A New Steganographic Method for Data Hiding in Microsoft Word Documents by a Change Tracking Technique

Tsung-Yuan Liu, *Student Member, IEEE*, and Wen-Hsiang Tsai, *Senior Member, IEEE*

***Abstract*—A new steganographic method for data hiding in Microsoft Word documents by a change tracking technique is proposed. The data embedding is disguised such that the stegodocument appears to be the product of a collaborative writing effort. Text segments in the document are degenerated, mimicking to be the work of an author with inferior writing skills, with the secret message embedded in the choices of degenerations. The degenerations are then revised with the changes being tracked, making it appear as if a cautious author is correcting the mistakes. The change tracking information contained in the stegodocument allows the original cover, the degenerated document, and, hence, the secret message to be recovered. The extra change tracking information added during message embedding is vital in a normal collaboration scenario, and so hinders ignorant removals by skeptics. Experiments demonstrate the feasibility of the proposed method.**

***Index Terms*—Stegodocument.**

I. Introduction

STEGANOGRAPHY is a study of techniques that embed secret information imperceptibly into a cover medium for the purpose of security protection or covert communication. Most of the research concentrated on images, audios, and videos as cover media [1]–[5]. Imperceptibility of data hiding is commonly achieved by exploiting the weaknesses of the human auditory and visual systems, using the techniques of, for example, changing the least-significant bits of the pixels of a cover image to embed information [6], or shifting lines, words, or characters by a small amount in an image containing text [7]. Other works hide information by adding redundant data, or making use of alternative representations of electronic data. For example, hidden information can be added in a text document by adding tabs and spaces at the end of the lines. Also, the different combinations of the color palette entries in a GIF image [8] can be used to embed secret data into the image file.

In this paper, a new steganographic method is proposed in which data embedding is disguised to be the product of a collaborative document authoring effort. That is, the stegodocument is made to appear to be the work of multiple authors. To facilitate communication of the authors during the collaborative document authoring process, the word processor records the exact modifications by an author and embeds the ways of revision as change tracking information into the document. From such change tracking information, we can discern the exact changes made by a prior author, and can recover a prior version of the document if necessary. Fig. 1 shows an example of the collaborative document authoring process in Microsoft Word, where an author is modifying a document and the word processor has tracked the author's modifications. The modifications by the author are clearly marked, with the deleted words stroked-through and newly inserted text underlined. Formatting changes are displayed as comment bubbles at the right-side margin of the page. Each collaborating author can accept or reject individual or all modifications made by another author. It is a common practice for a collaborating author to review and then accept or reject each modification in a document first before performing his or her own corrections.

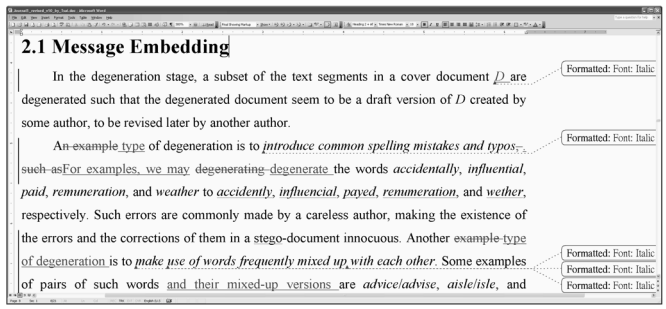

Fig. 1 Screenshot of Microsoft Word in a case of collaborative document authoring.

The basic idea of the proposed method is to degenerate the contents of a cover document D to arrive at another document D' by embedding a secret message M in D during the transformation process, as shown in Fig. 2. The degeneration introduces errors into the degenerated document D' such that the degenerated document appears to be a preliminary work by a virtual author A', which is to be revised later by another author A. A stegodocument S is then produced from D' by revising D' back to D with the changes being tracked, making it appear as if author A is correcting the errors in D'. On the other hand, by making use of the change tracking information in the stegodocument S, a recipient B of S can easily recover the original document D as well as the degenerated document D' from both of which the

Manuscript received June 16, 2006; revised November 27, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jessica J. Fridrich.

T.-Y. Liu is with the Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: gis91811@cis.nctu.edu.tw).

W.-H. Tsai is with the Department of Computer Science and Information Engineering, Asia University, Taichung 413, Taiwan, R.O.C. (e-mail: whtsai@asia.edu.tw).

