Communication and Secrecy: Issues in Digital Stenography

Christopher Jaynes, Susan Landau, AND *Allen Hanson*

COINS Technical Report 96-77
October, 1996

Computer Vision Research Laboratory
and *The Theoretical Computer Science Laboratory*
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA 01003

Abstract

We present the issues involved in information *obfuscation* both for the embedding of information within digital signals and the detection of the hidden information. More standard cryptographic techniques acheive security through an encoding so that an evesdropper is unable to understand the signal. Obfuscation, preceded by an encoding, provides a method for the embedding of information within other signals so that an evesdropper is unable to detect the original information.

A framework, based on information theory, is presented that ensures embedded signals are undetectable under a set of conditions called the *Grammatic Invariance Principle.* An algorithm for embedding information within digital images (a technique often referred to as stenography) is developed under the framework and results are presented.

1 Introduction

Secure communication of electronic data is a problem of widespread concern not only to the military community but to the private sector as well. Schemes to provide secure communication have been devised and used throughout history and recently, there has been a vast amount of research directed towards cryptographic methods of data security. In this paper, however, we propose a technique that is not cryptographic in nature. We investigate *obfuscation*, a technique that allows data to be hidden within another signal rather than encoded directly.

Information obfuscation is concerned with the secrecy of communication as opposed to the security of an encoded message. Rather than altering a message so as to make it unreadable to an eavesdropper, obfuscation embeds data into another signal so that an eavesdropper is unaware of its existence. Of course, more standard cryptographic methods along with obfuscation can provide both hidden and encoded data.

Cryptography has a large body of research literature attributed to it. Obfuscation, on the other hand, has historically drawn little attention from the scientific community. There are several reasons for this. Typically, obfuscation is very inefficient. This is because, once a message has been embedded, both the message and the enveloping data must be stored or transmitted. This cost is a primary reason that obfuscation is not in widespread use today.

Obfuscation however, can provide a secret means of transmitting important data, without the obvious use of cryptography. Take the example of a secret agent stationed in a remote part of the world. Obfuscation can be used to transmit messages in a way that appears "normal" to intruders. Because it is inefficient, obfuscation would probably not be used to transmit entire conversations but is useful as a method of secretly exchanging cryptographic keys, for example.

This paper focus on the issues of data obfuscation within digital images. We call the embedded data the *plaintext*. The digital image is the *envelope* that obfuscates the plaintext. We would like to embed the plaintext so that the envelope appears unaltered to an eavesdropper. A discussion of how information can be successfully embedded in this way is a major contribution of this paper.

In the next section we introduce the problem within an information theoretic framework. This gives us with the terminology used to discuss obfuscation. The conditions needed to insure secrecy of obfuscation are introduced as the *Grammatic Invariance Principle*. This framework, along with the invariance principle provide us with the impetus for the algorithm developed in later sections.

Next, we show that the more obvious techniques for hiding information within a signal are insufficient and can be detected through a close analysis of the statistical properties of the enveloping signal. A program that embeds information within computer images is analyzed. The program does not conform to the Invariance Principle and is shown to be susceptible to standard computer vision analysis.

Finally, a "stronger" approach to hiding information within computer imagery is introduced. The new method is analyzed using the same techniques applied to the previous approaches. The results of the analysis are discussed and the new approach is shown to be an

improvement over existing techniques.

2 Information Theoretic Framework

Information theory is, in a large part, concerned with models of communication. Communication requires a sender S, a receiver R and a channel C along which the communication passes from S to R. Cryptographers are interested in allowing S and R to communicate any number of messages via C, securely. Security means an eavesdropper is unable to read messages M_i through channel C.

But what it means for an intruder to "read" a message can be defined in several different ways [Micali et.al.]. We take an information theoretic view: if the intruder is able to gain any information from message M_i we say that he has successfully *read* the message. In fact, the amount of information that the intruder is able to gain, with respect to the total information contained in the message, is a reasonable measure of the extent to which the communication system has been breached.

For a source that generates an encrypted message M^e, with each symbol m^e appearing in the message with a probability $p(m^e)$, the amount of information at the source or the *self-information* of the source S is defined by the **prior-entropy** at S:

$$H_{\text{prior}}(M^e) = -\sum_{i}^{n} P(m_i^e) \cdot \log P(m_i^e) \tag{1}$$

When R receives a symbol, it must be decoded (or interpreted). Let $P(m^d \mid m^e)$ be the probability that the receiver decodes m^e into the correct interpretation, m^d. We define the remaining information at the receiver as the conditional uncertainty about the decrypted message M^d given M^e. We call this measure the, the **post-entropy** of M^d at R upon receipt of M^e:

$$H_{\text{post}}(M^d \mid M^e) = -\sum_{i}^{n} P(m_i^d \mid M_i^e) \cdot logP(m_i^d \mid M_i^e) \tag{2}$$

During normal communication, in which the receiver R fully understands S, $P(m_i^d|m_i^e) = 1, \forall i$, and post-entropy is zero. That is, the receiver can compute, with zero uncertainty, the original message from the message transmitted. However, as an intruder is able to correctly interpret the encoded data (even probabilistically), the post-entropy is reduced. The reduction in the post entropy provides us with a measure of the amount of information an intruder has been able to gain.

2.1 Measuring Information Gain

This framework provides us with a more concrete understanding of what it means for an intruder to compromise a secure communication. The extent to which any recipient (including an unwanted one) is able to breech a secure system is precisely the amount of information that

is gained upon receipt of message M^e. This is simply the difference between the information produced at S and the remaining entropy at R.

$$\begin{aligned} I_g &= H_{prior}(M^e) - H_{post}(M^d \mid M^e) \\ &= (-\textstyle\sum_i^n P(m^e) \cdot \log P(m_i^e)) - (-\sum_i^n P(m_i^d \mid m_i^e) \cdot \log P(m_i^d \mid m_i^e)) \end{aligned} \tag{3}$$

Consider the case of two parties A and B that wish to securely communicate the result of flipping a coin several times. However, an eavesdropper, E, is listening in on the channel in hopes of discovering the results as well. In an attempt to fool E, A and B have agreed on a code in which A will flip the coin and simply transmit the inverse of the outcome of each toss. If the coin lands heads up, A transmits "Tails" and "Heads" otherwise. Because the two outcomes are equally likely, the prior-entropy of each message is given by equation 1 as:

$$\begin{gathered} H_{prior}(M^e) = -(\tfrac{1}{2} \cdot \log \tfrac{1}{2} + \tfrac{1}{2} \log \tfrac{1}{2}) \\ H_{prior}(M^e) = 1 \end{gathered} \tag{4}$$

Suppose that I recognizes that with each transmission of "heads" the actual result was tails with a probability of .75 (he has not yet discovered the true underlying code). Therefore the information gain, I_g, using this observation, is given by:

$$\begin{gathered} I_g = H(M^e) - (-[.75 \cdot \log .75 + \tfrac{1}{2} \log \tfrac{1}{2}]) \\ I_g = 1 - .56 = .44 \end{gathered} \tag{5}$$

The intruder, using an approximation of the actual code, has successfully gained .44 bits of information per message.

