画像内暗号の検証について

s98461-u
富岡　俊幸
計算機科学講座
田中研究室

平成 14 年 2 月 19 日

目 次

第1章　序論

通常の暗号化は、秘密情報を「読めなくする」技術であり、既にインターネット上でも実用化されている。そのことで秘密は保たれることが多いが、そこに「秘密データがある」ことを隠すことは出来ない。この「秘密データの存在」を隠すことは出来ないことで、第三者に却って秘密情報に関する関心をかき立ててしまうことになる。また、場合によっては、「何かを秘密にすること」そのものが人と人との関係に悪影響を及ぼす可能性もでてくる。

ここで、「秘密データの存在」を隠す、という別の角度からみた情報セキュリティ技術に着眼してみた。これは Steganography（ステガノグラフィー）と呼ばれている。Steganography は「秘密が存在する事実」を他人に気付かせない技術であり、先に述べたような心配が少なくて済む。この技術は、現在、インターネットによる通信が日常生活に欠かせない存在になる中で、他人に気付かれることなくこちらの意志を相手に伝え、相手の心をこちらに読みとる方法として必要とされている。

本研究では実際に画素置換型 Steganography 技術を用いて、画像に秘密データを隠蔽・抽出する。そして、抽出したデータと秘密データ（隠蔽したデータ）、元画像と隠蔽後画像をそれぞれ比較・検証する。

第2章　Steganography

2.1　Steganography とは

2.1.1　Steganography の概要

Steganography ということばの語源はギリシャ語に由来すると言われており、「こっそりと書き込んだもの」との意味合いをもつ。すなわち、他人に気付かれることなく、一方から他方に秘密情報を伝えるための極意とか技術のことである。「あぶり出し絵」はそのような技の最もわかりやすい例であると言える。

先に述べたように、steganography は、暗号化されているされていないにかかわらず、「秘密情報の存在」そのものを他人の目から「見えないように」するものである。すなわち秘密の存在を「非可視化」する技術であると言える。「Digetal Watermarking（電子透かし）」、「画像深層暗号化」などと呼ばれる「電子透かし技術」は、電子データに関する「オリジナリティ」や「所有権」の主張、或いは「認証」のために、少量の「目印情報」をその電子データに埋め込んでおく技術である。この場合の「価値有る情報」とは、表に現れている電子データそのものである。

これに対して、Steganography では、原則として、なるべく大量の「秘密情報」を人目にさらさないために、「ダミー情報で人目をそらす」ことが目的であり、表に現れる情報にはそれほど重要な価値がないことが前提である。これまでは、Steganography と言えば Watermarking のことであると理解されて来たが、これは従来の Steganogrpahy での「埋め込み容量」が少なかったことに由来している。

2.1.2　電子透かしとは

電子透かし（digital watermark）はデジタルコンテンツを不正利用から保護するために、人に気づかれないように著作権者を示すマーク（情報）を、画像や音などのデジタルコンテンツにそっと忍ばせる技術である。その発想はお札や株券などに入っている透かしからきている。お札や株券類の偽者防止には、透かしの有無が重視される。

この技術は写真やグラフィック・アート、音楽データ等のディジタル著作物を対象とするものであり、これも情報秘匿技術に属する。

2.1.3　電子透かしの原理

●電子透かしは、次のような原理に基づいている。

キャリアデータの変更：

キャリアの一部を電子透かし情報に基づき変更する。

※キャリア…電子透かしを埋め込む対象のこと。画像、音等。

画像、音声などのデジタルコンテンツの冗長性の利用：

画像や音データの一部を変更しても、冗長性の高いコンテンツの場合は全体的な構成に影響を及ぼさない

ことを利用し、データの一部を書き換えて電子透かしとする。

人間の感覚のルーズさ、生理学的特徴の利用：

上記　で変更する場合、さらに人間の感覚の特性をうまく利用して、人が気づき難い部分のデータを変更していくことにより電子透かしとする。

2.1.4 電子透かしと Steganography の相違点

技術的には同じであるが、目的が違う。 電子透かしは著作権を明示することが目的であるため、埋め込んだデータが簡単に取り除かれるようなものでは困る。 一方、Steganography はキャリアはあくまでオトリである。本当に意味があるのは埋め込まれている情報であって、キャリア自体には意味はない。最大の問題は、埋め込まれた情報が勝手に第三者に読まれることである。つまり、電子透かしの用途は著作権保護、Steganography の用途は情報隠蔽である。

電子透かしの用途目的は著作権保護、Steganography の用途目的は秘密通信である。
一見、著作権保護と秘密通信に関係があるように思えないが、電子透かしの生い立ちはもともと秘密通信技術として研究されていたものであった。このため、技術的には全く同一である。秘密通信といえば一般には暗号であるが、電子透かしのような考え方で、「Steganography」として区別している。