Digital Object Identifier 10.1109/TIFS.2006.890310

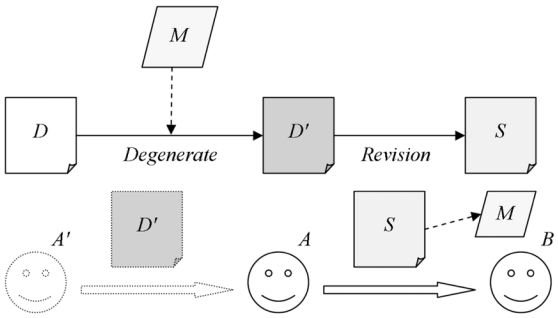

Fig. 2 Author A sends a stegodocument S with an embedded message M to a recipient B after embedding M into a cover document D to form S that appears to be the collaborative product of multiple authors A and A'.

embedded information can be extracted. We have chosen Microsoft Word documents as cover media, which provide change tracking facilities to materialize the proposed method. Communications via Word documents are commonplace for personal, business, or academic purposes these days, so transmissions of such documents will not be under close scrutiny. We note that any other document format that offers change-tracking facilities can also be used. For example, the OASIS open document format, which has become increasingly popular, can also be used for data hiding using the proposed method.

To the best of our knowledge, this paper is the first published work both on steganography in Microsoft Word documents and also on disguising for the steganographic purpose using collaborative writing. In the remainder of this paper, related works and merits of our approach over them are discussed in Section II. The proposed method is described in detail in Section III, followed by the description of a prototype implementation and the experimental results in Section IV to demonstrate the feasibility of the proposed method. In Section V, we discuss security considerations and, finally, in Section VI, we conclude with some suggestions for future works.

II. Related Works

Most of the works cited in the introduction use the technique of modifying a cover medium to embed information. This type of data hiding generally assumes that the cover medium used is unknown to an adversary, or otherwise, the discrepancies between the cover medium and the corresponding stegomedium will arouse suspicion. On the other hand, the proposed method provides legitimate cases in using a known cover document. For example, an already published document that is collaboratively authored can be used as a cover document. The stegodocument S appears to be the version of the paper before change tracking information removal and submission for publication. The transmission of S by one of the collaborating authors to another author, a colleague, or a supervised student of the author is reasonable. A colleague or a student receiving the document containing the change tracking information can learn of the mistakes made by a colleague and the appropriate corrections to be made thereof.

Linguistic steganography methods that generate the cover text directly, such as the forecast generator [9] and the SpamMimic spam generator [10], do not have the known cover problem. However, the text produced using these methods is often implausible to a human reader.

Linguistic steganography methods that manipulate a cover text using substitutions or syntactic transformations [11] produce more innocuous text, but are still often detectable by a human reader due to the inherent difficulties in natural-language processing and understanding. Many of the prior techniques, however, can be used to a greater effect in the proposed method in degenerating a cover document D to another D'. The inconsistencies and errors in D' are more tolerable in this setting because D' is disguised to be the draft work of an inferior author.

The proposed method addresses the issue of producing plausible text by disguising the degenerated text as the draft work of an inferior author. Translation-based steganography [12], on the other hand, uses the expected errors in the translation process, especially in machine translation, to solve the issue of producing implausible text. That is, information is hidden in the noise that occurs in language translation. In cases where sending imperfect translations to a colleague are reasonable, the stegodocuments resulting from translation-based steganography are inconspicuous. Furthermore, an improved technique proposed in [13] avoids the transmission of the original text and, hence, does not have the known cover problem. The translation-based approach, however, may be vulnerable to active attacks. In fact, most of the proposed steganography techniques that embed secret information in the subliminal channel of a cover medium are vulnerable in the presence of an active warden. In the active warden attack model, it is assumed that the adversary, Wendy, is allowed to introduce subtle modifications to passing stegoobjects between Alice and Bob, as long as the modifications do not interfere with normal communications between them. Specifically, if Alice sends to Bob a stegotext produced using the translation-based approach, Wendy can choose to lend a "helping" hand by correcting obvious errors in a translation and, thus, obliterate the hidden message. Similarly, if information is hidden in HTML files by adding useless spaces and line breaks [14] or by changing the case of letters in the tags [15], Wendy can simply remove all redundant spaces or alter all letters in the tags to be lowercase in a passing HTML file because she is certain such actions will not affect normal communications, while any information that could have been hidden in the file will be removed. In contrast, the proposed method embeds secret data in the change tracking information, with the degenerated text and the corresponding revised text intended for the recipient to see as is, and will thus not be tampered with ignorantly.