It is equation 3 that cryptographers are interested in minimizing. The obfuscation techniques proposed in this paper do not minimize equation 3 directly. Rather, we attempt to make the intruder *seem* to maximize his information gain, while the system remains secure. Wyner [Wyner] explores what he calls the *wiretap channel* and derives conditions under which the intruder cannot distinguish a message from noise contained in a communication channel. The obfuscation techniques developed later in this paper along with the post-entropy equation, are similar to this approach.

We use the fact that post-entropy is computable by R only subjectively. That is, R is sometimes unable to know the conditional probabilities used in equation 4 [Jaynes,'76]. The next section shows under what conditions R is misled in computing the post-entropy of a message. These conditions give us the requirements that should be met by a successful obfuscation system.

2.2 Information Theory and Communication

In this section we present the conditions necessary to modify a message without detection by the receiver. The following example provides an understanding of the principles behind

obfuscation and how the post-entropy measure is subjectively related to the receiver. This will lead to important guidelines for designing obfuscation techniques.

Imagine that we place a person in a large black box, inside the box the person is required to listen to questions to put her/him and answer by lighting a green light for yes and a red light for no on the front panel[1] [Massey].

The questions are required to be independent of each other and all questions must have possible answers that have equal distributions. In other words, the questioner has no statistical knowledge of the answers she can expect. For example, "Are you a woman?" is a valid question.

As the questioner proceeds she constructs a model of the person contained within the box. However, if the person in the box is replaced by a machine that randomly generates a response to each question, an equally valid model will be constructed by the questioner. Given the information, the questioner is unable to determine if a person is in the box or if the responses are generated randomly. In both cases, the questioner believes that she is gaining information about the contents of the box. In other words, for both cases, her post-entropy is zero.

One might argue that the questioner is far too restricted to gain useful information. For example, if we are allowed to ask questions that depend upon each other such as "Are you someone's mother?" followed by "Are you a woman?", then a random sequence of answers can be distinguished from a real person as being impossible or incoherent. Likewise, statistical models about humans provide information against which we can compare the sequence of responses from the box. That only a small percentage of humans are both female and taller than six feet, for example, may be valuable knowledge in guessing the true contents of the box. When the black box answers yes to "Are you a female who is taller than six feet" then we become less certain that the box contains a human.

Relaxing the restrictions in this way, gives us a clue at what a particular message source must do to fool a receivers post-entropy computation. In fact, we can formalize these concepts as the following principle.

The Grammatic Invariance Principle

A receiver R has a model of the source called M. The true information source, I produces symbols denoted by m_t^e at time t. The model source is known to produce symbols m_t^d at time t. $E[S_t]$ is the expected symbol produced by source S at time t. The following two conditions, then, make up the Grammatic Invariance Principle.

1. If I produces symbol, m_t^e and $m_t^d \rightarrow m_{\delta t}^d$ in model M,
   then $m_t^e \rightarrow m_{\delta t}^e$ and I must produce symbol $m_{\delta t}^e$. (Semantic Invariance)[2]
2. $E[M_t] = E[S_t]$, $\forall t$ (Syntactic Invariance)

The principle states that, for obfuscation to succeed, the information source must not deviate from the "grammar" that is defined by the domain. That is, obfuscation cannot alter

[1] A similar situation was described by J. Massey.

[2] a $\rightarrow$ b, means that a produces b. That is, if symbol a has been produced, then it follows that symbol b must also appear.

the semantic meaning of the signal for the receiver. Secondly, the syntactic properties of the signal must remain unchanged. In order for obfuscation to succeed, the intruder must believe that she has zero uncertainty about the true contents of the message. This can only occur under both conditions 1 and 2.

3 Detection of Hidden Signals

In order to elucidate the importance of conforming to the grammatic invariance principle given in the above section, we analyze an existing system that attempts to hide information in computer imagery. The system is called *Stego*[3] and was written by Romana Machado [Machado]. In describing the purpose of *Stego* Romana writes:

> ...What Stego does is hijack the least significant bit in each pixel of a picture. It uses that bit to store one bit of a secret message or file. Because digitized pictures have lots of pixels, it's possible to store lots of data in a single picture.

Immediately, we can see that the program does not satisfy the two requirements given above. By changing the least significant bit of each pixel, we are changing the statistical properties of something for which we have a model, specifically a digital image. Thus, the new image does not comply with the syntax of the chosen domain. Secondly, the bits in an image typically are not independent but depend on each other. Each pixel contributes to a more global picture with a semantic meaning. Changing each pixel independently will alter the semantics of the image and break the first condition of the invariance principle.

Therefore an intelligent questioner, one that has a model of an image in mind, can "query" an image and decide, with high probability, if the image was generated by a true source (a normal image) or one that is attempting to fool us (a Stego image).

To verify this, two programs were written in an attempt to discover hidden information within computer images. The first program uses the second order entropy measure to detect images that contain hidden signals. An expected entropy can be computed from a large sampling of digital images. By comparing the actual entropy with the expected value, we can tell when the entropy has increased (due to Stego) and the statistical model has been broken. Figure 1 shows the probability distribution of an image before and after 6000 bytes have been stored in it with the Stego algorithm. The statistical distribution is clearly changed.

3.1 Entropy Threshold

One possible means for detecting the presence of embedded information is to measure the information contained within the image itself. As more information is added to the image, the statistical distribution of the grey levels changes. Inter-pixel dependencies are destroyed as bits are flipped and this increases the local randomness, or entropy in the image.

Computation of the image entropy is a straightforward procedure that uses equation 1. However, we must assume that the image is an accurate sample of the information source that

[3]The term *Stego* is a derived from Stenography, a method of compressing written information through shorthand.

we wish to measure. In order to compute the probabilities correctly, we must assume that all of the pixel values appearing in the image define the possible outputs of the information source. If the information source is able to produce thousands of values for each pixel and we assume that the 255 different grey levels in the image represent the information sources alphabet, then clearly an incorrect entropy of the source will be computed.

This is not a strong assumption when working with computer images however. In fact, we can avoid this problem by working with images that are represented by one pixel per byte. Thus, the entire number of symbols is restricted to 255.

Higher order entropy calculations are performed by chaining together the rows and columns of pixels and then computing entropy over pairs, triples, and longer strings of pixels. The longer strings provide a new symbol, and possible alphabet. Higher order entropy computations provide a more accurate measure of the image entropy but are expensive to compute. The experiment below was restricted to the second order entropy measure, in which pairs of pixels are used in computing the image entropy.

3.2 Entropy Threshold Time Complexity

Detection of embedded signals should be efficient as well as accurate. Unfortunately, there is a direct tradeoff between accuracy and efficiency when estimating the image entropy. Higher order estimates provide a better estimate of the source entropy, but are costly to compute.

