用於網頁版權保護的資訊隱藏方法

A Steganographic Method for Copyright Protection of Web Pages

Ya-Hui Chang(張雅惠) and Wen-Hsiang Tsai(蔡文祥)

Department of Computer & Information Science

National Chiao Tung University

Emails: gis90577@cis.nctu.edu.tw & whtsai@cis.nctu.edu.tw

摘要

本篇論文提出一種在 HTML 檔案中藏入序號或是 logo 影像，以達到網頁的版權保護的新方法。我們利用在 Web 瀏覽器上的可視空白來將版權資料編碼並隱藏進 HTML 檔案的實際內容中，如此一來，就可保護網頁的實際內容，避免被非法使用。因為隱藏的版權資料是經由空白編碼，很容易遭毀壞，為了解決這個問題，在隱藏資料的過程中，我們將版權資料複製成多份，並藏入檔案中所有可隱藏的地方。如此，即使有一份或是多份的版權資料遭受到損壞，仍然有其他份完整的版權資料可用來證明網頁實際內容的版權。實驗結果顯示了我們所提出的方法確實可行與實用。

關鍵詞：版權保護、網頁、HTML 檔案、訊息隱藏(Steganography)

Abstract

A novel method for embedding a serial number or a logo within an HTML document for copyright protection of Web pages is proposed. Pseudo-spaces that are visible in Web browsers are utilized to encode copyright data into the actual text of an HTML file. Then the actual text can be protected from being directly copied by browsers for unauthorized uses. Since the copyright data are encoded by spaces, they can be destroyed easily. To solve the problem, the copyright data, when embedded, are duplicated as many times as possible to fill all the data-embeddable places in the protected HTML document. Thus, even if one or more copies of the copyright data are destroyed, there are still other copies for use to prove the copyright of the Web-page content. Experimental results show the feasibility and practicability of the proposed approach.

Keywords：Copyright protection, Web pages, HTML file, Steganography.

1. Introduction

Nowadays, with the popularity of computer networks, various kinds of digital media like image, text, and video can be distributed speedily through the public network environment. Because they might be copied for unauthorized uses by illicit users, many data hiding techniques for copyright protection have been developed to solve this problem [1-3]. Though many copyright protection methods have been proposed for images and videos, there is yet no effective method for copyright protection of text-type files, like Web pages. This weakness comes from the facts that the content of Web pages can be edited easily, and that there lacks redundant information in Web pages to hide copyright data. As a consequence, people may search for any interesting information by browsing Web pages on the Internet, and copy it directly for misuses.

Text-type files contain less redundant information for hiding data, compared with a picture or a video. Data hiding methods for text documents try to encode information directly in the text itself, such as exploiting the natural redundancy of languages, or in the text format, e.g., by adjusting the inter-word or interline space [4-5]. But the encoded information can be destroyed easily by editing the text content. Any people thus may take part of the content of a Web page and claim it to be his/her own. It is thus difficult to protect the copyright of a Web page. In this study, we propose a novel watermarking method for proving the copyright of Web pages.

1.1 Proposed Ideas

In this study, we assume that a protected Web page might be misappropriated under two situations. One is that an illicit user takes the protected Web page directly for misuses. The other is that an illicit user copies the actual text of the protected Web page when it is shown by a Web browser and uses the text copy for illegal purposes or transforms the text copy into another Web page. In this paper, we use the term *actual text* of an HTML document to mean the text that is displayed by a Web browser. No matter which situation is, it is desired to verify the copyright of the protected Web page in a certain way. Since the tags of HTML files are used to tell browsers how to display their actual text, we assume that the actual text is the most important part of a Web page and has to be protected. In this study, we adopt the idea of using spaces to embed copyright data into the actual text of HTML files for copyright proving.

Because browsers will ignore extra spaces, the hidden copyright data will not be displayed by Web browsers if they are encoded by *real spaces*. Therefore, the second situation described above cannot be solved because the hidden copyright data will not be replicated into the text copy. So, *special spaces* that can be displayed by browsers are introduced in this study to encode copyright data for solving the two situations. That is, the copyright of actual texts of protected Web pages, or copied actual texts from displayed Web page contents can be proved by our method.