III. Data Hiding by Change Tracking Techniques

In the proposed steganographic method, a binary secret message M is embedded inside a cover document D to obtain a stegodocument S. The embedding process is divided into two stages, the degeneration stage, and the revision stage, as shown in Fig. 2. The cover document D is partitioned into text segments $d_1, d_2, \ldots$, and d_n, and each segment d_i is either kept unchanged or degenerated into a new version during the degeneration stage to arrive at a degenerated document D' containing degenerated text segments $d'_1, d'_2, \ldots, d'_n$. The secret message is embedded during the degeneration process. In the revision stage, each previously degenerated text segment d'_i is revised

back to d_i with the revisions being tracked by using the "Track Changes" feature of Word, resulting in a final stegodocument S consisting of revised text segments $s_1, s_2, \ldots$ and s_n. Here, each s_i includes its original text segment d_i and the associated change tracking information. Message extraction from the stegodocument S is basically an inverse process of the message embedding process. For the remainder of this section, the details of message embedding and extraction will be described.

A. Message Embedding

In the degeneration stage, a subset of the text segments in a cover document D are degenerated such that the degenerated document seems to be a draft version of D created by some author, to be revised later by another author.

A type of degeneration is to introduce common spelling mistakes and typos. For example, we may degenerate the words accidentally, influential, paid, remuneration, and weather to accidently, influencial, payed, renumeration, and wether, respectively. Such errors are commonly made by a careless author, making the existence of the errors and the corrections of them in a stegodocument innocuous. Another type of degeneration is to make use of words frequently confused with each other. Some examples of pairs of such words and their mixed-up versions are advice/advise, aisle/isle, and complement/compliment. Specifically, we can degenerate, for example, a text segment $d =$ "advice" in the cover document into $d' =$ "advise" by using the pair of words advice/advise.

There are also many other types of degenerations, such as synonym replacements [16], syntactic transformations such as passivization and clefting [17], and techniques from linguistic steganography studies. As mentioned previously, the use of existing degeneration techniques in the proposed framework makes the resulting imperfect text less conspicuous.

In general, we define a degeneration set R_d to be the ordered set of possible degenerated text segments for a text segment d. We use the notation $R_d(j)$ to denote the jth element in R_d, and the notation $\Pr\{R_d(j)\}$ to denote the probability of occurrence for $R_d(j)$. The probabilities of occurrences are used during message embedding so that the system prefers substitutions that occur commonly and, thus, produces a more natural stegodocument. It is noted that $\sum_{j=1}^{c} \Pr\{R_d(j)\} = 1$, where $c = |R_d|$, the size of R_d.

During the proposed message embedding process, a subset of the text segments in the cover document is degenerated. The indices of the chosen text segments are called the embedding places, denoted as a set by $P = \{i_1, i_2, \ldots, i_G\}$, where G is the number of text segments degenerated and $1 \leq i_k \leq n$ for $1 \leq k \leq G$. With the previously defined notations, the text segments $d_i, i \in P$, are individually degenerated to be $R_{d_i}(j_i), 1 \leq j_i \leq |R_{d_i}|$, with the degeneration indices j_i dependent on the message M being embedded.

The message is treated as an m-bit bitstream $M = b_1 b_2 \cdots b_m$, where each b_i is a bit. A message length header H is added in front of M and strings of random 0's and 1's are padded to the end of M as being necessary during the message embedding. That is, we embed into the cover document the bitstream $M' = l_1 l_2 \cdots l_L b_1 b_2 \cdots b_m x_1 x_2 \cdots$, with $H = [l_1 l_2 \ldots l_L]$ being the message length header with the value m, and x_i being the padding bits that are selected randomly. The communicating parties should agree on the magnitude of L beforehand, such that it can accommodate the longest message that is to be communicated between the parties. The message is usually encrypted first before message embedding for enhanced security.

The message bits are embedded using Huffman coding at each embedding place in a way that is similar to that proposed by Wayner [18]. The previously mentioned probabilities of occurrences of $R_d(j)$ are used to assign variable-length Huffman codes to different degenerations. Shorter Huffman codes are assigned to degenerations with higher probabilities of occurrences and longer ones to those with lower probabilities of occurrences. A degeneration with a shorter Huffman code is more likely to match the message bits being embedded and, hence, more possibly to be selected as the choice of degeneration. The details of the message embedding process are presented in the algorithm below.