Computation of the image entropy can be broken into two steps. First, each element (pixel) in the signal, S, must be inspected in order to determine the probability of occurrence for each pixel value. If the image contains k pixels, then, for a first order estimate, we have $O(k)$. Higher order estimates are computed over several pixels and the probability of occurrence for strings of length n are computed. This still requires examination of each of the k pixels within the image only once. Thus, the first step is linear in the number of pixels, $O(k)$.

The second step in computing the image entropy entails summing the $p(x) \cdot \log[p(x)]$ values over the entire number of symbols. That is, given a source alphabet $\sum$, the product must be computed $|\sum|$ times. For higher order estimates, the size of the alphabet grows rapidly as does the time to compute the image entropy. The total time complexity for the Entropy Threshold calculation then is given by:

$$\begin{aligned} & O(k + |\sum|^n) \qquad \text{because } k << |\sum|^n \\ & = O(|\sum|^n) \end{aligned} \tag{6}$$

3.3 Test Results

Stego was run on 100 different images, with typical texts of varying lengths, while 100 images were left untouched. Entropy was computed in each image and then, using a single threshold the images were classified to determine which had been modified. The results are shown in table 1.

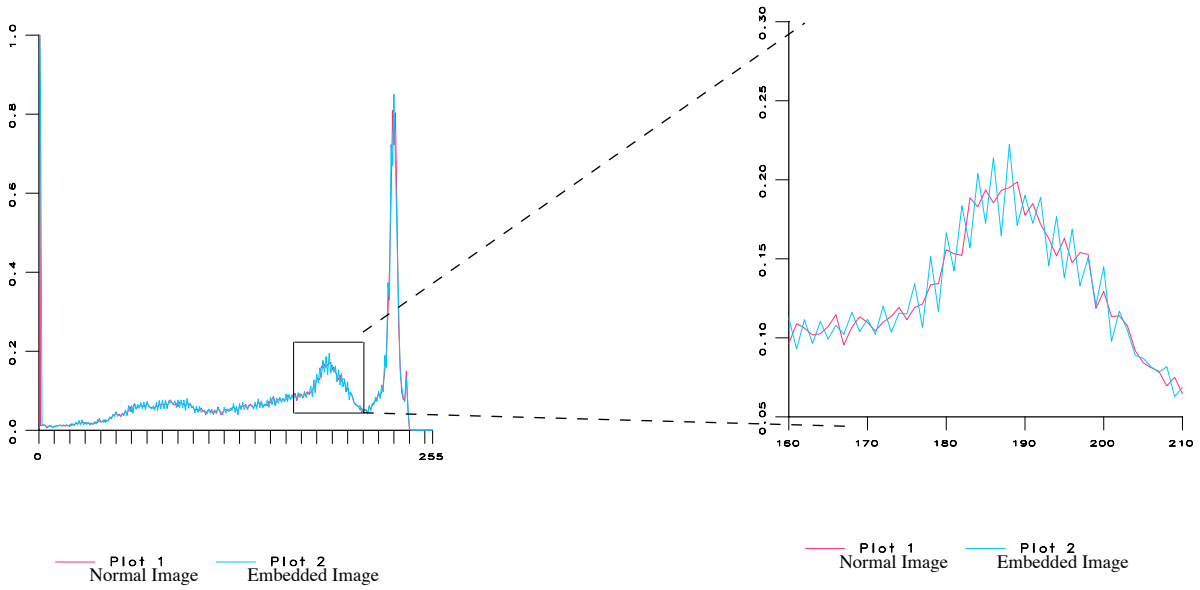


Figure 1: Introduction of "hidden" bits increases entropy in the pixel values.

Text Length (char)	Avg. Entropy (2nd order)	Detection Accuracy
50	3.9022	0.55
100	4.0812	0.57
1000	5.4102	0.65
2000	6.0141	0.73
Avg. Detected		62.5%

The results show that a single threshold on the 2nd order entropy cannot detect altered images with an acceptable accuracy. The problem in this method lies in the choice of a threshold. Choice of a threshold implys that expected entropy for all possible images is known. Clearly is not the case.

The test is too simple in that it can fooled by images that have an original entropy that deviates from the expected entropy. For example, the image of grass, shown in figure 3.3 contains a high 2nd order entropy and yielded a false positive in the experiment.

Another problem with the straight entropy measure above is that is changes very slowly with respect to the amount of information that has been embedded. The graph in figure 3.3 shows how the entropy changes as more information is added to the image.

3.4 Noise Measurements of the Probability Distribution Function

The main problem with the Entropy Threshold method of detecting embedded information lies in the choice of a threshold. There is no knowledge about an image entropy prior to embedding

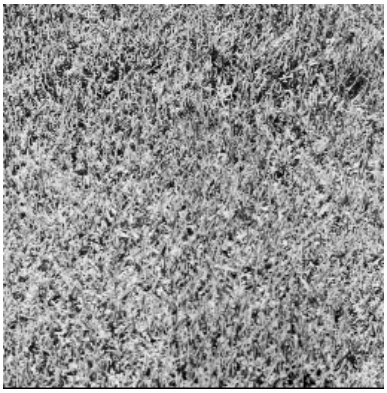


Figure 2: An image that contains unusually high entropy.

information. Measuring the current entropy then is clearly not the correct approach.

It is more important to measure a change in the probability distribution function (**pdf**) with respect to the distribution prior to the embedding process. Call the prior **pdf** of the image in question, $P(X)$. When information is added to the image the statistical nature of the image is changed. "Noise" has been added to the image and change in P occurs. If the final image **pdf** is $I(X)$ then we should like to detect the amount of noise in $I(X)$. By subtracting the two functions we arrive at:

$$D(X) = \mid I(X) - P(X) \mid \tag{7}$$

The new **pdf** D, gives a direct measure of the information change within the image. For images with no new information then $I(X) = P(X)$ and $\forall X\, D(X) = 0$.

When working with computer images, we have a discretized version of the **pdf**. If there are 255 grey levels, we define a new metric, $\mathcal{N}$ to be the average deviation of $I(X)$ from $P(X)$:

$$\mathcal{N} = \frac{\sum_{x=0}^{255} \mid I(x) - P(x) \mid}{255} \tag{8}$$

The problem with the metric $\mathcal{N}$ is that the prior **pdf** $P(X)$ remains unknown. Therefore, it must be estimated. The approach we have chosen is fairly simple. An estimated prior **pdf** is arrived at by smoothing $I(X)$. If information indeed was embedded into $I(X)$, then the smoothing operation will remove the local variations in the distribution function. If $I(X)$ is equal to $P(X)$ then there should be little change.

A median filter was chosen that conforms to the properties described above. The filter was applied to $I(X)$ by smoothing over the two previous and two succeeding samples of I.

Thus, a new metric ($\mathcal{N}$), can be computed directly from a single image as:

$$\mathcal{N} = \frac{\sum_{x=0}^{255} I(x) - \text{median}(I(x))}{255} \tag{9}$$

$\mathcal{N}$ gives a quantitative measure of the how "noisy" the **pdf** is but accounts for the general underlying shape of the **pdf** though the use of the median filter. Finally, the average deviation from the smooth curve yields $\mathcal{N}$.