電子透かしと Steganography の相違点としては、画像等に別の情報を目立たないようにして埋め込む技術であるから技術自体は先に述べたように同じである。電子透かしはコンテンツの著作権を明らかにするために著作者名などを埋め込んでいるので、主役はキャリアである。一方、Steganography では、キャリアはあくまでオトリとして使う。本当に意味があるのは埋め込まれている情報である。この情報を相手に伝えるために画像を借りて表面上偽装して通信する用途がある。

Steganography で用いられる技術は電子透かしと基本的には同じ技術だが、目的が違うために要求される機能はかなり異なる。電子透かしではキャリアが主でその著作権者を明示するものなので、電子透かしは簡単に取り除かれては困る。しかし、Steganography ではあくまで隠れている情報が主であるから、万一その情報が取り除かれても何も問題ない。逆に再度送信すれば済むことである。最大の問題は、埋め込まれた情報が勝手に第三者に読まれることである。このように、技術のルーツは同一でも目的によりかなり異なったものであることがわかる。

2.2 Steganographyの種類

2.2.1 画素置換型電子透かし

多値画像（白黒濃淡画像やカラー画像）へのSteganographyで有名なのは「画素置換型のSteganography」で、これは、それぞれの画素の明るさや色情報を複数ビットで記述する。カラー画像の場合だと、R（赤）、G（緑）、B（青）の各成分の強さをそれぞれ８ビットで表現することが多い。例えば、windowsでのBITMAPファイルなどが挙げられる。ここでは、そのような形式での画像データの第１ビット目を最上位、第８ビット目を最下位ビットとして扱う。そうした場合、画像を8枚のビットプレーンに分けることが出来き、最下位のプレーン0は各画素のLSB (least significant bit) を表し、最上位のプレーン7はMSB (most significant bit) の集合体である。

画素置換型というのは、プレーンそのものを署名データと置き換える方法である。最下位ビットは画像全体への視覚的影響が少なく、別のデータと置き換えても変化が少ない。よって、最下位ビットに秘密情報を埋め込んでも（置き換えても）、視覚的な埋め込みの証拠は残らず、秘密データを隠すことができる。この最下位ビットに埋め込む方法は、典型的なSteganography方式である。

一方、上位のビットプレーンで秘密情報との置き換えを行なうと画像に影響が出る。これは、上位ビットプレーンは画素の有効な情報を保有しているので、その成分を透かし情報に置き換えると視覚に著しく影響を与えるからである。この方式で埋め込まれるデータ量は、ダミー画像ファイルサイズの１/８（12.5 %) を越えることはない。最下位ビットだけでなく、その他のビットに埋め込んでも画質が劣化しない場合があり、その場合の埋め込み容量はダミー画像ファイルサイズの１/４程度になるが、この定まった部分に情報を埋め込む方式は、埋め込み情報を比較的容易に見破られてしまうという欠点がある。（ビットプレーンを１枚ずつ取り出して調べると、すぐに透かしを探知することが出来できてしまう）ただし、これは、デジタルコンテンツ（画像等）に秘密情報が埋め込まれていることを知っていることが前提である。

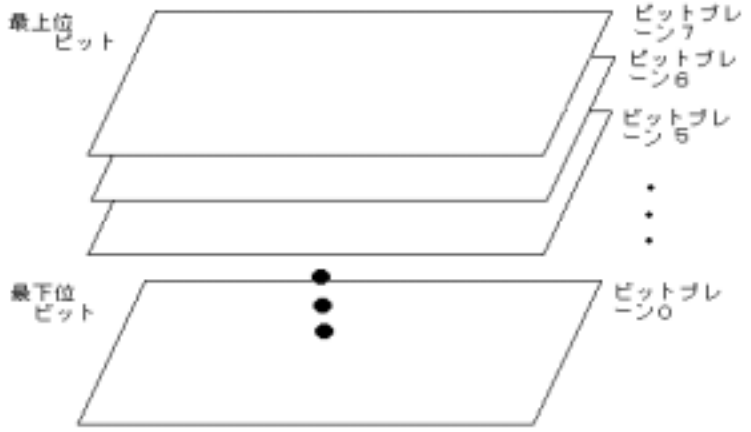

図 2.1: ビットプレーンの考え方

2.2.2 画像の周波数特性を利用した Steganography

これは、画像や音声を周波数成分に変換し、特定の周波数成分に透かし情報を入れる方式である。周波数成分への変換には FFT（高速フーリエ変換）や DCT（逆コサイン変換）が利用される変換時にどのように周波数成分に秘密情報を入れるかはさらに、いくつもの工夫が考えられている。

