ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Hiding Secret Image in Video

Sarita Pandey
M.Tech,
Department of Information Security,
Disha Institute of Technology.

Vimal Parganiha, M.Tech
Assistant Professor,
Department of Information Security,
Disha Institute of Technology.

ABSTRACT:

The Internet is always vulnerable to interception by unauthorized people over the world. The importance of reducing a chance of the information being detected during the transmission is being an issue now days. Some solution to overcome these issues is cryptography, but once it is decrypted the information secrecy will not exits any more. Hiding data for confidentiality, this approach of information hiding can be extended to copyright protection for digital media. The importance as well as the technique used in implementing data hiding is trying to discuss in details here. The traditional LSB modification technique by randomly dispersing the bits of the message in the image and thus making it harder for unauthorized people to extract the original message, is vulnerable to lose of valuable hidden secrete information. Here, we propose a data hiding and extraction. Procedure for AVI (Audio Video Interleave) videos embedding the secret message bits in DCT higher order coefficients. The secret information taken here is an grayscale image pixel values. The grayscale pixel values are converted to binary values and embedded those values in higher order coefficient value of DCT of AVI video frames. Data Hiding and Extraction procedure are experimented successfully. Various experiment results are show here. All experiments are done using Matlab 2010a simulation software.

Keywords:
Video; frame; data hiding; security; DCT.

INTRODUCTION:

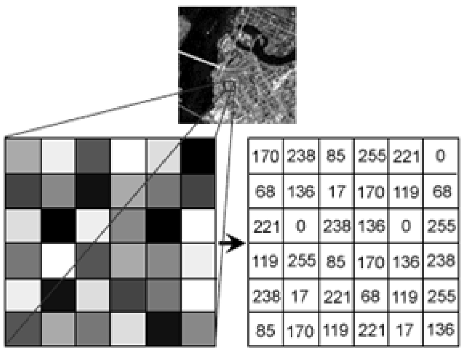
IMAGE:

An image is a two-dimensional picture, which has a similar appearance to some subject usually a physical object or a person.

Image is a two-dimensional, such as a photograph, screen display, and as well as a three-dimensional, such as a statue. They may be captured by optical devices—such as cameras, mirrors, lenses, telescopes, microscopes, etc. and natural objects and phenomena, such as the human eye or water surfaces. The word image is also used in the broader sense of any two-dimensional figure such as a map, a graph, a pie chart, or an abstract painting. In this wider sense, images can also be rendered manually, such as by drawing, painting, carving, rendered automatically by printing or computer graphics technology, or developed by a combination of methods, especially in a pseudo-photograph.

An image is a rectangular grid of pixels. It has a definite height and a definite width counted in pixels. Each pixel is square and has a fixed size on a given display. However different computer monitors may use different sized pixels. The pixels that constitute an image are ordered as a grid (columns and rows); each pixel consists of numbers representing magnitudes of brightness and color.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal



170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Each pixel has a color. The color is a 32-bit integer. The first eight bits determine the redness of the pixel, the next eight bits the greenness, the next eight bits the blueness, and the remaining eight bits the transparency of the pixel.

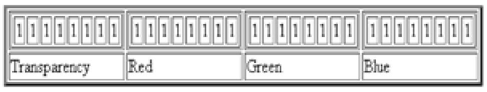
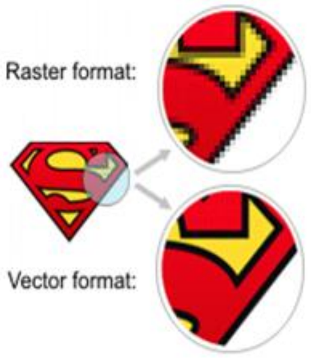


IMAGE FILE FORMATS:

Image file formats are standardized means of organizing and storing images. This entry is about digital image formats used to store photographic and other images. Image files are composed of either pixel or vector (geometric) data that are rasterized to pixels when displayed (with few exceptions) in a vector graphic display. Including proprietary types, there are hundreds of image file types. The PNG, JPEG, and GIF formats are most often used to display images on the Internet.



In addition to straight image formats, Metafile formats are portable formats which can include both raster and vector information. The metafile format is an intermediate format. Most Windows applications open metafiles and then save them in their own native format.

RASTER FORMATS:

These formats store images as bitmaps (also known as pixmaps).

- **JPEG/JFIF:**

JPEG (Joint Photographic Experts Group) is a compression method. JPEG compressed images are usually stored in the JFIF (JPEG File Interchange Format) file format. JPEG compression is lossy compression. Nearly every digital camera can save images in the JPEG/JFIF format, which supports 8 bits per color (red, green, blue) for a 24-bit total, producing relatively small files. Photographic images may be better stored in a lossless non-JPEG format if they will be re-edited, or if small "artifacts" are unacceptable. The JPEG/JFIF format also is used as the image compression algorithm in many Adobe PDF files.