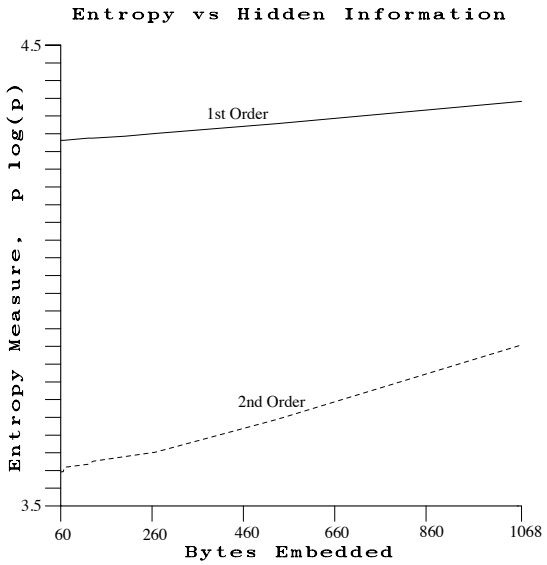


Figure 3: Although the simple entropy measures change as more information is added to the image. It changes very slowly and therefore cannot be used as a basis for detection.

3.5 Time Complexity of computing $\mathcal{N}$

Unlike the Entropy Threshold method, computation of the $\mathcal{N}$ metric does not require complex estimates of the image entropy. The time complexity depends only on the size of the image and the median filter. The steps in computing $\mathcal{N}$ for an image of size k, along with the time complexity for each step are shown below:

- Compute Histogram: $O(k)$
- Median Filter (filter size of m): $O(m * k)$
- Discrete Difference: $O(k)$
- Sum, average: $O(k)$

The total time complexity is the sum of the above values:

$$O(3k + mk) \tag{10}$$

For the experiments in this paper a the median filter was applied over 5 pixels. This is typically a small constant with respect to the total image size, k. Therefore, the final complexity is given by:

$$O(k) \tag{11}$$

Surprisingly, $\mathcal{N}$ can be computed in linear time. Unlike the Entropy Threshold method, there is no tradeoff between time and accuracy when using the $\mathcal{N}$ metric.

3.6 Test Results

Several experiments were run to arrive at empirical values for $\mathcal{N}$ both on unaltered images and images that had been embedded with information from the *Stego* program. The same set of 200 images from the Entropy Threshold experiment were used. 100 of the images images had text of lengths from 100 bytes to 4096 bytes embedded. An $\mathcal{N}$ value for all 200 images was computed and the results were encouraging. $\mathcal{N}$ was able to separate the two sets completely. Figure 3.6 shows the $\mathcal{N}$ values for the 200 random images.

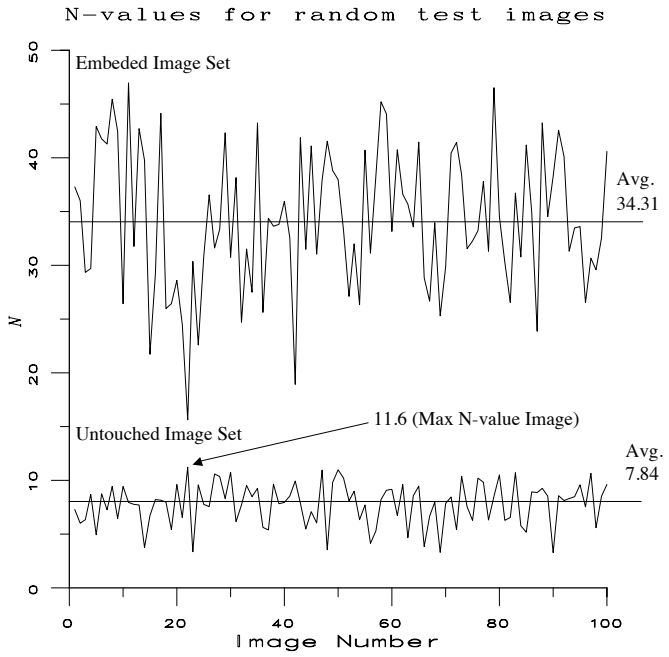


Figure 4: The $\mathcal{N}$, measure provides 100% separation on the test images.

The maximum value of $\mathcal{N}$ for an untouched image was 11.6. This was the "grass" image shown previously in figure 3.3. The unusually random nature of the image produced a high measure for $\mathcal{N}$. However, this measure was still smaller than the smallest value of $\mathcal{N}$ for the images that had been altered.

The same grass image was embedded with 2496 bytes of information, half of the image storage capacity. The $\mathcal{N}$-value was then recomputed. The recomputed value was 36.5, clearly within the range of other altered images. Because of the clear separation, a single threshold can be chosen in order to detect the presence of information on arbitrary images.

3.7 Detecting Information with the $\mathcal{N}$ metric, an Example

The steps in detecting whether an image contains embedded information are shown below from a single example. The example image is a closeup shot of an ant and was taken from the random set of 100 images that contained information. This particular image was selected as

a typical example of the type of images from the test set used in the above experiments. The image, after it has had 512 bytes embedded into it, is shown below.

Figure 5: A 256x256 image containing 512 bytes of hidden information.

Given image I and the precomputed threshold value T, the step-by-step procedure for detecting information within a single image is as follows:

1. Compute the probability density function P, of I. (Grey level histogram)
2. Compute the median filter, M of P.
3. Compute the difference, D by Subtracting P - M
   and taking the absolute value.
4. Compute the sum, S for all values of D
5. If $\frac{S}{\text{Number of possible grey levels}} > T$
   Image I contains information.

   else

   The image is unaltered.

3.7.1 The Probability Density Function

The first step in computing the $\mathcal{N}$ metric is to generate the histogram, or the probability density function of the image. Properties of the **pdf** are used to detect embedded information.

Next, the difference in the histogram of both the untouched image and the image containing the extra information is shown. The histogram can be thought of as the discretized probability density function that is used in computing $\mathcal{N}$.

3.7.2 Filtering the PDF

After the median filter is applied to both **pdf** functions, the resulting **pdf** is subtracted from the original to arrive at a difference curve. This curve, is shown below.