Algorithm 1: Message Embedding by Text Degeneration and Revision

Input: a cover document D partitioned into text segments $d_1, d_2, \ldots, d_n$; a message to be embedded $M' = l_1 l_2 \cdots l_L b_1 b_2 \cdots b_m x_1 x_2 \cdots = b'_1 b'_2 b'_3 \cdots$; and a secret key K.

Output: a stegodocument $S = \{s_1, s_2, \ldots, s_n\}$.

Steps:

1) Initialize the set P of embedding places to be empty.
2) Define an index p pointing to the position of the message bit b'_p which we are currently encoding, with initial $p = 1$.
3) Select an embedding place i randomly using K such that i is in the range of $1 \leq i \leq n$ and not in the set P; and then add i to P.
4) Construct as follows a Huffman tree T for the text segment d_i with degeneration set R_{d_i} of size c.
   a) Create leaf nodes $n_1, n_2, \ldots, n_c$, and assign a weight of $w_k = \Pr\{R_{d_i}(k)\}$ to node n_k for all $1 \leq k \leq c$.
   b) Initialize a set Q to contain all of the leaf nodes $n_1, n_2, \ldots, n_c$.
   c) Find in Q the node n' with the minimum weight w' and the node n'' with the second smallest weight w''; and then remove n' and n'' from Q.
   d) Create a new node η with weight $w' + w''$, and assign n' as its left child and n'' as its right child.
   e) If Q is empty, then tree T has been constructed and take η as its root; else, add node η to Q and go to Step 4c).
5) Degenerate text segment d_i to be $d'_i = R_{d_i}(j)$, where the degeneration choice j is determined as follows.
   a) Starting from the root of tree T, traverse T to the left child if b'_p is 0 or to the right child if b'_p is 1.
   b) Increment p and continue node traversal in a similar way until a leaf node n_j is reached.

c) Take the index j of n_j as the desired degeneration choice.

6) Repeat Steps 3) through 5) until the entire message has been embedded, that is, until $p > L + m$.
7) Revise each previously degenerated text segment d'_i back to d_i with the revisions made being tracked to yield stegotext segments s_i for all i in P.

We note that the proposed embedding procedure is generic with no restriction imposed on the elements in a degeneration set R_d. In our experiments, we used a common English errors database, a synonym database, and a collection of real-world collaboration editing entries to construct the degeneration database.

As an example, suppose that the degeneration set R_d for $d =$ "An illustrative example is shown," contains the entries "An illustrative example is shown." "The illustrating example is shown," and "An illustrating example is shown." The text d in the document D is degenerated to one of the three choices depending on the message being embedded. If the occurrence probabilities of the three entries are $4/6, 1/6$, and $1/6$, respectively, then Steps 4) and 5) of Algorithm 1 will effectively assign Huffman codes of 1, 00, and 01, respectively to the entries. In the case that the message to be embedded is 00, d will be degenerated as $d' =$ "The illustrating example is shown." Finally, d' is revised back to d with the changes being tracked to yield the stegotext $s =$ "~~The illustrating~~An illustrative example is shown."

B. Message Extraction

The change tracking information included in the stegodocument S allows simple recovery of the original document D and the degenerated document D', from both of which the embedded message can be extracted. The proposed message extraction method is described in the following algorithm.

Algorithm 2: Message Extraction

Input: a stegodocument $S = \{s_1, s_2, \ldots, s_n\}$ and a secret key K.

Output: the extracted message $M' = b'_1 b'_2 b'_3 \cdots = l_1 l_2 \cdots l_L b_1 b_2 \cdots b_m x_1 x_2 \cdots$.

Steps:

1) Recover the original documents $D = \{d_1, d_2, \ldots, d_n\}$ and the degenerated document $D' = \{d'_1, d'_2, \ldots, d'_n\}$ from S using the change tracking information and the related operations provided by Word.
2) Initialize the set P of embedding places to be empty.
3) Define an index p pointing to the position of the message bit b'_p which we are currently decoding, with initial $p = 1$.
4) Select the same embedding place i as that in message embedding using K and P.
5) Construct a Huffman tree T with leaf nodes $n_1, n_2, \ldots, n_c$ for the text segment d_i with a degeneration set R_{d_i} of size c using the same steps described in Algorithm 1.