- **EXIF:**

The EXIF (Exchangeable image file format) format is a file standard similar to the JFIF format with TIFF extensions. It is incorporated in the JPEG writing software used in most cameras. Its purpose is to record and to standardize the exchange of images with image metadata between digital cameras and editing and viewing software. The metadata are recorded for individual images and include such things as camera settings, time and date, shutter speed, exposure, image size, compression, name of camera, color information, etc. When images are viewed or edited by image editing software, all of this image information can be displayed.

- **TIFF:**

The TIFF (Tagged Image File Format) format is a flexible format that normally saves 8 bits or 16 bits per color (red, green, blue) for 24-bit and 48-bit totals, respectively, usually using either the TIFF or TIF filename extension. TIFFs are lossy and lossless. Some offer relatively good lossless compression for bi-level (black & white) images. Some digital cameras can save in TIFF format, using the LZW compression algorithm for lossless storage. TIFF image format is not widely supported by web browsers.

TIFF remains widely accepted as a photograph file standard in the printing business. TIFF can handle device-specific color spaces, such as the CMYK defined by a particular set of printing press inks.

- **PNG:**

The PNG (Portable Network Graphics) file format was created as the free, open-source successor to the GIF. The PNG file format supports true color (16 million colors) while the GIF supports only 256 colors. The PNG file excels when the image has large, uniformly colored areas. The lossless PNG format is best suited for editing pictures, and the lossy formats, like JPG, are best for the final distribution of photographic images, because JPG files are smaller than PNG files. PNG, an extensible file format for the lossless, portable, well-compressed storage of raster images. PNG provides a patent-free replacement for GIF and can also replace many common uses of TIFF. Indexed-color, grayscale, and true color images are supported, plus an optional alpha channel. PNG is designed to work well in online viewing applications, such as the World Wide Web. PNG is robust, providing both full file integrity checking and simple detection of common transmission errors.

- **GIF:**

GIF (Graphics Interchange Format) is limited to an 8-bit palette, or 256 colors. This makes the GIF format suitable for storing graphics with relatively few colors such as simple diagrams, shapes, logos and cartoon style images. The GIF format supports animation and is still widely used to provide image animation effects. It also uses a lossless compression that is more effective when large areas have a single color, and ineffective for detailed images or dithered images.

- **BMP:**

The BMP file format (Windows bitmap) handles graphics files within the Microsoft Windows OS. Typically, BMP files are uncompressed, hence they are large. The advantage is their simplicity and wide acceptance in Windows programs.

Digital vedio processing:

Digital video is a type of digital recording system that works by using a digital rather than an analog video signal. The terms camera, video camera, and camcorder are used interchangeably in this article.

History:

Starting in the late 1970s to the early 1980s, several types of video production equipment were introduced, such as time base correctors (TBC) and digital video effects (DVE) units (one of the former being the Thomson-CSF 9100 Digital Video Processor, an internally all-digital full-frame TBC introduced in 1980, and two of the latter being the Ampex ADO, and the Nippon Electric Corporation(NEC) DVE). They operated by taking a standard analog composite video input and digitizing it internally. This made it easier to either correct or enhance the video signal, as in the case of a TBC, or to manipulate and add effects to the video, in the case of a DVE unit. The digitized and processed video information from these units would then be converted back to standard analog video. Later on in the 1970s, manufacturers of professional video broadcast equipment, such as Bosch (through their Fernseh division), RCA, and Ampex developed prototype digital videotape recorders (VTR) in their research and development labs.

Bosch's machine used a modified 1" Type B transport, and recorded an early form of CCIR 601 digital video. Ampex's prototype digital video recorder used a modified 2" Quadruplex VTR (an Ampex AVR-3), but fitted with custom digital video electronics, and a special "octaplex" 8-head head wheel (regular analog 2" Quad machines only used 4 heads). The audio on Ampex's prototype digital machine, nicknamed by its developers as "Annie", still recorded the audio in analog as linear tracks on the tape, like 2" Quad. None of these machines from these manufacturers were ever marketed commercially, however. Digital video was first introduced commercially in 1986 with the Sony D-1 format, which recorded an uncompressed standard definition component video signal in digital form instead of the high-band analog forms that had been commonplace until then. Due to its expense, D-1 was used primarily by large television networks. It would eventually be replaced by cheaper systems using video compression, most notably Sony's Digital Betacam (still heavily used as an electronic field production (EFP) recording format by professional television producers) that were introduced into the network's television studios.