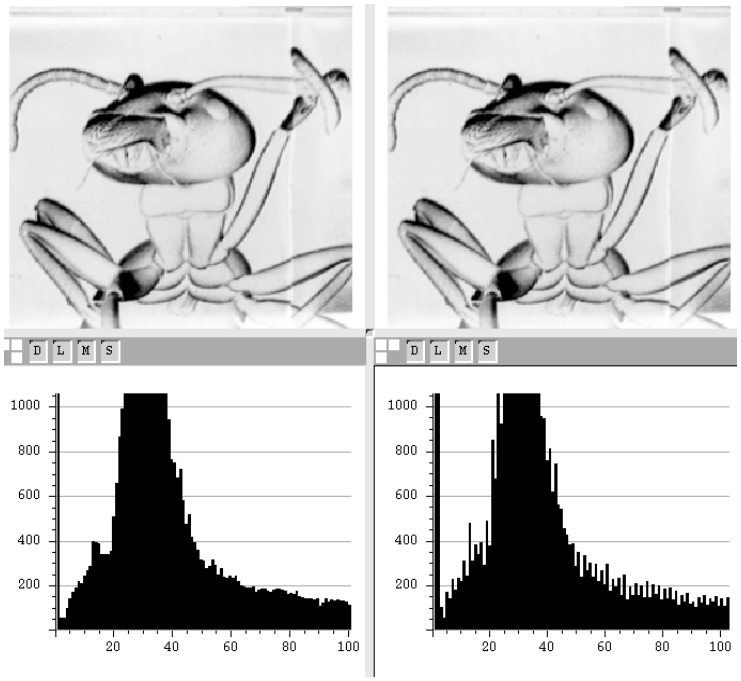


Figure 6: Untouched image (left) and embedded ant image (right) along with the **pdf** for each.

Once the **pdf** has been filtered the final value of $\mathcal{N}$ is computed. At each of the possible 255 grey levels, a difference between the filtered curve and the original **pdf** is computed. These differences are summed and averaged to compute the final value of $\mathcal{N}$. For the embedded ant image, $\mathcal{N} = 42.8$.

3.7.3 Image Classification

The final step in detecting hidden information is to simply classify the image according to the value of $\mathcal{N}$. A single threshold is applied and any image that exceed the threshold are considered to contain hidden information.

Thus, using standard computer vision techniques, we are able to detect significant changes in the probability density associated with an image that has information embedded into it using the obvious techniques.

However, this does not imply that hiding information is an impossible task. Indeed, one can imagine embedding only a few bits per image. Obviously, the sheer signal to noise ratio in this case makes it impossible to discover the embedded information and the message will remain undetectable by simple statistical tests. This is a degenerate case, however, and we are not concerned with it here. We would like to construct a secure, yet efficient, system.

By conforming to the Grammatical Invariance Principle, it is possible to embed information in a secure way. Similar to the *Stego* program, we use the domain of computer vision to show that obfuscation is indeed possible.

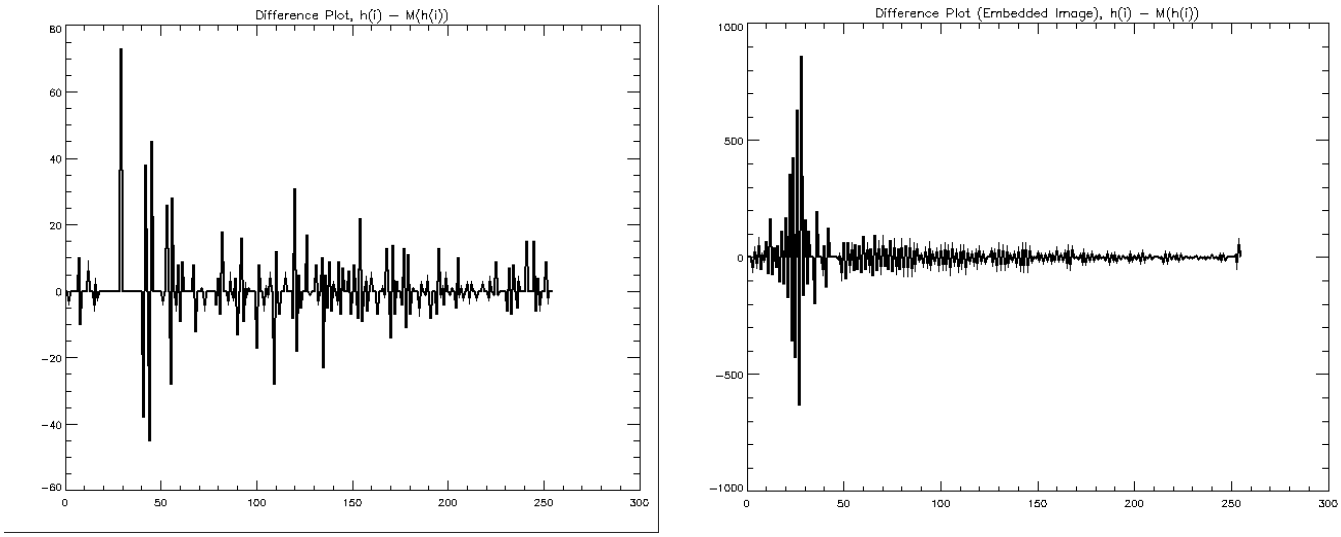


Figure 7: The resulting curves after $I(x) - P(x)$. The untouched image gives rise to a curve (left) that has a small amplitude and a corresponding $\mathcal{N}$ value of 4.45. The embedded image however, yields a much higher $\mathcal{N}$ value of 42.8

4 Obfuscation Within the Computer Vision Domain

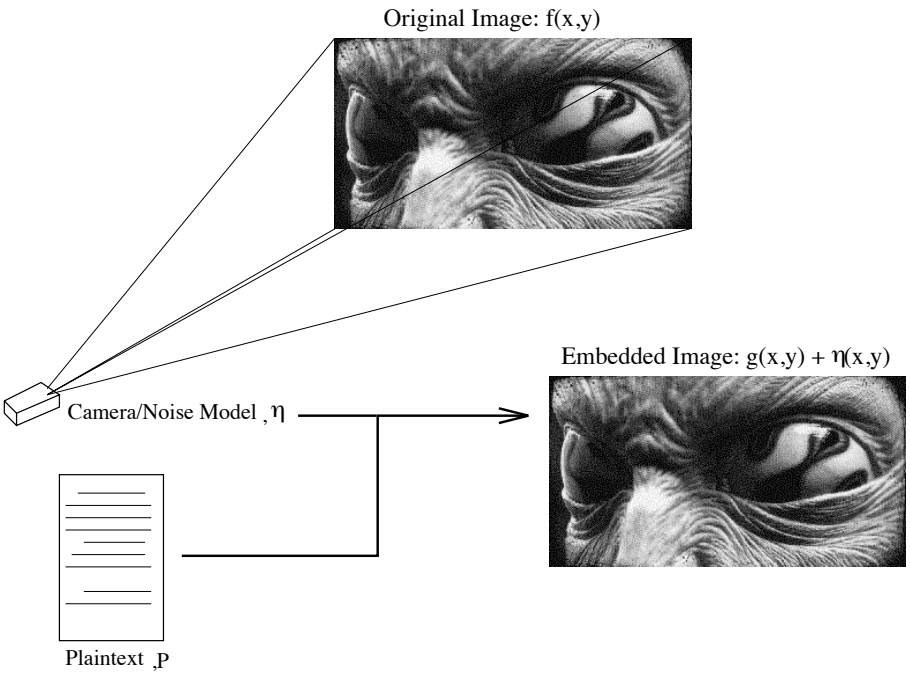
Previously, we have shown under what conditions the subjective computation of post-entropy can be misled. We show here that a system, conforming to these conditions can be constructed. Because digital imagery is a common form of media and it is well understood, it provides us with a rich domain for obfuscation.