In our method, two techniques of embedding copyright data for ownership verification of HTML documents are proposed. The first is to use a serial number as the copyright data, while the second is to use a binary logo image. The details are described in the following. Experimental results showing the feasibility of the proposed method will also be shown.

2. Encoding of Copyright Information

Since people might copy the actual text of an HTML document directly from a Web browser window, copyright data must be embedded in such a way that it can be copied simultaneously, as mentioned previously. Inserting real spaces into HTML documents directly does not work because browsers will not show the hidden data in the displayed Web pages. Hence, we propose a novel method for encoding copyright data in this study. In addition to the copyright data to be embedded, we propose the use of a specially-designed mark M and use different techniques to encode the copyright data and the mark into the actual text of an HTML document. The meaning of the specially- designed mark M will be explained in detail later.

The proposed technique for encoding copyright data is to insert spaces between words or sentences of the actual text. Because Web browsers ignore extra real spaces of HTML documents when displaying them, hidden data that are encoded by real spaces will be invisible in Web browser windows. Therefore, the specific string " " instead of a plain real space is used in this study to encode copyright data. The string " " in an HTML file, which can be viewed as a pseudo-space, will appear to be a real space when it is displayed in Web browsers. In this way, copyright data are encoded by allowing either one or two spaces between words or sentences of the actual text. More specifically, since there is usually only one real space between two words or two sentences, a data bit "1" is encoded by inserting additionally a pseudo-space before the already-existing real space, while a "0" is regarded to exist if there is just one space between two words or sentences (see Fig. 1 for an example). That is, two spaces, which are composed of one pseudo-space and one real space, between two words or sentences are interpreted as a "1." And a single real space is interpreted as a "0." Because a pseudo-space will be a real space when displayed in browser windows, an illicit user will get two real spaces after copying the actual text of an encoded "1" from a browser window. Therefore, two real spaces in a copied text should be interpreted as a "1" in the decoding process, and a real space as a "0."

The second proposed technique for encoding the previously-mentioned special mark M is to place a Big5 space between two words or sentences of the actual text (see Fig. 2 for an example). The mark M is used as a synchronization signal to indicate the beginning or the end of the bit stream of a copy of embedded copyright data. Because of this important function, it must be displayed in browser windows, yet being imperceptible to readers; that is, it should be *steganographic* when displayed. Also, it should not conflict with the encoded copyright data when the copyright data are destroyed. If we use two or more pseudo-spaces to encode M, a conflict will arise between the marks M and the copyright data when someone modifies the number of pseudo-spaces. On the other hand, the use of too many extra spaces will also arouse readers' notices when the spaces are seen in the Web page. Therefore, a Big5 space is a good choice and so is utilized to encode M in this study. By this idea, we can embed as many copies of the copyright data as possible in a Web page, using the mark M as synchronization signals, resulting in a more

effective and robust copyright protection technique, as will be explained in more detail subsequently.

proposed ideas. proposed ideas
(a) (c)
proposed proposed ideas
ideas.
(b) (d)

Fig. 1 An example of data hidden by first proposed technique. (a) a segment of HTML. (b) Encoded HTML (a code of "1" is inserted). (c) The display of (a) in browsers. (d) The display of (b) in browsers.

proposed ideas. proposed ideas
(a) (c)
proposed proposed ideas
ideas
(b) (d)

Fig. 2 An example of hiding the mark M by second proposed technique. (a) A segment of HTML. (b) Encoded HTML (a code of M is inserted). The display of (a) in browsers. (d) The display of (b) in browsers.

3. Serial Number Embedding and Extraction

A serial number may be used as copyright data and can be embedded into an HTML document for copyright protection by the encoding method described in the previous section. The data-embedding capacity of an HTML file, when compared with that of an image or a video, is relatively small, so it is appropriate to use a serial number, which is small in size, as copyright information. The processes of serial number embedding and extraction are described as follows.

3.1 Serial Number Embedding

Because the content of a Web page with hidden copyright data is text in type, it may be deleted or altered at any position. And because an illegal user may copy only part of the content of a Web page, the hidden copyright data might be just partially copied and so destroyed. In the proposed method, as many copies as possible of the serial number are embedded to fill all data-embeddable places in the HTML text in order to verify the copyright in the aforementioned situations. In this way, when one or a few copies of the hidden serial number are destroyed, there hopefully will still be other copies available for use to verify the copyright. Hence, we must know where each copy of copyright data is started and ended to avoid the undesired case that one destroyed copy may result in incorrect extraction results of the other copies. In the proposed embedding process, the mark M is inserted before the start and after the end of one copy of the hidden serial number. With M used as synchronization signals in this way, each copy of the serial number, if not destroyed, can then be extracted correctly.