Information Hiding:

This article is about the computer programming concept. For the practice of hiding data in a message or file, see Steganography. For data encryption, see Obfuscated code. In computer science, information hiding is the principle of segregation of the design decisions in a computer program that are most likely to change, thus protecting other parts of the program from extensive modification if the design decision is changed. The protection involves providing a stable interface which protects the remainder of the program from the implementation (the details that are most likely to change). Written another way, information hiding is the ability to prevent certain aspects of a class or software component from being accessible to its clients, using either programming language features (like private variables) or an explicit exporting policy.

Over view:

The term encapsulation is often used interchangeably with information hiding. Not all agree on the distinctions between the two though; one may think of information hiding as being the principle and encapsulation being the technique. A software module hides information by encapsulating the information into a module or other construct which presents an interface. A common use of information hiding is to hide the physical storage layout for data so that if it is changed, the change is restricted to a small subset of the total program. For example, if a three-dimensional point (x,y,z) is represented in a program with three floating point scalar variables and later, the representation is changed to a single array variable of size three, a module designed with information hiding in mind would protect the remainder of the program from such a change. In object-oriented programming, information hiding (by way of nesting of types) reduces software development risk by shifting the code's dependency on an uncertain implementation (design decision) onto a well-defined interface. Clients of the interface perform operations purely through it so if the implementation changes, the clients do not have to change.

Encapsulation:

In his book on object-oriented design, Grady Booch defined encapsulation as "the process of compartmentalizing the elements of an abstraction that constitute its structure and behavior; encapsulation serves to separate the contractual interface of an abstraction and its implementation. The purpose is to achieve potential for change: the internal mechanisms of the component can be improved without impact on other components, or the component can be replaced with a different one that supports the same public interface. Encapsulation also protects the integrity of the component, by preventing users from setting the internal data of the component into an invalid or inconsistent state. Another benefit of encapsulation is that it reduces system complexity and thus increases robustness, by limiting the interdependencies between software components. In this sense, the idea of encapsulation is more general than how it is applied in

OOP:

For example, a relational database is encapsulated in the sense that its only public interface is a Query language (SQL for example), which hides all the internal machinery and data structures of the database management system. As such, encapsulation is a core principle of good software architecture, at every level of granularity. Encapsulating software behind an interface allows the construction of objects that mimic the behavior and interactions of objects in the real world. For example, a simple digital alarm clock is a real-world object that a lay person can use and understand. They can understand what the alarm clock does, and how to use it through the provided interface (buttons and screen), without having to understand every part inside of the clock. Similarly, if you replaced the clock with a different model, the lay person could continue to use it in the same way, provided that the interface works the same.

Explanation and Implementation:

A. Secret Message Formulation:

Here our secret message is an image (babra.bmp taken from Matlablibray shown in figure a gray level image. Pixel values of first 8x8 of 128x128 sized image is shown in figure . Each pixel intensity is then converted into equivalent binary values. As the size of the babra.bmp image is 128x128 we got 128x128x8=131072 bit (the secret message bits to be hidden).

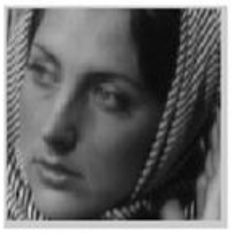
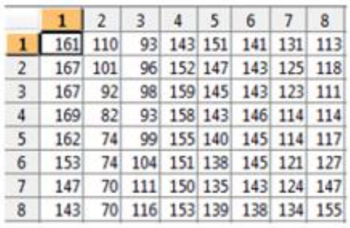


Fig. 6: Original image (secret message)



	1	2	3	4	5	6	7	8
1	161	110	93	143	151	141	131	113
2	167	101	96	152	147	143	125	118
3	167	92	98	159	145	143	123	111
4	169	82	93	158	143	146	114	114
5	162	74	99	155	140	145	114	117
6	153	74	104	151	138	145	121	127
7	147	70	111	150	135	143	124	147
8	143	70	116	153	139	138	134	155

figure (a)-(c) show conversion of original secret image to binary values for first 8x8 block producing 64x8 matrix containing 1's and -1's (converting 0's to -1).

B. Frame Extraction And Embedding Secret Message:

Here we have taken traffic.avi as a cover or host video and all frames are extracted (28 frames). The resolution of the original AVI is 120x160 pixels. The R-channel is used for encoding secret message after performing block DCT on those frames. As the size of original babra.bmp image is 128x128, hence we have to encode total 128x128x8 bits in the video frames. Here we embed 16 bits per 8x8 DCT higher order coefficient and in a particular frame we can embed.