音の周波数は理解し易い。高い音は周波数が高く、低い音は周波数が低いと表す。画像も同じように映像自体を周波数で表せるが、これを波数領域での表現という。周波数領域にすると画像の特徴が把握しやすくなる。画像も音もあるいはどんな複雑な信号であってもすべての信号は、sin,cos の単純な波形の集まりで構成されている。このため周波数領域へ変換すると画像の骨格を形成しているのか、どの周波数でどの部分が無駄な部分であるか、ということがわかり易くなる。　周波数領域への代表的な変換方法として、フーリエ変換がある。任意の信号はフーリエ変換を使うとその波形を構成している sin, cos の単純な波形の和に分割できる。

以下に、フーリエ変換の式を示す。

フーリエ変換の式：

$$f(t) = A0 + \frac{A1\cos 2\ \pi\ t}{T} + \frac{A2\cos 2\ \pi\ t}{T} + \cdots$$

$$f(t) = A0 + \frac{A1\cos 2\ \pi\ t}{T} + \frac{A2\cos 2\ \pi\ t}{T} + \cdots$$

ただし

$$A0 = \frac{1}{T}\int_0^T f(t)dt$$

$$An = \frac{2}{T}\int_0^T f(t)\frac{\cos 2\ \pi\ nt}{T}dt$$

$$Bn = \frac{2}{T}\int_0^T f(t)\frac{\cos 2\ \pi\ nt}{T}dt$$

$$n = 0, 1, 2, \cdots$$

2.2.3　色情報とその輝度を利用した Steganography

色情報を利用した電子透かしをいえば対象はカラー画像であるが、カラー画像に対する Steganography も実際には明るさ情報を利用することが多い。その場合にはＲＧＢ信号から輝度信号＋色信号の形の信号を作り輝度信号を利用する。

●カラー情報から輝度信号への変換方法

Ｒ，Ｇ，Ｂの３色からなる画像信号はそのまま使われるだけでなく、さまざまな表現形式がある。それらの主たる目的は輝度（明るさ）　と色の信号に分離することにより送信効果率等を上げることにある。

最も代表的なものはカラーテレビで使われているＹＩＱ表色系である。

白黒ＴＶとカラーＴＶの互換性を保つこと、および色成分の帯域を狭くすることなどのために使われている。

各成分への変換は次式で与えられている。

$$Y = 0.299R + 0.587G + 0.114B$$
$$I = 0.596R - 0.274G - 0.322B$$
$$Q = 0.211R - 0.523G + 0.312B$$

ここで　　Y：輝度成分
　　I, Q：色成分
　　R, G, B：赤、緑、青の各信号レベル

を表している。

使用については無条件で、この式に当てはめていくだけでよい。式より、Ｙ信号はＲ，Ｇ，Ｂの和になっている　ので一番大きな値になることがわかる。さらにＹ信号のなかでもＧ（緑）成分の係数は、0.587 で一番大きな値となる　ため影響が大きいことを示している。逆にＩ，Ｑの成分を色成分と表すが、相対的に小さい値となっていることがわかる。

※ＹＩＱ表色系：人間の目にはカラー成分に対して視感度が低いため、輝度成分を効果的に送信するための分解方法。

※帯域：情報を伝達するのに必要な周波数範囲。同じ情報を送るのならより狭い帯域を占有するほうが望ましい。

※ＹＣｂＣｒ表色系：ＪＰＥＧ等で利用されている表色系でＹＩＱ方式同様、色成分に対しては感度が低いことを前提にしている。

●色信号自体を利用した Steganography

人間の目には明るさより色に対して感度が低いという特徴があった。このことと、カラー画像は色成分だけ冗長性が増加している、という特徴により色情報を操作すると電子透かしが比較的埋め込みやすくなる。

具体的な方法として：

全画素について色をわずかに変更する方法:

しかし、全画素では１ビットしかデータを埋め込めないため、いくつかのブロックに分割する方法が考えられている。

透かし文字の部分だけパラメータを変更する方法:

表色変換するとき、すべての画素に対して同じ変換をするのではなく、電子透かしの部分だけ変換パラメータを少し変え、逆変換処理を行なう。

※表色変換：さまざまな目的に応じた色表現方法がある。

ＹＩＱ，ＹＣｂＣｒ表色系など。

以下に、このタイプ　の原理を挙げる

基本的には表色変換系の係数を秘密情報に基づいて変更する方式となっている。埋め込まれる秘密情報は画像のみとなり、文字情報は埋め込むことができない。埋め込みべきロゴ画像（透かし画像）はあらかじめ二値画像で作っておく。

ＹＣＣ表色変換系を利用した計算方法を以下に示す。

$$Y = 0.299R + (0.587 - \quad x)G + (0.114 + \quad x)B$$
$$Cb = (-0.299 + \quad x)R - 0.331G + (0.500 - \quad x)B$$
$$Cr = (0.500 - \quad x)R - (0.419 + \quad x)G + 0.081B$$