On the other hand, in the proposed method a key specified by the owner of a Web page is used to disarrange the serial number before it is hidden in a Web page, so that later on a person who claims to have the copyright of the actual text must provide a correct key to extract the serial number correctly for ownership proving. Furthermore, since many copies of the serial number are embedded, the owner can prove as well the copyright of even part of the Web-page content. On the other hand, some of the extracted copies might have flip-errors possibly due to illegal content editing. To solve this problem, a voting scheme is utilized in this study to recover the serial number correctly. The details of the proposed embedding algorithms are described as follows. The input is a serial number N and a HTML document H.

Step 1: Convert the serial number N into binary form and select a key to disarrange the resulting binary data by a random number generator with the key as input. Denote the result as $E = \{e_1, e_2, ..., e_n\}$ with n bits in length.

Step 2: Attach M to E by inserting M before and after E to produce a copy of complete copyright data $C = \{\mathrm{M}, e_1, e_2, ..., e_n, \mathrm{M}\}$.

Step 3: Encode C into the input HTML document H by inserting spaces between words or sentences of the actual text of H in the following ways, ignoring the original spaces before insertion:

$$\begin{cases} if\ C(i) = \mathrm{M},\ insert\ a\ Big5\ space; \\ if\ C(i) = 0,\ insert\ a\ real\ space; \\ if\ C(i) = 1,\ insert\ a\ pseudo-space\ and\ a\ real\ space. \end{cases}$$

Step 4: Embed as many copies of C sequentially as possible until the data-embedding capacity of H is exhausted.

It is noted that for two consecutive copies, the start mark of the second copy can be regarded as the end mark of the first. A flowchart for the above serial number embedding process is shown in Fig. 3.

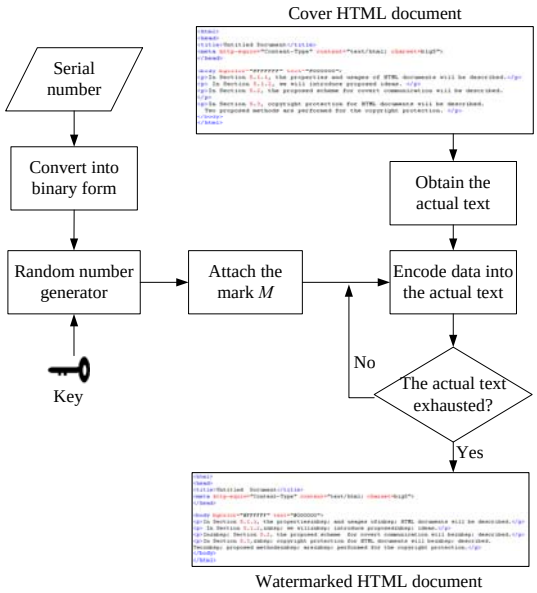


Fig. 3 Flowchart of serial number embedding process.

3.2 Serial Number Extraction

In the serial number extraction process, a person who claims to be the copyright owner of the actual text of an HTML file should provide a key and a serial number for use in the process to prove his/her ownership. Since each copy of the embedded serial number has a start mark and an end mark, we can extract it by checking these marks. And to prevent destroyed copies from affecting the extraction result, a voting scheme is utilized to recover the extracted serial number correctly, utilizing the advantage of embedding multiple copies of the serial number. The steps to extract and so recover the serial number are described as an algorithm as follows. The input to the algorithm is an HTML file or a text file in suspicion (denoted as *H*), which might be a partial or a complete copy of the content of a protected Web page.