TABLE I
OCCURRENCE PROBABILITIES AND HUFFMAN CODES FOR ENTRIES IN A DEGENERATION SET OF TRAVEL

j	$R_d(j)$	$\Pr\{R_d(j)\}$	Huffman code
1	go	0.6306889	0
2	travel	0.2118223	10
3	trip	0.0536723	1100
4	journey	0.0273936	11010
5	jaunt	0.0004189	110110
6	locomote	0.0000347	110111
7	move	0.0759693	111

6) Determine the choice of degeneration j such that $R_{d_i}(j) = d'_i$.
7) Decode the message bits encoded in j in the following way:
   a) Starting from the root of the tree T, traverse it to the leaf node n_j and note the path traversed.
   b) Analyze the path traversed, and set b'_p to be 0 if the path goes down a left child; or to be 1 if the path goes down the right. Increment the value of p for each child traversed.
8) Repeat Steps 4) through 7) until the entire message has been extracted, that is, until $p > L + m$.

As an example, given a revised text segment $s =$ "~~move~~travel," we can recover the original and the degenerated text segments to be $d =$ "travel" and $d' =$ "move," respectively. Suppose that the degeneration set R_d contains the seven entries go, travel, trip, journey, jaunt, locomote, and move, with the respective probabilities of occurrences as shown in Table I. The respective Huffman codes for the entries as constructed in Algorithm 2 are also shown. Since the degenerated text segment is "move," we can conclude that the bits "111" were previously embedded.

IV. EXPERIMENTAL RESULTS

The proposed methods were implemented using Microsoft C#.NET and Microsoft Office 2003, and the automation technique provided by Microsoft [19] was used to manipulate Word documents. The degeneration sets were constructed by using public linguistic databases as well as real past collaboration editing files kept by the second author.

The first degeneration database was constructed by using entries from the list of common errors in English compiled by Brian [20]. We filtered out entries that are automatically corrected by Microsoft Word by its AutoCorrect feature (such as parallelled, therefor, etc.), as these are unsuitable candidates for degeneration. Table II summarizes the resulting database used, which contains 760 degeneration sets in total. We have included the text d itself in the set R_d, such that if it is chosen as the degeneration choice, the text is effectively not degenerated.

We have also utilized the WordNet 2.1 lexical database [21] to obtain sets of synonyms for the degeneration purpose. The occurrence probabilities of the entries in R_d for both the common English errors and the synonyms are estimated by making use of the Google SOAP Search API [22]. Specifically, we query the Google web index to get the approximate

TABLE II
SUMMARY OF COMMON ENGLISH ERRORS DATABASE USED AS R_d

$\lvert R_d \rvert$	Examples of d and d'	Entries
2	breach (breech), canon (cannon), coarse (course), complement (compliment), criteria (criterion), currant (current), deformation (defamation), desert (dessert), device (devise), enquire (inquire), envelop (envelope), farther (further), feint (faint), freshman (freshmen), influencial (influential), loose (lose), marital (martial), medium (median), moral (morale), offense (offence), ordnance (ordinance), payed (paid), persecute (prosecute), pray (prey), quiet (quite), remuneration (renumeration), setup (set up), shear (sheer), $\cdots$	675
3	allusive (elusive, illusive), palette (palate, pallet), imminent (eminent, immanent), medal (metal, meddle), possessed of (possessed by, possessed with), weather (wether, whether) $\cdots$	71
4	morale (ethics, morals, moral), carat (caret, carrot, karat), epigram (epigraph, epitaph, epithet) $\cdots$	14

TABLE III
OCCURRENCE PROBABILITIES AND HUFFMAN CODES OF STUDY AND ITS SYNONYMS

word	pages on Google	occurrence probability	Huffman code
report	1560000000	0.1733984	000
subject	1180000000	0.1311603	001
work	2630000000	0.2923319	01
field	1850000000	0.2056327	10
study	758000000	0.0842538	1100
discipline	108000000	0.0120045	11010
sketch	71500000	0.0079474	110110
bailiwick	775000	0.0000861	1101110
cogitation	348000	0.0000387	1101111

number of web pages that contain an entry in R_d. After the occurrences of each entry in R_d are determined, the occurrence probabilities for the entries can be calculated. As an example, for $d =$ "study," we use WordNet to find its synonyms "report," "subject," "work," "field," "discipline," "sketch," "bailiwick," and "Cogitation." Table III shows the occurrence probabilities and the resulting Huffman codes for these words. We have also collected files of the second author's editing of his students' works, including 157 thesis chapters and 20 journal and conference papers in recent years. The collaboration editing entries were collected by first identifying sentences in the documents that contain change tracking information. The text before and that after editing were then extracted, and identical starting and ending phrases in the two text segments are trimmed away so that the entry can be more widely applicable. Finally, we group the entries according to the text after editing. The database constructed using these real-life collaboration entries is then used during the degeneration stage. The degenerations made by using this database are realistic and the resulting stegodocuments are indistinguishable from a real collaborative document. For simplicity, we name the common errors in English database, the synonym database, and the real collaboration entries database as databases 1, 2, and 3, respectively.

