 \).

\[
d = (p(K \cdot x + \beta, K \cdot y + \delta) - p(K \cdot x, K \cdot y))\quad (2)
\]

The number of bits, say \( n \), which can be embedded in this pixel, is calculated by

\[
n = \lceil \log_2 |d| \rceil \quad (3)
\]

A sub stream with \( n \) bits in the embedding data is selected and converted to integer value \( b \). Then, a new pixel \( p'(i,j) \) is computed as follows.

\[
p'(i,j) = p(i,j) + b\quad (4)
\]

Fig. 3 shows an example of the data hiding process. Let pixel values \( p(0,0), p(0,1), p(1,0) \) and \( p(1,1) \) be 100, 139, 143 and 112, respectively. We can calculate a difference value for each pixel. By Eq. (2), the difference values are 0, 39, 43 and 12, respectively. Then, we can calculate the length of hiding for four-pixel blocks. The result is 5, 5, 5 and 3, respectively. Finally, the integer value \( b \) for the secret bits are 10, 10, 10, and 0.1. So, the pixel of stego-image is 156, 164, 164 and 113. In here, pixel \( p(0,0) \) is reserved.

In the extraction process, we do not have to refer a cover image. We can get the cover image and secret data by using Eq. (5) for all pixels. The secret data are extracted using a simple arithmetic expression as defined below, where \( 0 \leq i \leq 3 \) and \( x,y=0,1,\ldots,127 \) and \( K \) is defined to 2.

\[
b = \begin{cases} 
\frac{p(i,j) - p(i+1,j) - p(i+1,j+1) + p(i+j+1)}{K} & \text{if } i=K \text{ or } j=K, \\
\frac{p(i) - p(i+1) + p(i+1,j) - p(i+j+1)}{K} & \text{if } i=K \text{ and } j=K, \\
\frac{p(i,j) - p(i+1,j) + p(i+1,j+1) - p(i+j+1)}{K} + p(i+j+1) & \text{if } i=K \text{ and } j=K, \\
\frac{p(i,j) - p(i+1,j) + p(i+1,j+1) - p(i+j+1)}{K} & \text{otherwise}
\end{cases} \quad (5)
\]

Fig. 4 shows an example of the data extracting process. Assume that original image pixels have 46, 112, 210 and 90. The cover image is generated by neighbor mean interpolation method as shown Fig. 4.

In the first, we calculate the difference value for the three neighboring pixels, excluding \( p(0,0) \); this is because \( p(0,0) \) does not include secret bits. Then, the length of hiding is calculated for each secret bit. The result is 5, 5, 5 and respectively. Next, the secret bits are divided into embedding lengths and converted to integer value \( b \). This calculated value is 19, 22 and 5 which correspond to \( p(0,1), p(1,0) \) and \( p(1,1) \). Finally, the four stego-image pixels are 46, 98, 150 and 89, respectively.

A cover image can be acquired by using stego-image without needing a reference to a cover image. For example, \( p(0,1) \) is calculated by \( p(0,0) + p(0,2)/2 = (46 + 112)/2 = 79 \) and the secret data embedding in \( p(0,1) \) is 98 – 79 = 19. \( p(0,1) \) is also calculated by \( p(0,0) + p(2,0)/2 = (46 + 210)/2 = 128 \) and the secret data embedding in \( p(0,1) \) is 150 – 128 = 22. Finally, \( p(1,1) \) is obtained by \( (2 \cdot p(0,0) + p(0,2))/2 + p(2,0)/2)/3 = (46 + 112 + 210)/3 = 253/3 = 84 \) and secret data embedding in \( p(1,1) \) is 89 – 84 = 5. This means that the cover image and secret data can be extracted by using stego-image only.