Step 1: Extract the embedded data C" according to the number of spaces between words or sentences of the actual text of the input HTML document *H* in the following way:

$$C''(i)=\begin{cases}\text{M}, & \textit{if a Big5 space is extracted};\\ 0, & \textit{if one real space is extracted};\\ 1, & \textit{if two real spaces, or one pseudo}-\textit{space and one real space are extracted.}\end{cases}$$

Step 2: Find in C" two marks of M, including a start one and an end one, to get a complete copy of the copyright data $E'=\{e'_1,e'_2,\ldots\ldots,e'_n\}$ by extracting the stream of data with a fixed length *n* between the two marks of M, where *n* is the bit-length of the serial number used in the embedding process. If the number of extracted bits between the two M's is not equal to *n*, regard this copy to have been destroyed and discard it.

Step 3: After k' complete copies of E' are extracted, regard each E' as a binary-valued vector and perform voting in the following way:

3.1 Get a result specified by

$$V(m)=\sum_{j=1}^{k'}E'_j \quad , \qquad (1)$$

where $1 \le m \le n$ and E'_j is the *j*-th copy of E'.

3.2 Reconstruct the copyright data E' by the following rule:

$$E(m)=\begin{cases}1 & \textit{if } V(m)>\dfrac{k'}{2},\\ 0 & \textit{if } V(m)<\dfrac{k'}{2},\end{cases} \qquad (2)$$

where $1 \le m \le n$.

Step 4: With E' available, recover the original serial number *N* using a reverse disarrangement process corresponding to the disarrangement process mentioned in Step 1 of the serial number embedding process using the same key and the same random number generator.

Fig. 4 shows a flowchart for extracting the serial number from an input HTML document.

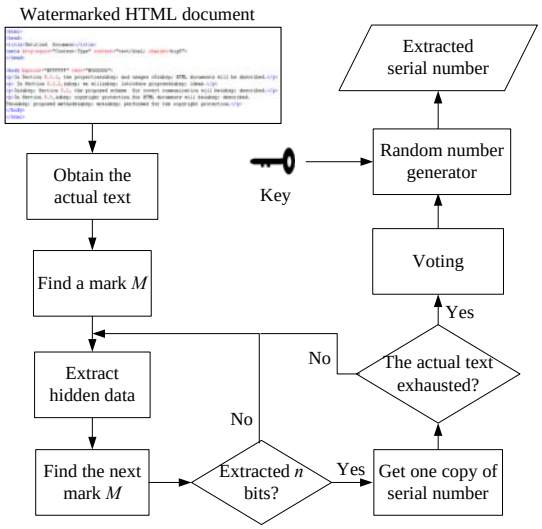


Fig 4. Flowchart of serial number extraction process.

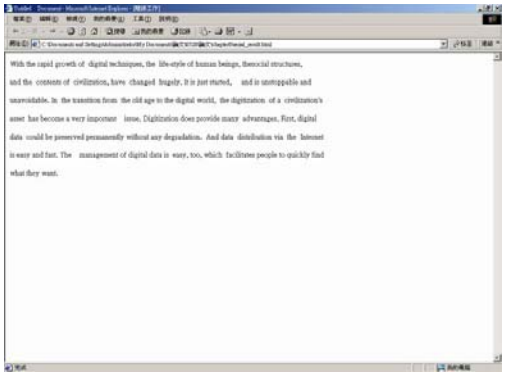
3.3 Experimental Results

An example of experimental results of embedding a serial number as the copyright information is shown in Fig. 5. Fig. 5(a) is an input HTML document, and Fig. 5(b) shows the embedding result. When part of the content of the encoded HTML file as shown in Fig.5(c) is copied, the hidden serial number still can be extracted correctly.

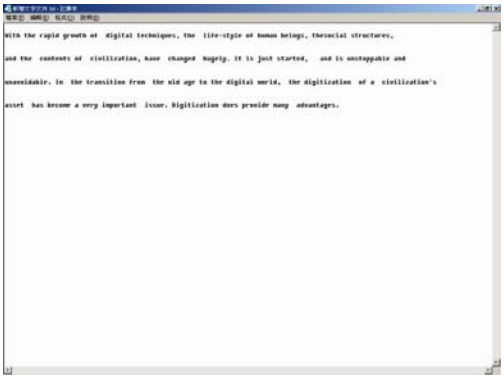
4. Logo Embedding and Extraction

In this section, the proposed method to embed a binary logo image into an HTML document as copyright data by the encoding method described previously is described. Because the size of an image is larger than that of a serial number, embedding a logo image for copyright protection can only be conducted for an HTML document with a large data-embedding capacity. An advantage of the scheme is that a copyright logo is more persuasive than a serial number. If the actual text is modified slightly, some pixels of the embedded logo image may be destroyed, but hopefully the remaining logo data can still be retrieved to reconstruct the embedded logo approximately to prove the copyright of the actual text. The processes of logo image embedding and extraction are described as follows.

