Development of a Facial Feature Based Image Steganography Technology

Pranay Marella
Department of Computer Science and Engineering
Mississippi State University
Starkville, MS, USA
pm1079@msstate.edu

Jeremy Straub, Benjamin Bernard
Department of Computer Science
North Dakota State University
Fargo, ND, USA
jeremy.straub@ndsu.edu,
ben.bernard@ndsu.edu

***Abstract*— A new image steganography method is proposed for data hiding. This method uses least significant bit (LSB) insertion to hide a message in one of the facial features of a given image. The proposed technique chooses an image of a face from a dataset of 8-bit color images of head poses and performs facial recognition on the image to extract the Cartesian coordinates of the eyes, mouth, and nose. A facial feature is chosen at random and each bit of the binary representation of the message is hidden at the least significant bit in the pixels of the chosen facial feature.**

Keywords—steganography, facial features, image data hiding, least significant bit, facial recognition

I. Introduction

As the world grows ever more digital, there is an increasing need for individuals, businesses, and nations to employ data hiding techniques when transferring sensitive data over the Internet. Steganography is one way to address this need. Steganography is a data hiding technique that seeks to make data invisible in plain sight. Unlike cryptography, the process of concealing the content of the method by encrypting it, steganography seeks to prevent discovery of the actual data itself as well as the fact that data is being transmitted at all. Steganography does this by hiding the data within a cover medium, making it less obvious that a message is being transmitted.

The proposed method builds on prior work that has been used to hide messages both in images and on the human body. In fact, one of the first known uses of steganography was when Greek messengers tattooed messages on their shaved heads and delivered the messages after their hair grew back [1].

Modern steganography can be used to exchange secret message between organizations, for securing online banking, and voting systems [2]. Not all uses are socially beneficial, however, as it is also used by attackers to send viruses and criminals use it for covert communication. Modern steganography is also a noteworthy alternative form of secret communication in countries where encryption is illegal [3]. It can be divided into several sub-fields: network steganography, text steganography, audio steganography, image steganography, and video steganography. Image steganography is the most widely used form of steganography [4].

In image steganography, picture data is usually stored in either 24-bit or 8-bit image files [5]. Pixels in images are made up of the three primary colors: red, green, and blue. Twenty-four-bit images use 3 bytes per pixel to represent a color value [5]. Most image steganography techniques fall within one of two domains: the spatial domain and the frequency domain (which is also sometimes referred to as the transform domain) [6]. Spatial domain techniques typically implement some form of a substitution method where redundant parts of a cover image are substituted with the secret message [4]. Frequency (or transform) domain techniques use the transformed coefficients of the input image, through different transforms such as the discrete Fourier transform and discrete cosine transform processes [2].

This paper presents initial work on an implementation that uses a targeted approach to image steganography. It uses least significant bit encoding on selected facial features in order to minimize data detectability within a carrier image.

Specifically, the proposed technique selects a picture from a dataset of pictures of people with different head poses, from the Head Pose Image Database [7]. After selecting the picture, it passes the picture through a facial feature recognition model, built using a face recognition module [8], and extracts the Cartesian (x, y) coordinates of three different features from the face: the eyes, the mouth, and the nose. Facial imagery was chosen specifically because of the availability of algorithms for identifying facial features. The program then chooses one of the features and uses least significant bit encoding to encode the secret message. When decoding, the user has to provide the picture that the data was encoded into and provide the facial feature that was chosen for encoding. These two pieces of information could easily be sent in a covert way, for example via a social media post.

II. Image Steganography

Most image steganography techniques can be classified under one of two domains: the spatial domain or the frequency domain. Key goal of steganography techniques include hiding the data in a way that avoids detection and maximizing the amount of hidden data that can be stored [6] in a given cover image.

A. Spatial Domain Techniques

Spatial domain techniques typically implement a substitution method where redundant parts of a cover image are substituted with the secret message [4]. In the spatial domain, a common technique that is used is Least Significant Bit (LSB) insertion. LSB insertion can follow two schemes: sequential or scattered. The insertion in sequential embedding schemes place the message bits in sequential order whereas in the scattered embedding scheme, the message bits are scattered throughout the image using a random sequence [9].