### Table 1

<table>
<thead>
<tr>
<th>Method</th>
<th>Lena</th>
<th>Baboon</th>
<th>Airplane</th>
<th>Peppers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest neighbor interp.</td>
<td>22.56</td>
<td>16.91</td>
<td>21.03</td>
<td>22.22</td>
</tr>
<tr>
<td>Neighbor mean interp.</td>
<td>26.82</td>
<td>20.09</td>
<td>24.66</td>
<td>25.99</td>
</tr>
</tbody>
</table>

### Table 3

<table>
<thead>
<tr>
<th>Method</th>
<th>Pure payload</th>
<th>PSNR (dB)</th>
<th>Pure payload</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOISINGER-ET-AL.</td>
<td>&lt;1024</td>
<td>–</td>
<td>&lt;1024</td>
<td>–</td>
</tr>
<tr>
<td>MACQ AND DWEYAND</td>
<td>&lt;2048</td>
<td>–</td>
<td>&lt;2048</td>
<td>–</td>
</tr>
<tr>
<td>FRIEDRICH-ET-AL.</td>
<td>1024</td>
<td>–</td>
<td>1024</td>
<td>–</td>
</tr>
<tr>
<td>GOELJAN-ET-AL.</td>
<td>24108</td>
<td>39.00</td>
<td>2905</td>
<td>39.00</td>
</tr>
<tr>
<td>VLEESCHOUWER-ET-AL.</td>
<td>1024</td>
<td>30.00</td>
<td>1024</td>
<td>29.00</td>
</tr>
<tr>
<td>XUAN-ET-AL.</td>
<td>85507</td>
<td>36.60</td>
<td>14916</td>
<td>32.80</td>
</tr>
<tr>
<td>CELIK-ET-AL.</td>
<td>74600</td>
<td>38.00</td>
<td>15776</td>
<td>38.00</td>
</tr>
<tr>
<td>NOLI-ET-AL.</td>
<td>54690</td>
<td>48.20</td>
<td>5421</td>
<td>48.20</td>
</tr>
<tr>
<td>PROPOSED</td>
<td>200688</td>
<td>41.46</td>
<td>425199</td>
<td>35.46</td>
</tr>
</tbody>
</table>

Please cite this article as: K.-H. Jung, K.-Y. Yoo, Data hiding method using image interpolation, Computer Standards & Interfaces (2008), doi:10.1016/j.csi.2008.06.001.
4. Experimental results

Overall comparison between the existing data hiding methods and the proposed method are presented in Fig. 5. In the proposed method, an input image is scaling-down and then generates a cover image using interpolation methods. Also the original image can be used as input image for satisfying the reversible data hiding. This cover image is used for embedding secret data. An input and cover image shown in Fig. 5 are used for comparison results between traditional interpolation methods and the proposed neighbor mean interpolation. And the proposed data hiding method and other embedding methods are compared, where an input image and the generated stego-image are used for PSNR.

In our experiments, the four 512×512 gray images were used as cover images as shown in Fig. 6. This input image is scaled down to generate a 256×256 image and the generated 256×256 image is used as input source of the interpolation method. And then a 512×512 cover image is used after applying interpolation algorithm. We employed the peak signal-to-noise ratio (PSNR) as a measure of the quality of the interpolation and hidden data results. The original image can be exactly extracted when the process of the scaling-down is skipped. In our method, the quarter size of the cover image can be recovered when the scaling-down process is executed. We inserted this process to emphasize that the difference between an original image and a cover image can’t be discriminated when the PSNR is sustained above 30 dB in human visual system.

In the interpolation experimental results, four cover images are used and compared with regarding to PSNR value and capacity. In the reversible data hiding experimental results, the four cover images are used for comparison of pure payload. Also four cover images are used on the comparison of other data hiding methods that are not reversible.

4.1. Interpolation results

The 256×256 image is used to produce a 512×512 image. The PSNR is compared with the original 512×512 image. The 256×256 image is produced by scaling-down.