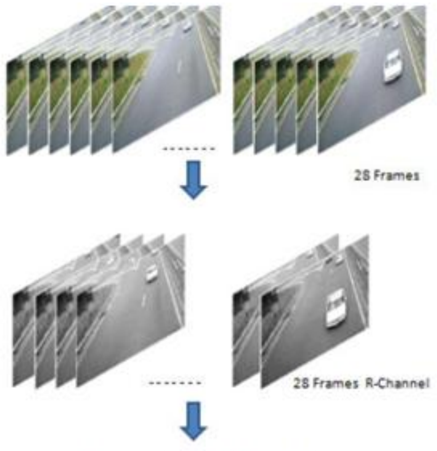


Fig Frame Extraction

C. Decoding and Reconstruction of Secret Message:

Decoding is done in reverse way of encoding.

Step 1: First video frames are extracted.

Step 2: R-channel frames are processed by 8x8 block DCT

Step 3: 8x8 block processed R-Channel original frame values

are subtracted to get secret message.

Step 4: From extracted secret message the image is reconstructed.

The figure 12 shows the reconstructed secret image that was embedded in the host video recovered without much more distortion.

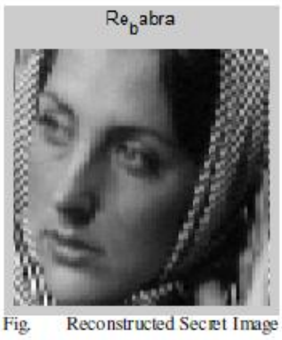


Fig. Reconstructed Secret Image

CONCLUSION:

Here we have discuss the process data hiding technique applied to AVI video to insert image secretly with much more perceptually lose of information to the cover media.

The method used here is tested for 28 frames only by embedding 128x128 image. More information can also be hidden in other channel of a frame giving more capacity of data to hide. Robustness of the scheme is not been discussed here. Quality of the video after encoding is almost similar to the original perceptually.

REFERENCES:

1. A.A.Zaidan, B.B.Zaidan, Fazidah Othman, "New Technique of Hidden Data in PE-File with in Unused Area One", International Journal of Computer and Electrical Engineering (IJCEE), Vol.1, No.5,ISSN: 1793-8198, 2009, pp 669-678.

2. Framework for Hidden Data in the Image Page within Executable File Using Computation between Advance Encryption Standared and Distortion Techniques", International Journal of Computer Science and Information Security (IJCSIS), Vol. 3,No 1 ISSN: 1947-5500, 2009, P.P 73-78.

3. G. Sahoo and R. K. Tiwari, " Designing an embedded Algorithm for Data Hiding using Steganographic Techniques by File Hybridization " , International Journal of Computer Science and Network Security (IJCSNS),Vol.8,No.1,January 2008.

4. S. Katzenbeisser, F. Petitcolas, " Information Hiding Techniques for Steganography and digital watermarking ",2000, pp 17-76.

5. Anderson. R. J , F. A. P. Petitcolas, " On the limits of steganography", IEEE J. Selected Areas in Commun., Vol .16, Issue 4, ISSN:0733-8716 , 1998, pp 474 – 481.

6. Debnath Bhattacharyya, P. Das, S. Mukherjee, D. Ganguly, S.K. Bandyopadhyay, Tai- hoon Kim, "A Secured Technique for Image Data Hiding", Communications in Computer and Information Science, Springer, June, 2009, Vol. 29, pp. 151-159.

7. Steganography and Steganalysis, J.R. Krenn, January 2004.

8. Spyridon K. Kapotas, Eleni E. Varsaki and Athanassios N. Skodras, "Data Hiding in H.264 Encoded Video Sequences", IEEE 9th Workshop on Multimedia Signal Processing, October 1-3, 2007, Crete, pp. 373-376

9. C. Cachin, "An Information-Theoretic Model for Steganography", in proceeding 2nd Information Hiding Workshop, vol. 1525, pp. 306-318, 1998.
10. D. Artz, "Digital Steganography: Hiding Data within Data", IEEE Internet Computing, pp. 75-80, May-Jun 2001.

11. E.T. Lin and E.J. Delp, "A Review of Data Hiding in Digital Images," in Proceedings of the Image Processing, Image Quality, Image CaptureSystems Conference, PICS '99, Ed., Apr. 1999, pp. 274--278.

12. F.A.P Peticolas, R.J. Anderson and M.G. Kuhn, "Information Hiding – A Survey", in proceeding of IEEE, pp. 1062-1078, July 1999.

13. J. Zollner, H. Federrath, H. Klimant, et al., "Modeling the Security of Steganographic Systems", in 2nd Workshop on Information Hiding, Portland, April 1998, pp. 345-355.