Computer images are usually structured as an array of values (pixels), representing the image intensity corresponding to a point in the image plane. A simple image contains one byte per pixel, allowing 255 different levels of intensity. Color images are more complex, and are usually made up of three color planes, one for each of the primary colors. Each plane is an array of pixels, similar to a single greyscale image. This is shown in figure **??**

Thus, color images, have the potential for up to three times the amount of storage offered by a greyscale image. However, from the standpoint of complexity, the use of color imagery offers no improvement over the Obfuscation Algorithm on a single greyscale image. When using the Obfuscation Algorithm, there is no difference between a color image and a greyscale image that is three times as large. Therefore work here is restricted to 8-bit greyscale images, but can easily be extended to work within several different color planes in the obvious ways.

Rather than systematically store data in the last bit of an image, we disperse the embedded data according to the statistical model of a digital camera. By distributing information according to a given model, we generate a message that is conforms statistically to a typical digital image and does not violate the interdependencies between pixels given in most images.

Given a perfect image (one produced by a computer, for example), we can then alter the image in such a way as to model the physical properties of a camera. Passing the image through the camera model will introduce noise. The introduced noise will contain the embedded



information. The overview of this approach is shown in figure 4

When an image is captured by a digital camera, noise arises from two distinct sources. First, the lens system introduces distortions in the final picture. The second source of error lies in the digital system of the camera itself. The continuous signal is discretized and information is lost. The discrete signal is perturbed by noise within the digital circuitry before it is produced as a final array of grey-level values.

It is common (although somewhat arbitrary) to model the digital noise with a Gaussian distribution. [Gonzalez, Yu] With a standard deviation of σ and the "true" value of a pixel in the image is x_0 then the probability of the pixel appearing with a value x is given by:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{[-(x-x_0)^2/2\sigma^2]} \tag{12}$$

A new obfuscation algorithm that conforms to the requirements in section 2, can be constructed under this model of noise. We select an image that is free of sensor noise (i.e. one generated by a computer). Call this image I. The message, M is embedded into I so that is conforms to the given model of sensor noise.

This new image is indistinguishable from standard digital images up to the accuracy of the model. With the use of a more sophisticated model of the imaging process, it would be possible to detect "false" images in a manner similar to the methods used to detect *Stego* images. However, the approach introduced here will work with any arbitrarily accurate sensor model that can be provided.

4.1 The Information Obfuscation Algorithm

The Information Obfuscation Algorithm embeds information into a digital image according to a Gaussian noise distribution. This allows the information to masquerade as camera noise. In order to do this, a random number generator that has a distribution given by a Gaussian was written. Given a particular value for σ the random number generator will produce a value that can be thought of as the greyscale deviation for a given pixel.

The Gaussian distribution is shown in figure **??**. The distribution curve is shifted so that only positive values are produced by the random number generator. The sequence of values produced by the number generator is fixed and known ahead of time by both the embedding and extracting algorithms. The sequence, then, can be thought of as a function $\mathcal{X}$ where $\mathcal{X}(i)$ produces the greyscale deviation at pixel i.

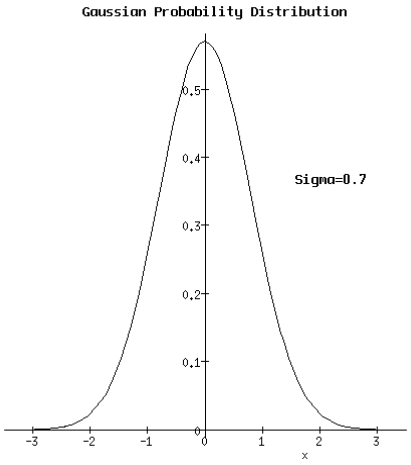


Figure 8: The noise model is based upon a simple Gaussian distribution. However, more complex models can easily be used with the Algorithms proposed here.

The output of the random number generator is discretized into integer values, so the integer value can be added directly to an image pixel to arrive at the new value. For example, given pixel i, with a value $V(i)$, and a random value of 1.12, the new value is $V(i) + 1.0$. That is, $\mathcal{X}(i) = 1.0$. This new value then, is able to store a single bit of information from a message. Values that deviate by more than one are able to store multiple bits of the message with a single pixel.

Information Obfuscation Algorithm

- Let $V(i)$ be the pixel value at i.
- $\star$ is a function such that $a \star M[b]$ embeds b bits of message M in the value a.
- Let $\mathcal{X}$ be the "noise distribution function" such that $\mathcal{X}(i) > 0$ iff $V(i)$ is a pixel that contains noise, and $\mathcal{X}$ distributes noise that conforms to a Gaussian.

1. For each pixel p_i in I or M is empty.
2. If $\mathcal{X}(i) > 0$
3. $V(i) = V(i) \star M_i[\mathcal{X}(i)]$
4. Remove $\mathcal{X}(i)$ bits from message M_i.

The algorithm assumes that the image is big enough to store all of the data contained in M. Because the amount of storage will be dependent on our noise model (particularly σ), we can not be sure that this is the case. We assume that the text has been broken into appropriately small pieces so that we can expect the information will fit into the image.[4]

It is important to note that the distribution of noise in the image is determined *a priori* and given by the $\mathcal{X}$ function. The receiver, using the known $\mathcal{X}$ function and an extraction algorithm, can invert the process and recover the original message simply by extracting the appropriate number of low order bits where $\mathcal{X}(i) > 0$.

4.2 Obfuscation Space Requirements

Because the obfuscation algorithm proposed here is probabilistic in nature, the amount of information that can be stored in a given image is not constant. The amount of information that can be stored in an image depends on the size of the image, and the particular random sequence, $\mathcal{X}$ used by the embedding process.

Assume we have an image with k bytes and a message that contains l bits. Let P_l^k be the probability that l bits can be stored in an image of k bytes. It is important to derive an expression for P_l^k so that image size and σ can be chosen to give a high probability that a message can be successfully embedded.

Let $P(x)$ be the probability that a pixel deviates by x in the image. $P(x)$ is governed by a Gaussian distribution. $P(x)$ is the function described in the Obfuscation Algorithm. For simplicity, assume that for all $x > 1.5$ the value of $P(x)$ is negligible (see figure 8). When $0.5 \leq x \leq 1.5$, a single bit is stored at the current pixel, thus we consider this probability only, and call it $P(1)$ for simplicity.

$$P(1) = P(0.5 \leq x \leq 1.5) = \sum_{0.5}^{1.5} \frac{1}{\sigma\sqrt{2\pi}} e^{[-x^2/2\sigma^2]} \tag{13}$$

Thus the probability that l bits cannot be stored in the image is:

[4]See section 4.2

$$(1 - P(1))^{k-l+1} \tag{14}$$

And, the probability of success is given by:

$$1 - (1 - P(1))^{k-l+1} \tag{15}$$

Because the number of bytes in an image is typically very large, the probability that messages of moderate lengths can be embedded is very high. In fact, the plot below shows, that even as the number of bits in the message grows very large, the probability of success remains high.