LSB insertion changes the last bit in every byte of memory in a pixel until the secret message has been embedded. For example: if one wanted to embed the letter "A" (in 8-bit binary representation: 10000001) into the following pixels:

11001110 00000010 11110010 00110011.

10110111 00000110 11110000 11110110

The least significant bit of each of the pixels is flipped to embed the A, if the flip is needed. This results in:

11001111 00000010 11110010 00110010.

10110110 00000110 11110000 11110111

When performing steganography, image file formats are critical to consider. Most images are not sent in a raw bitmap form, instead they are compressed. There are two forms of compression in images: lossless (reversible) and lossy compression (irreversible) compression [10]. Examples of lossless image formats include the Portable Network Graphics (PNG) format and the Graphics Interchange Format (GIF) [5]. The most common type of lossy image compression is the Joint Photographic Experts Group (JPEG) format. Least significant bit insertion is most effective in lossless image formats.

For most modern purposes, 24-bit bitmap images are commonly used. Because each pixel has red, green and blue components, 3 bits can be stored in each pixel. However, 24-bit images are larger in size than lower color fidelity ones, which can – in some environments – cause unwanted attention to be drawn to the picture. Alternately, 8-bit BMPs or GIFs could be used, however these formats may not be able to hold larger sized messages. Even 24-bit image files may have issues with message size. Distortion can occur, depending on the size of the message that is being encoded [11]. Splitting the message across multiple images can correct this problem but, again, may draw attention to the transmission in certain environments.

LSB insertion exploits the weakness of the human visual system, as compared to the fidelity of image color [12]. Because, in a 24-bit bitmap, there are 8 bits representing each of the red, green, and blue (RGB) values in each of the pixels, there is not a significant or visibly noticeable difference between 11111111 of blue and 11111110 of blue to the human visual system [6]. The LSB process can make significant changes to an image. On average, LSB insertion changes about half of the bits in an image [5].

Beyond this basic LSB technique, several enhanced techniques exist. Yang, et al. [13], for example, proposed an LSB method that used adjacent pixel value differentiating (PVD). This method considers the difference value of two consecutive pixels in order to estimate how many secret bits can be embedded into the two pixels. Gupta, et al. [11], proposed an enhanced least significant bit method for hiding data in a 24-bit image. The method proposes performing LSB insertion on only one-color channel to reduce distortion. Chang, et al. [14] developed an algorithm that embeds a message into a bitmap image by dividing the bitmap image into a bit-plane image from the LSB-plane to the MSB-plane for each pixel. Kharrazi, et al. [15] proposed an LSB +/- method which operates by incrementing or decrementing the last bit instead of changing it. LSB insertion can also be used in other cover mediums such as audio and video.

LSB insertion is easy to understand and implementation of this technique can be performed easily. LSB insertion creates a stego-image that is close to the cover image which makes it hard for someone to recognize any modifications. LSB's hidden data capacity is low, due to only one bit per pixel being used for data hiding [16]. LSB has some robustness problems. An attacker can prospectively retrieve the secret message by retrieving the LSBs [16], if the use of LSB steganography is detected or otherwise identified for a particular image. Utilizing only a fraction of the image may reduce the impact of this, somewhat, as it adds another piece of information that the prospective decoder needs to have in order to be successful in retrieving the stored data. The embedded data can also be lost by manipulating the image (resizing, compressing the image, etc.) [17].

B. Frequency Domain Techniques

Transform domain techniques use the transformed coefficients of the input image through different transforms such as the discrete Fourier transform and discrete cosine transform techniques, among others [2]. JPEG is the most common file type in use and has been projected to account for up to 95% of the images on the web [18]. JPEG takes high quality images and compresses them using the lossy discrete cosine transform (DCT) technique to achieve the requisite levels of compression [5].

The JPEG process works as follows [19], [20]: The original N x N image is split into 8 x 8 blocks. DCT is then applied to each of the blocks resulting in coefficients. In the quantization stage, the transformed coefficients are multiplied by a quantization coefficient and the result is rounded to the nearest integer. The rounded integers are ordered based on the zig-zag sequence, for encoding. Finally, Huffman encoding is applied to remove the redundant integers that occurred during quantization.