(a)



(b)



(c)

Fig. 5 An experimental result. (a) Input HTML document. (b) Encoded HTML document. (c) Copied content.

4.1 Logo Embedding Process

Embedding many copies of a complete logo image into an HTML document is difficult due to the limit of the small data-embedding capacity of the HTML document. For this reason, a *logo sampling* technique is employed to obtain a smaller-sized logo for use in the proposed embedding process. An $n \times n$ binary logo image W is first divided into non-overlapping 2×2 blocks. And the four pixels of each block are assigned into four subclasses, say a, b, c, and d, respectively, as shown in Fig. 6. Then we can get four sampled copies of W, each corresponding to a subclass and with 1/4 resolution of that of W. If one or two sampled copies are lost or destroyed, an approximate W can still be reconstructed by the remaining sampled copies. Besides this logo sampling technique, a voting process will be preformed in the extraction process to recover the embedded sampled copies if the data-embedding capacity of the HTML file is large enough to embed many copies of W.

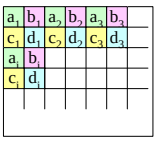


Fig. 6 Sampling a logo image to four subclasses.

Since the hidden data include four reduced-sized sampled copies of W, we must know where one sampled copy is started and ended, and which copy is being extracted. Four different marks are utilized for the four corresponding subclasses to achieve two goals. One goal is to mark the start and the end points of each subclass. If a sampled copy corresponding to one subclass is destroyed, it will be known and discarded without affecting the other extracted sampled copies. The other goal is to determine which subclass of W an extracted sampled copy belongs to, and this can be accomplished by

checking the marks of the extracted sampled copy.

Let W be an $n \times n$ binary logo image, and let the hidden data be denoted as α. The proposed data embedding algorithm can now be described as follows.

Step 1: Use an owner-defined key to disarrange the binary logo image W, resulting in a distinct one denoted by W'.

Step 2: Divide W' by the aforementioned logo sampling technique to get four subclasses, a, b, c, and d:

$$a = \{a_1, a_2, \ldots, a_\ell\},$$
$$b = \{b_1, b_2, \ldots, b_\ell\},$$
$$c = \{c_1, c_2, \ldots, c_\ell\},$$
$$d = \{d_1, d_2, \ldots, d_\ell\}$$

where $\ell = \frac{n}{2} \times \frac{n}{2}$.

Step 3: Attach to each subclass, which is taken as a bit stream, different start marks and an identical end mark, respectively. Use the mark M as the end mark for all subclasses and at most two bits to encode the four start marks in the following way: encode the start mark M_a for subclass a by M, M_b for b by M followed by a "0," M_c by M followed by a "1," and M_d by M followed by the bit pair "00," resulting in the following bit streams for the four subclasses, respectively:

$$A = \{M_a, a_1, a_2, \ldots, a_\ell, \mathrm{M}\},$$
$$B = \{M_b, b_1, b_2, \ldots, b_\ell, \mathrm{M}\},$$
$$C = \{M_c, c_1, c_2, \ldots, c_\ell, \mathrm{M}\},$$
$$D = \{M_d, d_1, d_2, \ldots, d_\ell, \mathrm{M}\}.$$

Step 4: Encode A, B, C, and D between words or sentences of the actual text of the input HTML document H sequentially in the following way:

$$\begin{cases} \text{if } \alpha(i) = \mathrm{M}, \text{ insert a } Big5 \text{ space}; \\ \text{if } \alpha(i) = 0, \text{ insert a real space}; \\ \text{if } \alpha(i) = 1, \text{ insert a } pseudo-\text{space and a real space}. \end{cases}$$

Step 5: Embed as many copies of the four subclasses as possible to exhaust the data-embedding capacity of H. Assume that the final embedding result includes: k_a copies of subclass a, k_b copies of b, k_c copies of c, and k_d copies of d. For every two consecutive subclasses, regard the start mark of the second copy as the end mark of the first, and ignore the end mark of the first.