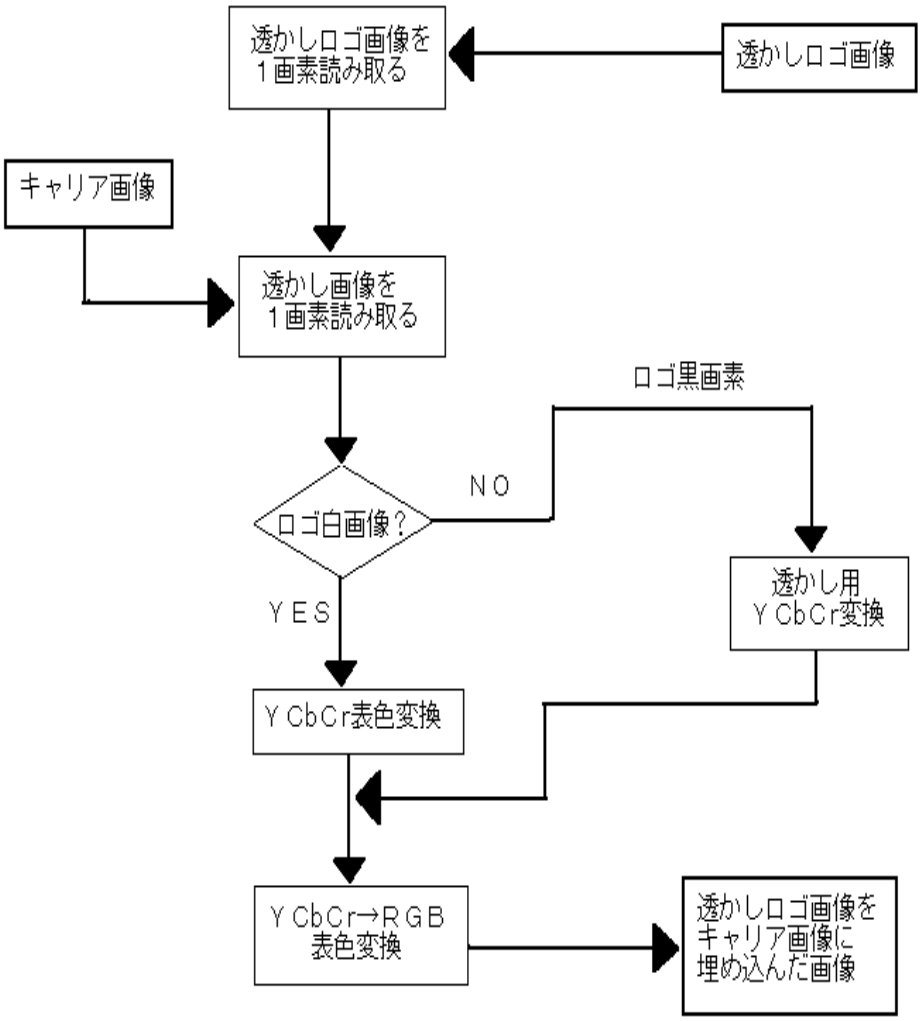

図 2.2: 表色変換系を用いた可視型 Steganography(　タイプの原理)

第３章　画像のSteganography

3.1　ＢＭＰについて

ビットマップとは画像データのフォーマットの一つである。

Windows のビットマップには、通常のＧＤＩで使われるデバイス依存ビットマップ（ＤＤＢ）の他にもＤＩＢ（デバイス独立ビットマップ）というものがある。これは、ピクセルの色を単純な配列としてまとめたビットマップで、プログラムの中で直接配列を使うのと同じようなに（あるいは配列としても）扱う事が可能である。

フルカラービットマップは、１ピクセルが３バイト（上位から順にＢＧＲ）の構成をしている。ビットマップの本体は、このピクセルの情報が下から上、左から右の順に並んでいるが、これは通常のビットマップの構造とは上下逆である。

ＤＩＢを作る時は、各ピクセルの色を格納したビットマップ本体の他に、ビットマップの大きさや色の形式などの情報を収めた BITMAPINFOHEADER とその BITMAPINFOHEADER とパレットをまとめた BITMAPINFO という２つの構造体がある。最も、フルカラーではパレットは使わないので、BITMAPINFO に入るのは BITMAPINFOHEADER だけである。実際にＤＩＢを扱う時には、BITMAPINFO 構造体と各ピクセルの色の情報を格納したビットマップ本体へのポインタが必要になる。