Fig. 3 shows some extracts of the stegodocuments produced using the proposed method. The stegodocuments shown in the left column are produced by using a combination of databases 1 and 3, while those on the right are produced by using a combination of databases 1 and 2. We observe that the ways of degenerations are distinctively different when using the different databases.

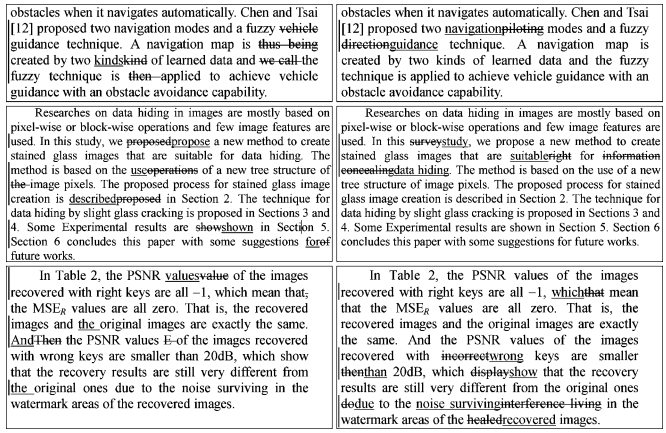

Fig. 3 Extracts of stegodocuments produced using Algorithm 1. Stegodocuments on the left were produced by using databases 1 and 3; while those on the right were produced by using databases 1 and 2.

We conducted a set of experiments to quantitatively measure the message embedding capacity of the proposed method when using the above databases. We have selected five theses and seven journal and conference papers as the test documents. The numbers of words and sentences of the documents are listed in Table IV. The maximum message length embeddable for a document is message dependent due to the Huffman coding used in the message embedding. In this set of experiments, we embedded random 0's and 1's into all embeddable places in the test document and repeated the embedding ten times for each test document. The average bits embeddable for each document and the average file size of each document after embedding are shown in Table IV. We also measured the embedding bit rate, that is, the number of bits embedded for each bit of the document. On average, 0.33 b can be embedded into each word and 5.42 b can be embedded into each sentence when using a combination of databases 1 and 2, and 0.07 b per word and 1.19 b per sentence can be embedded when using a combination of databases 1 and 3.

The embedding capacity is higher when using databases 1 and 2 because the WordNet database used produced many synonyms for a text segment. The embedding capacity of the proposed method by using the synonym database is lower than other works that hide by using direct synonym replacements. For example, [23] achieves an embedding bit rate of about 250 to 1. This is mainly due to the overhead in storing Microsoft Word documents. To increase the message embedding capacity, we

TABLE IV
EXPERIMENTAL RESULTS OF MESSAGE EMBEDDING CAPACITY

Document	Sentences	Words	Databases 1 & 2			Databases 1 & 3		
			Capacity (bits)	File size (bytes)	Bit rate	Capacity (bits)	File size (bytes)	Bit rate
Paper 1	240	4,075	1,558	176,742	1/907	261	154,982	1/4745
Paper 2	351	4,903	1,768	213,453	1/966	374	196,096	1/4197
Paper 3	287	3,518	1,163	131,482	1/904	207	116,941	1/4513
Paper 4	223	4,341	1,421	126,566	1/712	321	113,203	1/2825
Paper 5	222	3,857	1,241	172,493	1/1112	301	157,798	1/4194
Paper 6	350	5,545	2,141	170,138	1/636	470	151,347	1/2577
Paper 7	338	8,978	2,178	214,323	1/787	451	182,989	1/3247
Thesis 1	1,204	15,414	4,121	482,611	1/937	1,605	549,837	1/2741
Thesis 2	1,415	20,547	7,282	607,232	1/667	1,620	521,984	1/2577
Thesis 3	1,213	18,918	6,585	574,362	1/698	1,750	684,698	1/3130
Thesis 4	1,456	21,903	7,276	739,994	1/814	1,477	437,197	1/2368
Thesis 5	1,150	20,254	6,773	494,182	1/584	1,145	459,264	1/3208
Total	**8,449**	**132,253**	**43,509**	**4,103,578**	**1/754**	**9,983**	**3,726,336**	**1/2986**

can use a larger degeneration database that contains more degenerations. Alternatively, we can simply compress the stegodocument because Word documents can sometimes be compressed effectively and it is normal to send zipped documents.