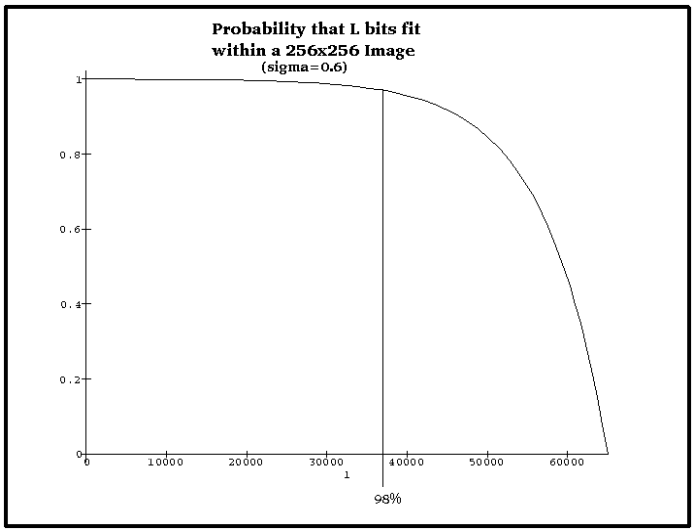


Figure 9: Probability that l bits can be stored in an 256x256 byte image. With σ=0.6 There is a 98% chance that 39544 bits (4943 bytes) can be stored in the image using the Obfuscation Algorithm.

5 Attacks on Obfuscation

There are two possible "attacks" on the obfuscation algorithm. First, an intruder may attempt to discover the existence of hidden information. These attacks will most likely use the same methods discussed in section 3. Secondly, the intruder may also want to extract the Embedded information from an image. The latter type of attack includes attacks of the former type.

It should be clear that discovering information within a signal can only occur when the Grammatic Invariance Principle has been broken. If the embedding process does not alter a signal with respect to some model, it will remain indistinguishable from noise.

Of course, an intruder may be able to develop a more sophisticated model than was used to embed the image. In this case, the information may be detectable as an anomaly in the

expected model. Thus, the more accurate the model used to embed the information the harder it will be to detect the hidden information.

If we assume that the embedding process uses an arbitrarily accurate model, an intruder is forced to actually extract possible sequences of information without knowing if it exists a priori. This type of attack is known as *Brute Force Search.*

5.1 Brute Force Search

A direct search of the image for information could eventually discover a recognizable string of symbols. However, if the intruder is unaware of the $\mathcal{X}$ and σ value used by the Obfuscation Algorithm, the search space is intractable.

Likewise, attempts to discover the value of $\mathcal{X}$ are equally futile. The number of possible $\mathcal{X}$ very large and guessing the correct $\mathcal{X}$ sequence is equivalent to guessing the location of the hidden information from within the image envelope.

6 Experimental Results

The strong obfuscation algorithm was implemented in the 'C' programming language and then tested for its strength. The set of 200 images used in the previous experiments were used. Half of the images had texts of lengths from 100 bytes to 4096 bytes embedded.

Both of the detection techniques that were used in the previous sections were applied to all 200 images. The same thresholds were applied to the data in order to classify the data as either embedded or regular.

Text Length (char)	Avg. Entropy / Detection Accuracy	$\mathcal{N}$-value / Detection Accuracy
50	3.8126 / 0.18	7.91 / 0.0
100	3.8130 / 0.18	8.15 / 0.0
1000	3.8910 / 0.21	7.99 / 0.0
2000	3.9001 / 0.21	8.21 / 0.0
Avg. Detected	19.5%	0.0%

Using the detection techniques from the previous experiments, the embedded information is no longer detectable to any degree of accuracy. The entropy threshold experiment was only able to perform at an accuracy of 19.5%. Simple yes/no guessing on each image could give us a 50% classification accuracy.

It is interesting to note that the $\mathcal{N}$-metric was unable to detect any of the embedded images. This is because a threshold was chosen that separated the previous set of images entirely. Because of the nature of the embedding algorithm, the $\mathcal{N}$-metric cannot be separated from the regular images. The $\mathcal{N}$ values for the 100 embedded images and the 100 regular images is shown below.

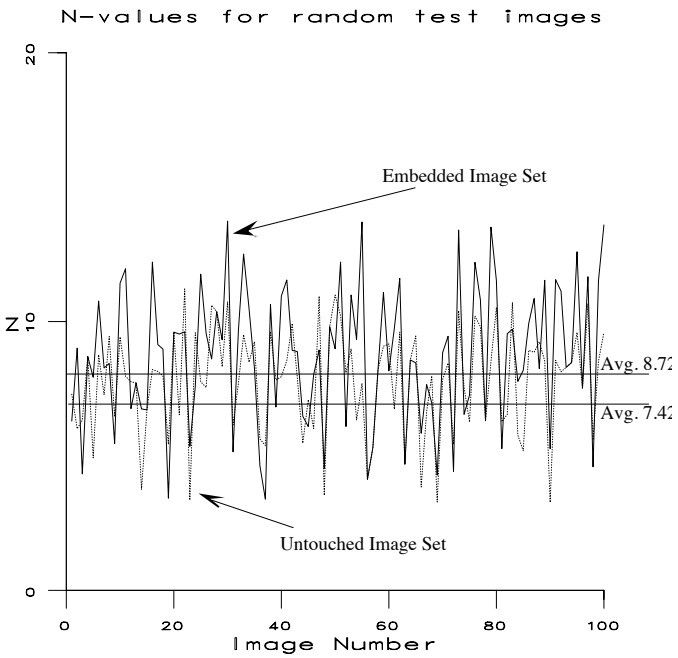


Figure 10: The $\mathcal{N}$, measure is unable to separate the test images.

7 Verification Techniques

Embedding a computer image with the plaintext clearly increases the amount of information contained within the image and the new image has very little redundancy. Thus, the image envelope is sensitive to noise during transmit. The corruption of only a few bits normally makes no difference when the information transmitted is an entire image. For an image that is acting as an envelope for important data, losing a few bits can alter the final extracted message.

An aggressive intruder, may simply alter the bits within the images in an attempt to corrupt or change the underlying image. Using the obfuscation techniques developed in this paper, we present a method of image verification that will make this detectable. The technique can be used for a number of different applications. Including verification of important digital photographs to detect tampering.

A simple method for image verification is to compute the checksum of the image and then embed this checksum using the obfuscation techniques proposed in the previous sections. We call the embedded checksum the image **signature**.