III. Proposed Technique

The proposed technique seeks to use an unusual method of determining where to store data, drawing on prior work in image processing and facial recognition, to reduce data and transmission detectability. The proposed approach can draw from existing images on social media and other sources and, through this, there is no need for any image file or key to be to be transferred.

The algorithm begins with the selection of an 8-bit color PNG image. A face recognition API [8] is used to identify the Cartesian coordinates of the lip, top lip, chin, left eye, left eyebrow, right eye, right eyebrow, nose bridge, and nose tip. It then combines the points of the bottom lip, top lip, and chin into

the mouth. It combines the points of the left eye, left eyebrow, right eye, and right eyebrow into the eyes, and it combines the nose bridge and nose tip into the nose. A facial feature is selected, and its Cartesian coordinates are added to and subtracted from by 1, 2, 3 and 4 and appended to the list of the Cartesian coordinates for use. A maximum payload capacity is determined and a message (or portion thereof), the image and coordinates are then sent to an encoding function. Each letter in the message is turned into 8-bit binary form based on ASCII value and the least significant bit of each of the pixels at the given Cartesian points are changed until the message is encoded.

To decode the message, the image is sent through the feature identification algorithm to identify the points for the feature that has been used. The addition and subtraction step are again performed for each point to get the complete point set. The image and the point set are then sent to the decoding function that retrieves the message from the identified points.

A test program was created which can both encode and decode data stored in images in this way. For testing, a collection of 8-bit color PNG images, which are a subset of the Head Pose Image Database [7], were used. The dimensions of all the images in the dataset are 384 pixels wide by 288 pixels high. The dataset was created by feeding the images from the Head Pose Image Database through a face recognition module [8] and removing the images that the face recognition module wasn't able to extract facial features from. For each test, a particular image was selected by the user.

The selected picture is sent to the facial feature recognition API [8] to get the coordinates. The user is prompted to select the facial feature for use for encoding. The name of the selected image and the feature that was chosen are displayed to the user and that information along with the list of points is passed into the encoding function. The program prompts the user to enter the message that is to be encoded with the maximum payload capacity displayed in bytes. This message, along with the list of points is passed into the encoding functions.

The receiver selects decode and provides the name of the picture they want to decode, the path to the dataset with the encoded picture, the path to the original dataset, and the facial feature to target the decoding at when decoding the image. The program takes the original image and sends it through the coordinate identification function to identify the points that the program needs to decode at. Thereafter, the points are sent to decoding function which retrieves the original message.

Figures 1 to 3 demonstrate the technique in action. Figure 1 shows it being used to encode data in the area of a subject's nose. Figure 2 shows data being encoded in a subject's mouth area and Figure 3 shows data being encoded in the area of a subject's eyes. The images in Figures 1 to 3 show the subject image before encoding (left), the region identified for data encoding (second from left), the image after the data was encoded (third from left) and the area where the data was hidden and recovered from (right).

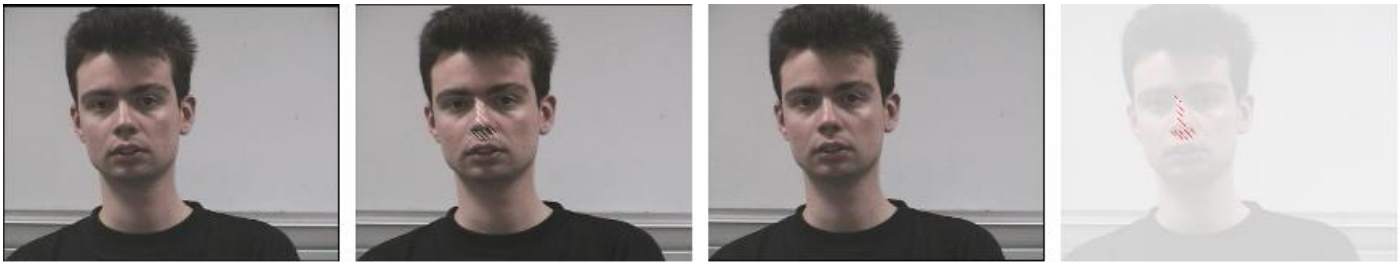

Fig. 1. Depicts feature identification of the subject's nose.