Table 1 shows a comparison of the PSNR values for the three interpolation methods: two traditional interpolation methods and the proposed neighbor mean interpolation method.

The results show that the NMI method is a little better than the other methods. Results demonstrate that the proposed NMI method is better than other two interpolations. We can apply other interpolation methods for the hiding of data. This paper demonstrates that the scaling-up algorithm, like interpolation, can be utilized for the hiding of data. We use a simple interpolation, but the image processing field is mainly treated.

4.2. Data hiding results

Regarding the three interpolation methods, the data hiding method utilizes a cover image that is produced by the neighbor mean interpolation method.

Table 2 shows the results of a comparison of LSB substitution methods that are not reversible. Note that capacity represents amount of maximal capacity.

In our proposed data hiding, the proposed irreversible method is applied 3-bits LSB substitution to embed secret data more for unchanged first pixel on the neighbor four pixels. Table 3 shows the pure payload and PSNR compared with other reversible data hiding methods [9–16]. Note that the pure payload and the PSNR of other reversible data hiding methods referenced by Ni et al.’s results [16]. Goljan et al.’s method is estimated on average values when the embedding amplitude is four, and the data of Vleeschouwer et al.’s method is obtained by using Ni et al.’s implemented results. Results show that our proposed method has higher PSNR value and larger data embedding capacity. Note that the listed pure payload is the amount secret data which overhead information has been excluded, but our proposed method does not need additional information for data embedding. The PSNR on the proposed method is average value for four secret images (Fig. 7).

5. Conclusions

We have proposed an interpolation method based on the traditional scaling-up method. The proposed neighbor mean interpolation method had a low-time complexity and high-calculation speed. And then, the data hiding method, based on the proposed neighbor mean interpolation, was proposed, which applicable to data hiding and reversible data hiding method.

The neighbor mean interpolation method used a neighboring pixel values to calculate the mean value, and then the value was inserted by a pixel that has not been allocated yet. The PSNR of the proposed method was comparable with those of the nearest neighbor and bilinear interpolation methods. Results demonstrated that the proposed NMI method was better than the other two interpolations.

The data hiding method was proposed, in which a cover image was used as an input for the result of two times scaling-up. The cover image and secret data could be extracted from stego-image without the need for any extra information.

We showed that the proposed method is comparable with those whether are reversible or not. In comparison with irreversible data hiding methods, it is shown that our method is changeable to irreversible data hiding method to embed secret data more. Especially, it was possible to embed 2 times much more when produced a large difference between neighboring pixels like as Baboon image. In other comparison with reversible data hiding methods, our experimental results have shown that the proposed method provided a better way to hide data. It could embed 112 KB–400 KB more compared with other reversible method that can contain highest capacity and keep a high visual quality, which the PSNR was guaranteed to be higher than 35 dB.

References

Ki-Hyun Jung received his B.Sc. degree in Computer Engineering from Kyungpook National University in 1995 and the M.Sc. degree in Computer Engineering from Kyungpook National University in 1997, South Korea. He has been employed as a senior researcher at Agency of Defense Development, South Korea. He received the Ph.D. degree in Computer Science from Kyungpook National University in 2007, South Korea. Currently, he is an Assistant Professor at the School of Computer Information, Yeungjin College, South Korea. His current research interests are information hiding, watermarking, cryptography, network security, game and mobile programming, and virtual reality.

Kee-Young Yoo received his B.Sc. degree in Education of Mathematics from Kyungpook National University in 1976 and the M.Sc. degree in Computer Engineering from Korea Advanced Institute of Science and Technology in 1978, South Korea. He received the Ph.D. degree in Computer Science from Rensselaer Polytechnic Institute in 1992, New York, USA. Currently, he is a Professor at the Department of Computer Engineering, Kyungpook National University, South Korea. His current research interests are cryptography, authentication technologies, smart card security, network security, DRM security, and steganography. He has published over a hundred technical and scientific international journals on a variety of information security topics.