After the above algorithm is performed, the entire hidden data will be in the following form:,

$$\alpha = \overbrace{M_a, a_1, a_2, \ldots, a_l}^{\text{copy 1 of subclass } a} \overbrace{M_b, b_1, b_2, \ldots, b_l}^{\text{copy 1 of subclass } b} \ldots \overbrace{M_x, x_1, x_2, \ldots, x_l, \mathrm{M}}^{\text{copy } k_x \text{ of subclass } x}$$

where $x = \{a, b, c, d\}$. A flowchart of the watermark embedding process is shown in Fig. 7.

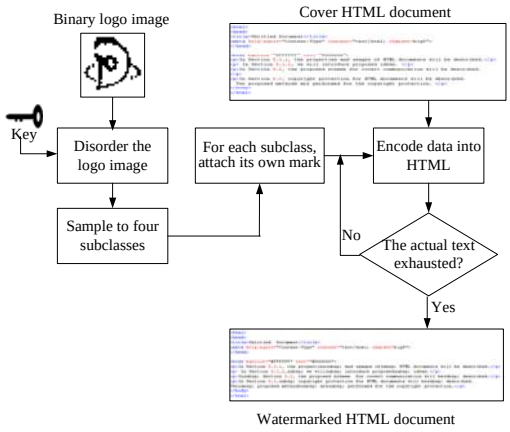


Fig. 7 Flowchart of proposed logo embedding process.

4.2 Logo Extraction Process

In the proposed logo extraction process, a person who claims to be the copyright owner of a Web page should provide the correct key that was used in the embedding process to extract a particular logo image to prove the ownership. Since the hidden logo is divided into four subclasses and each subclass has its own start mark and end mark, the logo can be reconstructed by checking the start and the end marks to determine which subclass has been extracted, and whether the extracted subclass is correct. If one or two subclasses are destroyed, the remaining subclasses can still be used to reconstruct a noisy version of the original logo, and people thereby can still claim his/her ownership of the Web page. If the data-embedding capacity of an HTML file is large enough, more than one copy of each subclass will be hidden. Then, a voting scheme is utilized for each subclass in the extraction process to enhance the correctness of each extracted subclass.

Let the given logo be an $n \times n$ binary image. And let α' be the extracted data from a given input HTML document H. A detailed algorithm of the proposed logo extraction process is described as follows.

Step 1: Extract one bit or a mark of α' at a time according to the number of spaces between two words or two sentences of the actual text of H in the following way:

$$\alpha'(i)=\begin{cases}\text{M}, & \textit{if a Big5 space is found};\\ 0, & \textit{if a real space is found};\\ 1, & \textit{if two real spaces or a pseudo}-\textit{space}\\ & \textit{followed by a real space is found}.\end{cases}$$

Step 2: Find a start mark and an end mark to get one copy of one of the subclasses by extracting the bit stream with fixed length ℓ between the two marks, where ℓ is the bit-length of every subclass in the embedding process. And determine which subclass is extracted by the start mark. If the number of the extracted bits is not equal to ℓ, regard this copy as having been destroyed and discard it. Let the extracted subclasses, denoted as a', b', c', and d', be as follows:

$$a' = \{a_1', a_2', \ldots, a_\ell'\},$$

$$b' = \{b_1', b_2', \ldots, b_\ell'\},$$

$$c' = \{c_1', c_2', \ldots, c_\ell'\},$$

$$d' = \{d_1', d_2', \ldots, d_\ell'\},$$

where $\ell = \frac{n}{2}\times\frac{n}{2}$.

Step 3: Perform a voting process in the following way where k_a, k_b, k_c, and k_d denote the numbers of copies of subclasses a', b', c', and d', respectively:

3.1 compute

$$V_x(m)=\sum_{j=1}^{k_x'} x_j', \tag{3}$$

where $1 \le m \le \ell$, $x=\{a,b,c,d\}$, and x_j' is the j-th copy of x'.

3.2 Recover the four subclasses by the following rule:

$$x(m)=\begin{cases}1, & \textit{if } V_x(m) > \frac{k_x'}{2};\\ 0, & \textit{if } V_x(m) < \frac{k_x'}{2}.\end{cases} \tag{4}$$

Step 4: Reconstruct a disarranged logo image W' by the four extracted subclasses obtained in the last step.