※ＤＩＢの座標は、一番下が原点（０，０）になっている

3.1.1　ビットマップの構造

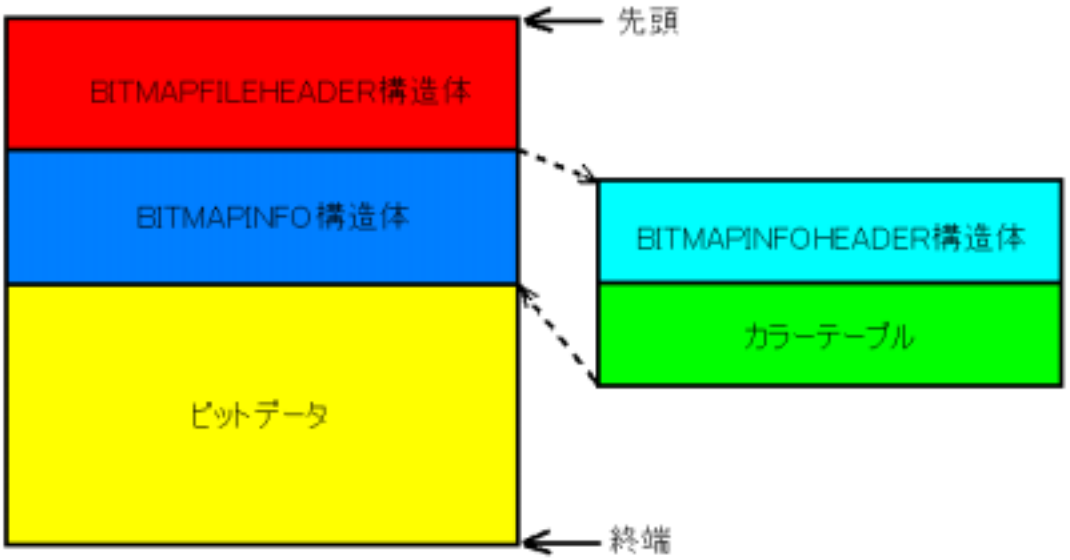

図 3.1: ビットマップの構造

図３．１のように、ビットマップファイルの構造は、以下のようになっている。

1.BITMAPFILEHEADER

2.BITMAPINFOHEADER

3. ビットマップのピクセル列

これらのヘッダ内で、最低限指定する必要があるのは BITMAPFILEHEADER の先頭２バイトとオフセット、ファイルサイズ、それに BITMAPINFOHEADER のビットマップのサイズを形式である。そのほかの情報は０でよい。

必要なメモリ容量は、ビットマップの横１ラインの長さを dwLength バイト（４バイト境界）、高さを dwHeight ピクセルとすると以下のようになる。

dwLength × dwHeight ＋ sizeof(BITMAPFILEHEASER) ＋ sizeof(BITMAPINFOHEADER)

3.1.2 ビットマップの各ファイル構造 (メンバ) の説明

● BITMAPFILEHEADER 構造体

ビットマップファイルの「ファイル」としての情報が入っている。ファイルの種類を見分ける。

まずデータの先頭から sizeof(BITMAPFILEHEADER) バイト分はこの構造体を指している。

bfType ファイルタイプを示す。そのファイルがビットマップであることを認識する。
”BM” という文字列を入れておく必要がある。通常は 0x4d42 が入る。

bfSize ビットマップデータ全体のサイズを入れておく。

bfOffBits ビットマップデータ列の開始位置（データの先頭からのオフセット）を設定。
ファイルの先頭からビットデータまでの距離をバイト単位で示す。これはビットマップファイルからダイレクトにビットデータを読み出したいときなどに使う。

● BITMAPINFOHEADER 構造体

ビットマップファイルのビットマップ画像としての情報が入っている。この構造体のメンバには、ビットマップの色数、幅、　高さなどの情報が入っている。

BITMAPFILEHEADER 構造体に続く、sizeof(BITMAPINFOHEADER) バイト分はこの構造体を指している。

biSize この構造体のサイズ（sizeof(BITMAPINFOHEADER)）を入れておく。

biWidth ビットマップの幅を設定。

biHeight ビットマップの高さを設定。

biPlanes 1 を入れておく。

biBitCount １ピクセルに必要なビット数を入れておく。

biCompression　圧縮タイプを入れておく。通常はＢＩ＿ＲＧＢ。

biSizeImage　ビットデータのサイズが入っているが、これはビットデータが圧縮されているときに必要なものであり、無圧縮。つまり、biCompression でＢＩ＿ＲＧＢが入っているときは、ビットデータのサイズは biWidth × biHeight × biBitcount でほぼ決まるので、ここでは０を指定する。

biXPelsPerMeter 水平成分の解像度を pixel/m で示しますが、普通は 0。

biYPelsPerMeter 垂直成分の解像度を pixel/m で示しますが、普通は 0。

biSizeImage イメージのサイズを入れておく。(が、特に必要ともされないようなので、０に。)

biClrUsed １ピクセルに必要なビット数を入れておく。(biBitCount で設定されているので、０に。)

●パレットデータ

BITMAPINFOHEADER 構造体に続いてパレットデータが続きく。フルカラービットマップの場合はパレットは存在しない。

パレットデータは RGB の各１バイトを持つ RGBQUAD 構造体の配列である。配列の要素数は、表現可能な色数分存在する。

●ビットマップデータ列

ビットマップのデータは、画像の左下座標から横方向に１ラインづつ格納されている。注意するべきなのは、　１ラインのバイト数はダブル・ワード（４バイト）の倍数でなければならない点である。フルカラー２４ビットでは、3バイトで　１ピクセルを表現する。各１バイトが RGB 値をそれぞれ保持する。（32 bit では、４バイトで１ピクセルを表現する。　この場合は１バイトがダミーになる。）フルカラー以外はパレットを所有している。ピクセルは色情報を収めたパレットの番号を　保持している。

