Abstract This paper proposes a complete practical methodology for minimizing additive distortion in steganography with general (non binary) embedding operation. Let every possible value of every stego element be assigned a scalar expressing the distortion of an embedding change done by replacing the cover element by this value. The total distortion is assumed to be a sum of per-element distortions. Both the payload-limited sender (minimizing the total distortion while embedding a fixed payload) and the distortion-limited sender (maximizing the payload while introducing a fixed total distortion) are considered. Without any loss of performance, the nonbinary case is decomposed into several binary cases by replacing individual bits in cover elements. The binary case is approached using a novel syndrome-coding scheme based on dual convolutional codes equipped with the Viterbi algorithm. This fast and very versatile solution achieves state-of-the-art results in steganographic applications while having linear time and space complexity w.r.t. the number of cover elements. We report extensive experimental results for a large set of relative payloads and for different distortion profiles, including the wet paper channel. Practical merit of this approach is validated by constructing and testing adaptive embedding schemes for digital images in raster and transform domains. Most current coding schemes used in steganography (matrix embedding, wet paper codes, etc.) and many new ones can be implemented using this framework.

Keywords: STEGANOGRAPHY, STEGO, MATRIX EMBEDDING, WET PAPER CODES, JAVA, MATLAB, VITERBI ALGORITHM.

1. INTRODUCTION

There exist two mainstream approaches to steganography in empirical covers, such as digital media objects: steganography designed to preserve a chosen cover model and steganography minimizing a heuristically-defined embedding distortion. The strong argument for the former strategy is that provable undetectability can be achieved w.r.t. a specific model. The disadvantage is that an adversary can usually rather easily identify statistical quantities that go beyond the chosen model that allow reliable detection of embedding changes. The latter strategy is more pragmatic—it abandons modeling the cover source and instead tells the steganographer to embed payload while minimizing a distortion function. In doing so, it gives up any ambitions for perfect security. Although this may seem as a costly sacrifice, it is not, as empirical covers have been argued to be incognizable [1], which prevents model-preserving approaches from being perfectly secure as well. While we admit that the relationship between distortion and steganographic security is far from clear, embedding while minimizing a distortion function is an easier problem than embedding with a steganographic constraint (preserving the distribution of covers). It is also more flexible, allowing the results obtained from experiments with blind steganalyzers to drive the design of the distortion function.

In fact, today’s least detectable steganographic schemes for digital images [2]–[5] were designed using this principle. Moreover, when the distortion is defined as a norm between feature vectors extracted from cover and stego objects, minimizing distortion becomes tightly connected with model preservation insofar the features can be considered as a low-dimensional model of covers. This line of reasoning already appeared in [5] and [6] and was further developed in [7]. With the exception of [7], steganographers work with additive distortion functions obtained as a sum of single-letter distortions. A well-known example is matrix embedding where the sender minimizes the total number of embedding changes. Near-optimal coding schemes for this problem appeared in [8] and [9], together with other clever constructions and extensions [10]–[15]. When the single-letter distortions vary across the cover elements, reflecting thus different costs of individual embedding changes, current coding methods are highly suboptimal [2], [4]. This paper provides a general methodology for embedding while minimizing an arbitrary additive distortion function with a performance near the theoretical bound. We present a complete methodology for solving both the payload-limited and the distortion-limited sender. The implementation described in this paper uses standard signal processing tools—convolutional codes with a trellis quantizer—and adapts them to our problem by working with their dual representation.

These codes, which we call the syndrome-trellis codes (STCs), can directly improve the security of many existing steganographic schemes, allowing them to communicate larger payloads at the same embedding distortion or to decrease the
distortion for a given payload. In addition, this work allows an iterative design of the distortion function to minimize detectability measured using blind steganalyzers on real cover sources [4], [5], [16]. This paper is organized as follows. In the next section, we introduce the central notion of a distortion function. The problem of embedding while minimizing distortion is formulated in Section III, where we introduce theoretical performance bounds as well as quantities for evaluating the performance of practical algorithms with respect to each other and the bounds. The syndrome coding method for steganographic communication is reviewed in Section IV. By pointing out the limitations of previous approaches, we motivate our contribution, which starts in Section V, where we introduce a class of syndrome-trellis codes for binary embedding operations.

We describe the construction and optimization of the codes and provide extensive experimental results on different distortion profiles including the wet paper channel. In Section VI, we show how to decompose the problem of embedding using nonbinary embedding operations to a series of binary problems using a multilayered approach so that practical algorithms can be realized using binary STCs. The application and merit of the proposed coding construction is demonstrated experimentally in Section VII on covers formed by digital images in raster and transform (JPEG) domains. Both the binary and nonbinary versions of payload and distortion-limited senders are tested by blind steganalysis. Finally, the paper is concluded in Section VIII. This paper is a journal version of [17] and [18], where the STCs and the multilayered construction were introduced. This paper unifies these methods into a complete and self-contained framework. Novel performance results and comparisons are included.