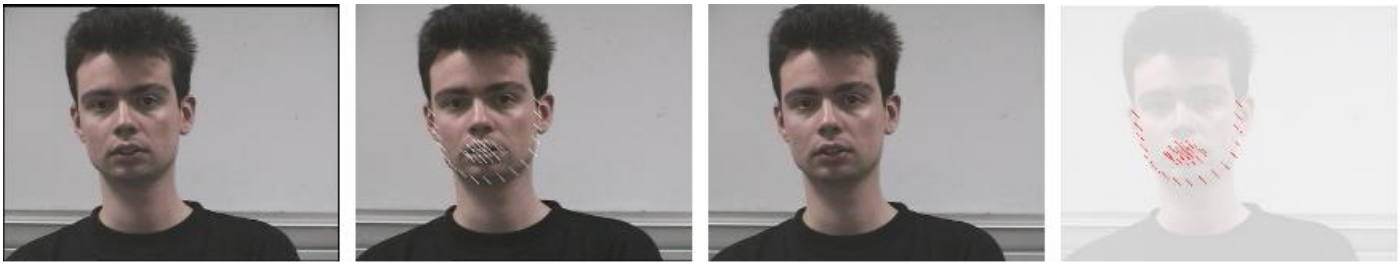

Fig. 2. Depicts feature identification of the subject's beard / facial hair.

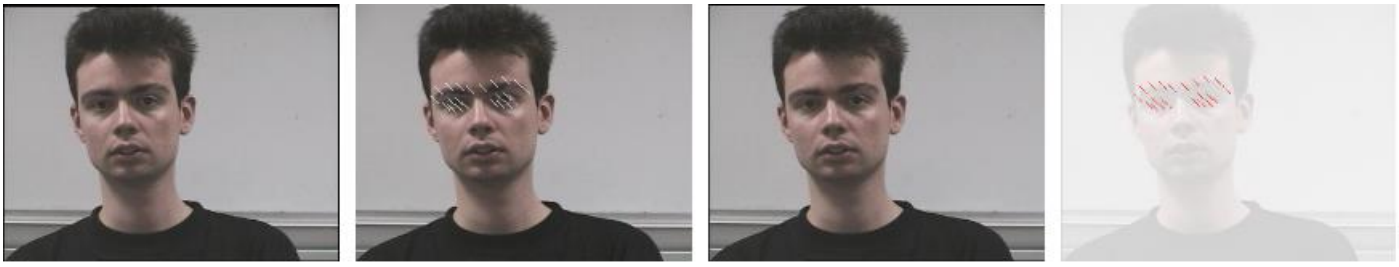

Fig. 3. Depicts feature identification of the subject's eyes and eyebrows.

Fig. 4. Image of Figure 1 region where data is encoded at 200% enlargement.

Figure 4 provides a closer look at the particular area of encoding for Figure 1 at a 200% zoom level. There is no visually obvious change to the image from the data's inclusion.

IV. Conclusions and Future Work

This paper has proposed a method to embed secret messages in images of human faces. This initial work has used 8-bit colored PNG images; however, it can be generalized to other formats and color fidelity levels. The proposed method embeds the secret message in areas based on facial feature identification. It can embed a secret message with little loss to image quality and no noticeable impact to the image. Face images were used initially because there are commonly available facial feature recognition algorithms; however, this technique could potentially be applied to other human features as well as objects of different types.

Future work will involve more testing of the proposed technique and, in particular, testing it with additional images, file types and datasets. It will also focus determining on the extent to which the payload size can be increased, relative to a given image size without impairing functionality. Changing the facial feature recognition model, to detect more points of the selected facial feature, will also be pursued to prospectively increase the supported message size. Studies of the technique with lossy techniques and combined with cryptographic techniques are also planned.

Acknowledgment

This work was supported by the U.S. National Science Foundation (award # 1757659). Facilities and some equipment were provided by the NDSU Institute for Cyber Security Education and Research. Thanks are also given to Dr. Simone Ludwig.