２５６色 8ビットでは、１バイトで１ピクセルを表現。

１６色 4ビットでは、1/2 バイトで１ピクセルを表現。

２色 １ビットでは、1/8 バイトで１ピクセルを表現。

3.2　画像への情報の隠蔽・抽出

3.2.1　隠蔽・抽出方法

本研究では、ビットマップ画像に秘密情報としてある文章を隠蔽した。
方法は画素置換型電子透かしを用いた。秘密情報は例えば、人の名前、日記、ＩＤなど何でもよい。

3.2.2　隠蔽アルゴリズム

まず、秘密データをビットマップ画像に隠蔽するアルゴリズムのフローチャートを以下に示す。

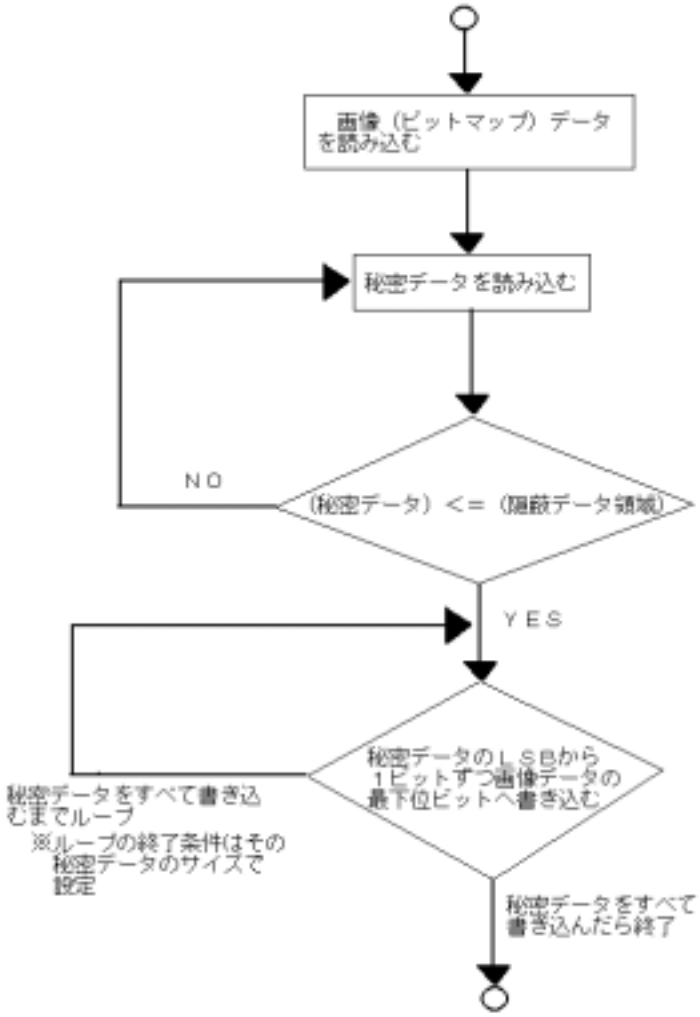

図 3.2: 隠蔽フローチャート

●隠蔽アルゴリズムの詳細

ビットマップ画像を読み込む際に、その画像のヘッダ情報（データのサイズ等）を取得する。

読み込んだ画像より、秘密データを隠蔽できる容量を求める。

（隠蔽容量)＝（画像データの幅）×（画像データの高さ）×　３　／　８

※データの隠蔽はＲＧＢ値の下位１ビットへ行なうことより

読み込んだ秘密データのサイズが隠蔽容量以下であるならば次へ。

秘密情報を画像データへの書き込む方法：

秘密データを８ビット単位で処理する（書き込む)。また、８ビット単位にしたデータを最下位１ビットずつ画像データに書き込む。８ビットに分割のしたデータを１ビットずつ処理するには右シフト代入演算子を用いる。最下位ビットのみを画像データに書き込んでいき、１ビット右シフトを７回繰り返す（８ビットに対応することになる)。つまり、最下位ビット以外にはすべてマスクをかけ、書き込んだら１ビット右シフトし、最下位ビット以外はマスクをかける、という手順である。画像データに書き込む場所はその画素データの最下位ビットである。秘密データを１ビット埋め込むごとに次の画像データの最下位ビットへ移る。