V. SECURITY CONSIDERATIONS

In the proposed method, the degeneration sets and the key used are agreed upon by the sending and receiving parties beforehand. The degenerations in the degeneration database should model realistic errors to counter visual steganalysis. Our way of using existing collaboration data as conducted in our experiments described previously can achieve this purpose. Specifically, an adversary inspecting a stegodocument yielded by our experiments could not tell whether it is really an actual author making the mistakes, or whether the mistakes are introduced for the steganographic purpose by using the proposed method.

Next, it is noted that the proposed approach is robust against statistical steganalysis [24] because the degenerations are chosen according to their occurrence probabilities, resulting in the occurrence frequencies of degenerations in a stegodocument being in line with those of normal documents. To ensure the statistical properties of the degenerations of a stegodocument to be as close to that of a normal document, we can encrypt the message first before embedding so that the bits of the encrypted message are randomly distributed. Nevertheless, there is still a chance for the statistics of the degenerations to stray away from that of a normal document. To ensure maximal statistical coherence, we can alter the occurrence probabilities of the degenerations appropriately during message embedding. For example, we can halve the occurrence probability of a degeneration after it has been chosen so that it is less likely to be chosen later again, thus achieving the desired statistical coherence with a normal document.

In addition, to ensure that the scheme is as inconspicuous as possible to adversaries, the degeneration database used should only be known by the communicating parties. This can be achieved, for example, by using an evolving degeneration database, that is, we modify the degeneration database dynamically when normal collaboration documents are transmitted between the parties. One way to implement this idea is for the communicating parties to add into the degeneration database a key-dependent subset of the degenerations derived from collaboratively working on a normal document. In this way, an adversary cannot determine the exact contents of the degeneration database being used. Alternatively, a sender, after having embedded information in a stegodocument using the proposed method, may manipulate the unused portions of the stegodocument to include degenerations outside their agreed degeneration database to mislead adversaries. The extra degenerations so introduced are assumed to be ignored by the receiver.

VI. CONCLUSION

A new steganographic method for data hiding in Microsoft Word documents has been presented in this paper. The data embedding is disguised such that the stegodocument appears to be the product of a collaborative writing effort. Information is embedded in the degeneration stage of document transformation with steganographic effects. The degeneration stage introduces different degenerations mimicking an author with inferior writing skills, with the secret message embedded in the choices of degenerations using Huffman coding. The proposed message embedding and extraction methods have been implemented, proving the feasibility of the proposed method.

The proposed change tracking technique for the steganography purpose is special in that the modifications made during embedding are essential information that is not to be tampered with ignorantly. On the contrary, methods that are based on redundant or unused information, imperceptibility, or alternative representations, are vulnerable against an active warden, who can process all suspects without affecting normal cases of usages. The degeneration database should contain realistic and evolving degenerations, such that the warden cannot distinguish between legitimate collaboration cases and covert communication cases.

Future work can investigate using arithmetic coding instead of Huffman coding in data embedding [25] for improved embedding capacity. Other data-hiding methodologies based on

disguising under collaborative efforts are open future research topics. In particular, data hiding in source code versioning using the CVS is an interesting future work.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for many useful comments and for the suggestion to use Huffman coding during message embedding.

REFERENCES

[1] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Syst. J.*, vol. 35, no. 3–4, pp. 313–336, 1996.

[2] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[3] M. Wu, H. Yu, and A. Gelman, "Multi-level data hiding for digital image and video," presented at the SPIE Photonics East, Boston, MA, 1999.

[4] N. F. Johnson, Z. Duric, and S. Jajodia, *Information Hiding: Steganography and Watermarking—Attacks and Countermeasures*. Norwell, MA: Kluwer, 2001.

[5] R. Chandramouli, M. Kharrazi, and N. Memon, "Image steganography and steganalysis concepts and practice," *Digital Watermarking Lecture Notes in Computer Science 2939*, pp. 35–49, 2004.