References

[1] M. Warkentin, M. B. Schmidt, and E. Bekkering, "Steganography and Steganalysis," in Enterprise Information Systems Assurance and System Security, IGI Global, 2006, pp. 287–294.

[2] K. Muhammad, J. Ahmad, M. Sajjad, and M. Zubair, "Secure Image Steganography using Cryptography and Image Transposition," NED Univ. J. Res., vol. XII, no. 4, pp. 81–91, 2015.

[3] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, Digital watermarking and steganography. Morgan Kaufmann Publishers, 2008.

[4] C. P. Sumathi, T. Santanam, and G. Umamaheswari, "A Study of Various Steganographic Techniques Used for Information Hiding," Int. J. Comput. Sci. Eng. Surv., vol. 4, no. 6, 2013.

[5] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," Computer (Long. Beach. Calif)., vol. 31, no. 2, pp. 26–34, Feb. 1998.

[6] Y. J. Chanu, T. Tuithung, and K. Manglem Singh, "A short survey on image steganography and steganalysis techniques," in 2012 3rd National Conference on Emerging Trends and Applications in Computer Science, 2012, pp. 52–55.

[7] N. Gourier, D. Hall, and J. L. Crowley, "Estimating Face Orientation from Robust Detection of Salient Facial Features," in ICPR INTERNATIONAL WORKSHOP ON VISUAL OBSERVATION OF DEICTIC GESTURES, 2004.

[8] A. Geitgey and J. Nazario, "Face Recognition," GitHub Repository, 2017.

[9] N. Hamid, A. Yahya, R. B. Ahmad, and O. M. Al-Qershi, "Image Steganography Techniques: An Overview," Int. J. Comput. Sci. Secur., vol. 6, no. 3, 2012.

[10] A. Said and W. A. Pearlman, "An image multiresolution representation for lossless and lossy compression," IEEE Trans. Image Process., vol. 5, no. 9, pp. 1303–1310, 1996.

[11] S. Gupta, G. Gujral, and N. Aggarwal, "Enhanced Least Significant Bit algorithm For Image Steganography," IJCEM Int. J. Comput. Eng. Manag., vol. 15, no. 4, 2012.

[12] S. Das, S. Das, B. Bandyopadhyay, and S. Sanyal, "Steganography and Steganalysis: Different Approaches," Int. J. Comput. Inf. Technol. Eng., vol. 2, no. 1, 2008.

[13] C.-H. Yang, C.-Y. Weng, S.-J. Wang, and H.-M. Sun, "Adaptive Data Hiding in Edge Areas of Images With Spatial LSB Domain Systems," IEEE Trans. Inf. Forensics Secur., vol. 3, no. 3, pp. 488–497, Sep. 2008.

[14] K. Chang, C. Jung, S. Lee, and W. Yang, "High Quality Perceptual Steganographic Techniques," Springer, Berlin, Heidelberg, 2004, pp. 518–531.

[15] M. Kharrazi, H. T. Sencar, and N. Memon, "Benchmarking steganographic and steganalysis techniques," 2005, vol. 5681, p. 252.

[16] N. Akhtar, P. Johri, and S. Khan, "Enhancing the Security and Quality of LSB Based Image Steganography," in 2013 5th International Conference on Computational Intelligence and Communication Networks, 2013, pp. 385–390.

[17] Mohit, "An Enhanced Least Significant Bit Steganography Technique," Int. J. Adv. Res. Comput. Eng. Technol., vol. 5, no. 6, 2016.

[18] H. Liang, X. Zhang, and H. Cheng, "Huffman-code based retrieval for encrypted JPEG images," J. Vis. Commun. Image Represent., vol. 61, pp. 149–156, May 2019.

[19] A. A. Attaby, M. F. M. Mursi Ahmed, and A. K. Alsammak, "Data hiding inside JPEG images with high resistance to steganalysis using a novel technique: DCT-M3," Ain Shams Eng. J., vol. 9, no. 4, pp. 1965–1974, Dec. 2018.

[20] Hsien-Wen Tseng and Chin-Chen Chang, "Steganography using JPEG-compressed images," in The Fourth International Conference onComputer and Information Technology, 2004. CIT '04., pp. 12–17.