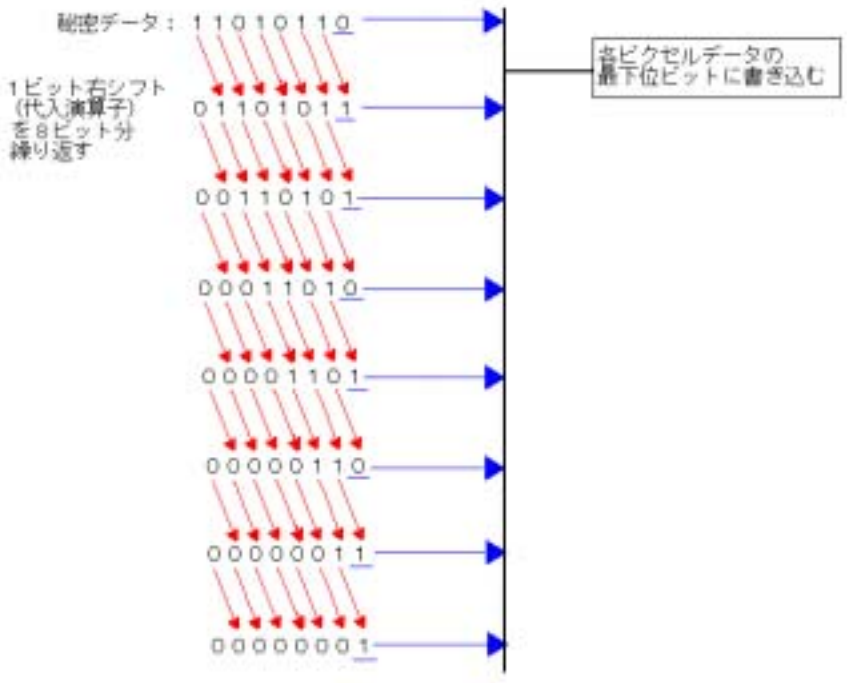

図 3.3: １ビット右シフトのうごき

以下に、本研究で用いた画像データへ秘密情報を隠蔽方法のしくみの例を挙げ、説明する。
各ピクセルデータは８ビットで構成されており、その各最下位ビットに秘密情報を書き込む。

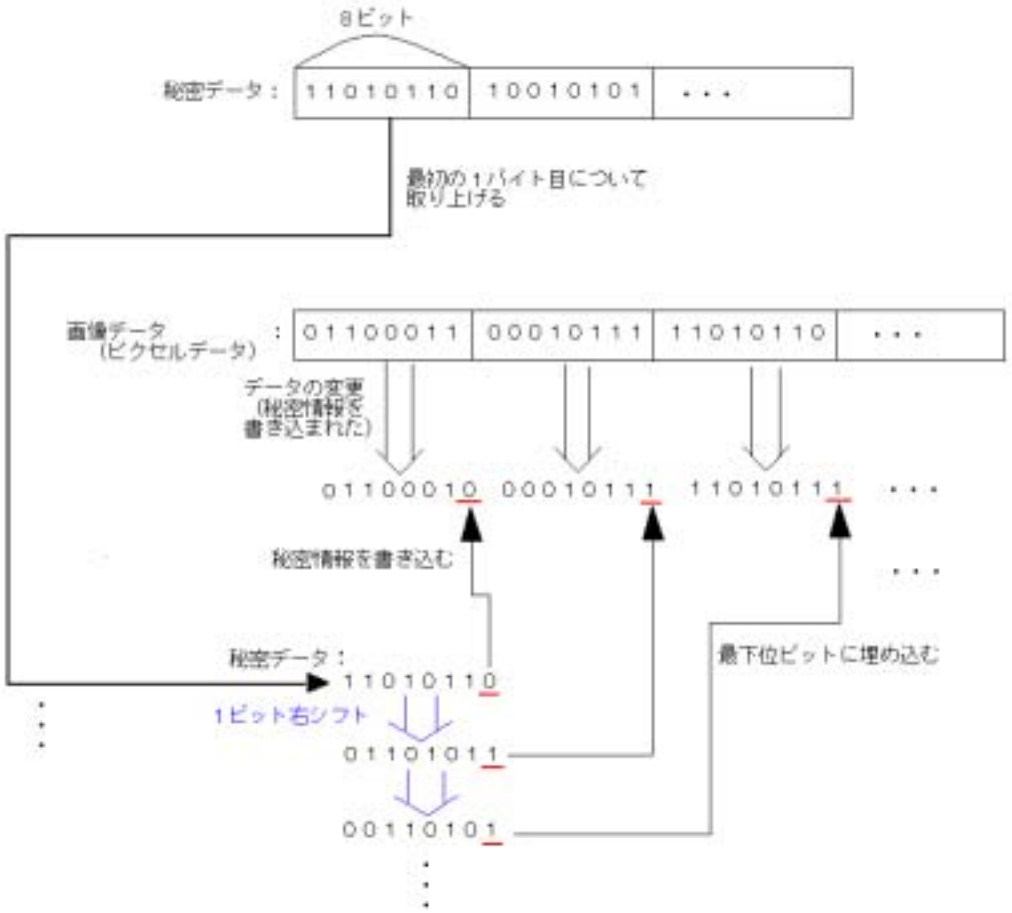

図 3.4: 画像データへ秘密情報を書き込む一例

図３．３では、まず、秘密情報の最下位ビット：０を書き込んでいる。ここで、そのピクセルデータの最下位ビット：１は先の秘密情報の最下位ビット（０）を書き込んでいるので、０に変更される。次に、秘密情報を１ビット右シフトさせる。　　つまり、