The pixel values of the image are simply summed to arrive at a value. The checksum can be stored as the least significant bit within the image bytes. Obviously, then, the checksum cannot include the least significant bit in its calculation. If the checksum exceeds an integer of k bits, the overflow bits are ignored. This checksum gives a unique value corresponding to the image. The probability that two grey-level images have the same checksum is:

$$P = \frac{1}{2^k} \tag{16}$$

Obviously, $k = 8$ is too small and yields a 1 in 256 chance of an altered image having the same checksum. However, P grows small very quickly and a size of $k = 24$ yields a P of $1/16,777,216$

Using this checksum method, an image can be verified easily. Simply extract the embedded signatures C and, using the upper seven bits of each pixel in the image, recompute the image sum S. If the two signatures match, the image is authentic.

An intruder cannot access the image signatures without knowledge of the $\mathcal{X}$ values used by the embedding process. Furthermore, the process of embedding the checksum into the image will alter any subsequent checksum values, making them useless.

7.1 Improvements on Verification

The simple checksum method is still susceptible to certain purposeful attacks. One can imagine an intruder that swaps pixels within the image without changing them. This will not effect the checksum of the image. Even worse, entire subsets of pixels that have the same checksum can be swapped in and out of the image without altering the checksum of the full image.

The way around this problem is to devise a signature that is position dependent. For example, a signature could be generated by either adding or subtracting the pixel values according to the value of the random sequence generated by the obfuscation algorithm. For example, for an image of length k, we can use $\mathcal{X}$ to either add or subtract values to the image pixel I.

$$C = \sum_{1}^{k} (-1)^{(\mathcal{X}^k)} * I(k) \tag{17}$$

Several signatures can be computed for a single image by using different variations on the above technique. For example, a straight summation of pixels can be computed and embedded followed by another checksum using the above equation. An intruder, then is unable to know the number of different signatures and, even when simply swapping pixels, runs the risk of modifying a particular signature.

8 Conclusions

We discussed, in section 2, the principles of successful information obfuscation. Using these principles, we have shown that the obvious techniques of information hiding do not withstand anything other than superficial analysis.

We proposed an alternative approach to obfuscation, one that conforms to the *Grammatic Invariance Principle*. The algorithm was shown to be resistant to the type of analysis applied to existing obfuscation techniques.

We have focused on the domain of computer vision because it provides us with a well studied source of signals. Computer images typically are redundant enough to allow room

for obfuscation to take place. However, the techniques applied here are not restricted to this domain.

The obfuscation principles explored here could be applied to embedding information within auditory signals through the modification of frequencies. An appropriate model of the digital recording process, along with a model of the physical system being recorded could be used to successfully embed information on audio recordings.

It has been pointed out that about 80 of spoken English is redundant. ??. This surprisingly large amount of redundancy could also be used to embed information. The model that the embedded information must conform to, in this case, is the complex system of the English language. In the future, we hope to explore methods to embed information within a arbitrary signals including written English text.

We have shown under what conditions obfuscation should take place and that it is indeed possible in practice. Although there is historically little research of obfuscation, we believe that it is an exciting area that has tangible value to the cryptographic community. Obfuscation provides us with the ability to authenticate digital images and, by combining obfuscation with standard cryptographic methods of encoding, communication of critical data can be secret as well as secure.

9 Acknowledgements

I wish to thank Professors Susan Landau and Al Hanson for their continued support and advice throughout this research. Al encouraged me to approach a novel topic that encompassed ideas far from my normal field of research. Susan introduced me to the deeper aspects of cryptography which inspired the work in this paper. She insisted that I state my case clearly and concisely, commented on earlier drafts, and is largely responsible for the lucidity of this paper.

References

[Abramson] N. Abramson, *Information Theory and Coding*, pp. 33-38 McGraw Hill, 1963.

[Bell] D. A. Bell, *Information Theory and its Engineering Applications*. Sir Issac Pitman and Sons Ltd, 1968.

[Denbigh] K. Denbigh and J. Denbigh, *Entropy in relation to incomplete knowledge*. Cambridge University Press, 1985.

[Denning] D. Denning, *Cryptography and Data Security*. Addison-Wesley, 1982.

[Gonzalez] R. Gonzalez and R. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[Jaynes,'76] E. Jaynes, *Foundations of Probability Theory, Statistical Inference, and Statistical Theories of Science*. W.L. Harper and C. Hooker eds. Reidel 1976.

[Jaynes,'85] E. Jaynes, *Maximum Entropy and Bayesian Methods in Inverse Problems*, C. Smith and W. Grandy, Jr. eds. Reidel, 1985.

[Koblitz] N. Koblitz, *A Course in Number Theory and Cryptography*. Springer-Verlag, 1987.

[Lint] J.H. van Lint, *Introduction to Coding Theory*, Second Addition, Springer-Verlag, 1991.

[Massey] J. L. Massey, "Information, Machines, and Men", in *Philosophy and Cybernetics*, F. Crosson, K. Sayre eds., Simon and Schuster,1968.

[Abu-Mostafa] Y. Abu-Mostafa, *Complexity in Information Theory*, Springer-Verlag, 1988.

[O'Neill] E. O'Neill, *Introduction to Statistical Optics*, Addison-Wesley, 1963.

[Kolmogorov] A. Kolmogorov, "Three approaches for defining the concept of information quantity", *Information Transmission*, vol. 1, pp. 3-11, 1965.

[Longpre] L. Longpre, *Resource Bounded Kolmogrov Complexity, A Link Between Computational Complexity and Information Theory*, PhD Thesis. Dept. of Computer Science. Cornell University, 1986.

[Machado] R. Machado, "What is Stego?", *The Stego InfoPage* Internet World Wide Web Site, http://www.nitv.net/ mech/Romana/stego.html, 1994.

[Micali et.al.] S. Micali, C. Rackoff, an and B. Sloan, "The notion of security for probabilistic cryptosystems", *SIAM Journal on Computing*, 1988.

[Sakrison] D. Sakrison, *Communication Theory: Transmission of Waveforms and Digital Information*. John Wiley and Sons, Inc. 1968.

[Huang] T. Huang, "Efficient Coding of Graphics", pp. 337-354. in *The Information Theory Approach to Communications*. Springer-Verlag, 1980.

[Shannon'48] C. E. Shannon, "Mathematical Theory of Communication", *Bell Syst. Tech. J.* vol. 27, pp. 379-423, 623-658. 1948.

[Shannon'49] C. E. Shannon, "Communication Theory of Secrecy Systems", *Bell Syst. Tech. J.* vol. 28, pp. 656-715. 1949.

[Shannon'51] C. E. Shannon "Prediction and Entropy of Printed English", *Bell Syst. Tech. J.* vol. 30, pp. 50-64. 1951.

[Weiner] N. Weiner, *Cybernetics*, John Wiley and Sons, 1948.

[Wyner] A. D. Wyner, "The wire-tap channel", *Bell Syst, Tech J.* vol. 54, pp. 1355-1387, 1975.

[Yu] F. T. S. Yu, "Image Restoration, Uncertainty, and Information", *Journal Opt. Soc. Am.,* , vol 58, 742 (1968).

[Yu] F. Yu, *Optics and Information Theory*, Wiley-Interscience, 1976.