Step 5: Use the key that was selected by the owner in the data embedding process and applying an inverse disarrangement process corresponding to the disarrangement process mentioned in Step 1 of the embedding process to get the original logo image W.

Fig. 8 shows a flowchart for the above algorithm for extracting a logo image from an input HTML document.

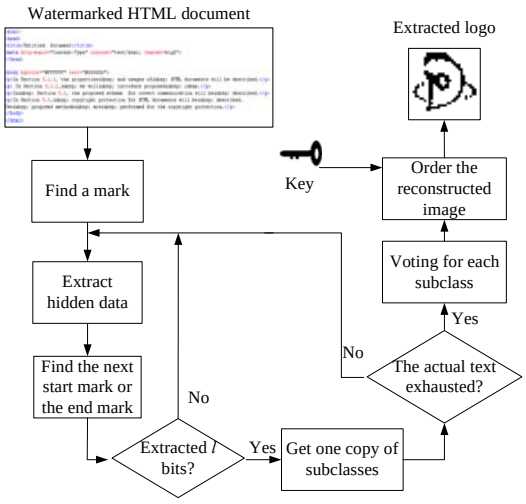


Fig. 8 Flowchart of logo extraction process.

4.3 Experimental Results

An experiment result of embedding a binary logo image as the copyright information is shown in Fig. 9. A copyright logo of size 32×32 is shown in Fig. 9(a). Fig. 9(b) is an input HTML document, and the embedding result is shown in Fig. 9(c). If only part of the content of the encoded HTML file is copied, a noisy version of the hidden logo image can still be extracted. Fig. 9(d) shows a copy of part of the Web page, and Fig. 9(e) is the corresponding extracted copyright logo.

5. Conclusions

In this paper, a data hiding method that can be employed to embed either a serial number or a binary logo image into an HTML document for copyright protection has been proposed. Besides taking an HTML file directly, an illicit user might copy the actual text displayed in Web browser windows for misuses. Thus, copyright data must be visible in the browsers and pseudo-spaces that are visible in browsers are utilized to encode copyright data in this study. And because many copies of the copyright data are embedded by the proposed method, the owner can prove his/her copyright of the HTML document by extracting at least one copy of his/her copyright data with a higher probability. Experimental results show the feasibility of the proposed approach.

References

[1] F. T. Leighton, I. J. Cox, J. Kilian, and T.

Shamoon, “Secure spread spectrum watermarking for multimedia,” *IEEE Trans. on Image Processing*, Vol. 6, No. 12, pp. 1673-1687, 1997.

[2] H. Y. Chang, “Data hiding and watermarking in color images by wavelet transforms,” *Master Thesis*, Department of Computer and Information Science, National Chiao Tung University, Taiwan, Republic of China, 1999.

[3] C. I. Podilichuk and E. J. Delp, “Digital Watermarking: Algorithms and Applications,” *IEEE Signal Processing Magazine*, Vol. 18, No. 4, pp. 33-46, Jul 2001.

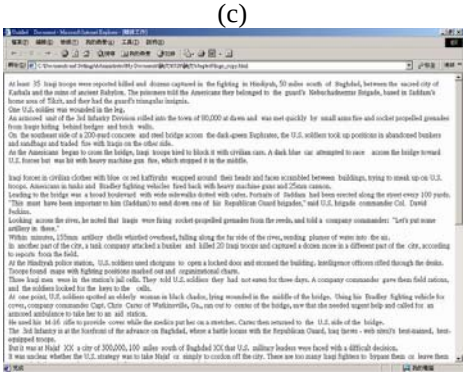
[4] P. Wayner, “Strong Theoretical Steganography,” *Cryptologia*, Vol. XIX/3, pp. 285-299, 1995.

[5] W. Bender, et al., “Techniques for data hiding,” *IBM System Journal*, Vol. 35, No. 3 & 4, Feb. 1996.

(a)

(b)

(c)



(d)

(e)

Fig. 9 An experimental result. (a) Input logo. (b) Input HTML document. (c) Encoded HTML document. (d) Copied content. (e) Extracted copyright logo.