[6] D. C. Wu and W. H. Tsai, "A steganographic method for images by pixel-value differencing," *Pattern Recognit. Lett.*, vol. 24, no. 9–10, pp. 1613–1626, 2003.

[7] J. T. Brassil and N. F. Maxemchuk, "Copyright protection for the electronic distribution of text Documents," *Proc. IEEE*, vol. 87, no. 7, pp. 1181–1196, Jul. 1999.

[8] M. Kwan, GIF Colourmap Steganography 2003, online at: [Online]. Available: http://www.darkside.com.au/gifshuffle/.

[9] L. Bourbeau, D. Carcagno, E. Goldberg, R. Kittredge, and A. Polguère, "Bilingual generation of weather forecasts in an operations environment," in *Proc. 13th Int. Conf. Computational Linguistics*, Helsinki, Finland, 1990, pp. 318–320.

[10] *Spam Mimic*, [Online]. Available: http://www.spammimic.com.

[11] K. Bennett, "Linguistic steganography: Survey, analysis, and robustness concerns for hiding information in text," Purdue Univ., West Lafayette, IN, CERIAS Tech. Rep. 2004–13, May 2004.

[12] C. Grothoff, K. Grothoff, L. Alkhutova, R. Stutsman, and M. Atallah, "Translation-based steganography," in *Proc. Information Hiding Workshop*, 2005, pp. 213–233.

[13] R. Stutsman, C. Grothoff, M. Attallah, and K. Grothoff, "Lost in just the translation," in *Proc. ACM Symp. Applied Computing*, 2006, pp. 338–345.

[14] F. Johnson and S. Jajodia, "Steganalysis: The Investigation of Hidden Information," in *Proc. IEEE Information Technology Conf.*, Syracuse, NY, Sep. 1998, pp. 113–116.

[15] G. Sui and H. Luo, "A new steganography method based on hypertext," in *Proc. Radio Science Conf.*, Aug. 2004, pp. 181–184.

[16] M. Chapman, I. D. George, and R. Marc, "A practical and effective approach to large-scale automated linguistic steganography," in *Proc. Information Security Conf.*, Malaga, Spain, Oct. 2001, pp. 156–165.

[17] M. Topkara, C. Taskiran, and E. J. Delp, "Natural language text watermarking," presented at the SPIE Int. Conf. Security, Steganography, and Watermarking of Multimedia Contents VI, San Jose, CA, Jan. 2005.

[18] P. Wayner, "Mimic functions," *Crypt.*, vol. XVI, no. 3, pp. 193–214, 1992.

[19] *Microsoft Office 97 Visual Basic Programmer's Guide*, Microsoft Professional Editions Series, 1997.

[20] P. Brian, Common Errors in English. [Online]. Available: http://www.wsu.edu/brians/errors/.

[21] WordNet v2.1, a lexical database for the English language. Princeton Univ., Princeton, NJ, 2005.

[22] *Google, Google SOAP Search API (beta)*, [Online]. Available: http://code.google.com/apis/soapsearch/.

[23] I. A. Bolshakov, "A method of linguistic steganography based on collocationally-verified synonymy," in *Proc. 6th Information Hiding Workshop*, Toronto, ON, Canada, May 2004, pp. 180–191.

[24] H. Wang and S. Wang, "Cyber warfare: Steganography vs. steganalysis," *Commun. ACM*, vol. 47, no. 10, pp. 76–82, 2004.

[25] P. Wayner, *Disappearing Cryptography: Information Hiding: Steganography and Watermarking*, 2nd ed. San Mateo, CA: Morgan-Kaufmann, 2002, pp. 81–128.

Tsung-Yuan Liu (S'04) received the B.S. degree in electrical engineering from the University of the Witwatersrand, Johannesburg, South Africa, the M.B.A. degree from National Taiwan University, Taipei, Taiwan, R.O.C., and is currently pursuing the Ph.D. degree at the College of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

His research interests include information hiding, image processing, web search, data mining, and artificial intelligence.

Wen-Hsiang Tsai (SM'91) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, R.O.C., the M.S. degree from Brown University, Providence, RI, and the Ph.D. degree from Purdue University, West Lafayette, IN.

Currently, he is the President of Asia University, Taichung, Taiwan, and a Chair Professor with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. His research interests include image processing, computer vision, virtual reality, and information copyright, and security protection.

Dr. Tsai is currently the Computer Society Chair of the IEEE Taipei Section.