2) EXISTING SYSTEM:

- In special domain, the hiding process such as least significant bit (LSB) replacement, is done in special domain, while transform domain methods: hide data in another domain such as wavelet domain.
- Least significant bit (LSB) is the simplest form of Steganography. LSB is based on inserting data in the least significant bit of pixels, which lead to a slight change on the cover image that is not noticeable to human eye. Since this method can be easily cracked, it is more vulnerable to attacks.
- LSB method has intense affects on the statistical information of image like histogram. Attackers could be aware of a hidden communication by just checking the Histogram of an image. A good solution to eliminate this defect was LSB matching. LSB-Matching was a great step forward in Steganography methods and many others get ideas from it.

3) PROPOSED SYSTEM:

- In this paper it is planned to introduce a method that embed 2 bits information in a pixel and alter one bit from one bit plane but the message does not necessarily place in the least significant bit of pixel and second less significant bit plane and fourth less significant bit plane can also host the massage.
- Since in our method for embedding two bits message we alter just one bit plane, fewer pixels would be manipulated during embedding message in an image and it is expected for the steganalysis algorithm to have more difficulty detecting the covert communication. It is clear that in return complexity of the system would increase.
- In our method there are only three ways that a pixel is allowed to be changed:
  - Its least significant Bit would alter (So the gray level of the pixel would increased or decreased by one level)
  - The second less significant bit plane would alter (So the gray level of the pixel would increase or decrease by two levels)
  - The fourth less significant bit plane would alter (So the gray level of the pixel would increase or decrease by eight levels)

4) HARDWARE REQUIREMENTS:

- Processor: Any Processor above 500 MHz.
- Ram: 128Mb.
- Hard Disk: 10 GB.
- Compact Disk: 650 Mb.
- Input device: Standard Keyboard and Mouse.
- Output device: VGA and High Resolution Monitor

5) SOFTWARE REQUIREMENTS

- Operating System: Windows XP.
- Coding Language: JAVA
- Simulation: MATLAB (for checking Histogram of Original & Stegno image)
6) MODULE:

- Input Module:
- Watermark embedding
- Authenticator Watermark
- Spread Spectrum
- Watermarked content

**Modules Description:**

1) **Input Module:**
The Input Module is designed such a way that the proposed system must be capable of handling any type of data formats, such as if the user wishes to hide any image format then it must be compatible with all usual image formats such as jpg, gif, bmp, it must be also compatible with video formats such as avi, flv, wmf etc., and also it must be compatible with various document formats, so that the user can be able to use any formats to hide the secret data.

2) **Watermark embedding:**
Watermarking is a technology for embedding various types of information in digital content. In general, information for protecting copyrights and proving the validity of data is embedded as a watermark. Watermarked content can prove its origin, thereby protecting the data.

3) **Authenticator Watermark:**
In this module we encrypt the data embedded image. The purpose of authenticator watermark of a block is invariant in the watermark embedding process; hence the watermark can be extracted without referring to the original content. The encryption and decryption techniques used in this module.

4) **Spread Spectrum:**
We flip an edge pixel in binary images is equivalent to shifting the edge location horizontally one pixel and vertically one pixel. A horizontal edge exists if there is a transition between two neighboring pixels vertically and a vertical edge exists if there is a transition between two neighboring pixels horizontally. We use spread spectrum watermark morphological content.

5) **Watermarked content**
The watermarked content is obtained by computing the inverse for the main processing block to reconstruct its candidate pixels. Use this module we can see the original and watermarked content.

6) **Module I/O:**
**Module Input:**
We give original content as input with watermark data embedding. We view flipping an edge pixel in binary images as shifting the edge location one pixel horizontally and vertically.

**Module Output:**
The output of the project is we reconstruct the pixel horizontally and vertically. We can see the original watermarked data and embedding content.

7) **INPUT DESIGN:**
The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

7.1 **OBJECTIVES:**
1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.
8) OUTPUT DESIGN:
A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.
1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.
The output form of an information system should accomplish one or more of the following objectives.
- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

9. SYSTEM TESTING
The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

9.1 TYPES OF TESTS:
9.1.1 Unit testing:
Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

9.1.2 Integration testing:
Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

9.1.3 Functional test:
Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:
- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

9.1.4 System Test:
System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

9.1.5 White Box Testing:
White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.
9.1.6 Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box, you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

10.1 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

**Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

10.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

10.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed.

11. SAMPLE SCREENS:

11.1 Screenshots:
Reference


Authors Biography

ARUN KUMAR.R Completed M.TECH in CSE with specialization (COMPUTER SCIENCE ENGINEERING) from CVSR college of engineering affiliated by JNTUH in 2012. Currently he is working as an Assistant professor in CSE department at Vidya Jyothi Institute of Technology, Hyderabad. he is having 4 years experience as an assistant professor. He published 3 international journals. His areas of research interests include NETWORKS, ANALYSIS OF ALGORITHM, COMPILER AND LANGUAGE PROCESSING.