１１０１０１１０　⇒　０１１０１０１１

となる。このときの最下位ビットは１である。また、書き込むピクセルデータの位置は移り、その最下位ビット：１に先の秘密情報の最下位ビット（１）を書き込む。この場合、１に１を書き込むので、結果的にピクセルデータの変更はない。以上のように、この処理を秘密情報を全て画像データに書き込む（隠蔽）まで繰り返す。

3.2.3　抽出アルゴリズム

この方法は簡単にいえば、隠蔽の方法の逆を行なえばよい。

隠蔽の場合と処理の構造は同じであり、以下にその主な概要を示す。

画像データの最下位ビットを取りだし、８個まとめて（ＬＳＢから)、(抽出）データを構成する。

●具体的な抽出方法

以下の　、　を８回ループする。(１バイトずつまとめてデータを構成するため）

※i　＝　0〜7

（隠蔽後画像データ）＆１　…　ビットＡＮＤ演算子で最下位ビット以外にはマスクをかけ、最下位ビットのみを求める。

（抽出したデータ）|＝(１＜＜i)　…　左シフト演算子を用いて、下位ビットから順に秘密データ（抽出データ）を格納していく。

この詳細は下図に示す。

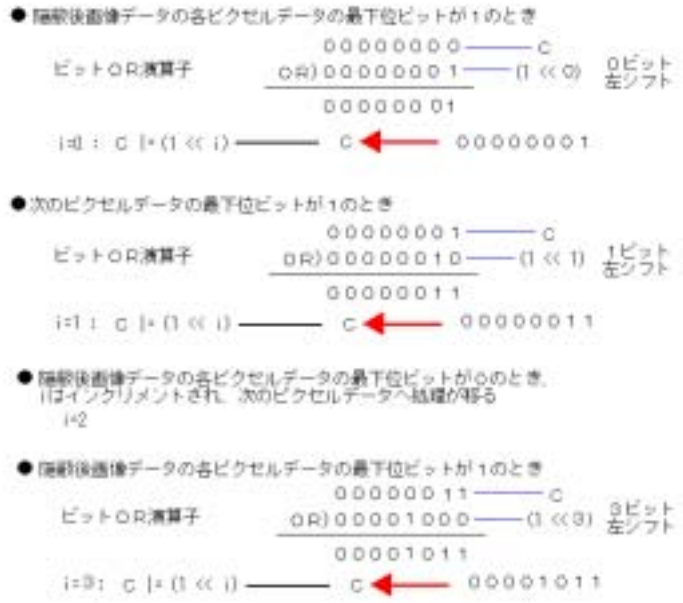

図 3.5: 抽出方法においての左シフトのうごき

●補足

この処理で、ピクセルデータの最下位ビットが１ならば１を、０ならば0が抽出データとして求められる。これを最下位ビットから順に格納していく。秘密データのＬＳＤから順に、画像データに書き込んだため、抽出する際にもＬＳＤから順に求める必要があり、この方法はそれを満たしている。

第４章　元画像と隠蔽後画像の比較・検証と結果

4.1　元画像と隠蔽後画像

4.1.1　比較・検証方法として

元画像と隠蔽後画像とを比較・検証するために、２つの画像の差分をとった。

お互いの画像データ（それぞれ対応するピクセルデータの各成分同士）を比較した。

もし、等しければそのピクセルデータの各成分を変更する。その結果、等しい部分の画像データが変更され、その部分の色が黒になる。（Ｒ，Ｇ，Ｂ成分を０に変更する。）

もし、等しくなければ、その部分の画像データを黒以外の色に変更する。こうすることで２つの画像データを比較した場合、等しい部分は黒に、そうでない部分は黒以外の色に変更されることになり、画像データが秘密情報の隠蔽によって変更されていることが確認できる。

4.1.2　差分について

元画像のピクセルデータと隠蔽後画像のピクセルデータのそれぞれの成分（Ｒ、Ｇ，Ｂ）を取りだし、同じ成分同士を比較する。取りだし方法はＲ成分が上位８ビットに格納されているので、１６ビット左シフトで下位８ビットに移して、他のビットはマスクする。ビットマップは上位ビットから順に、Ｒ，Ｇ，Ｂの順に８ビット単位で格納されている。

次に、その成分差をとる。差がない場合は、その成分を０にする。各成分（Ｒ，Ｇ，Ｂ）が０になるということは、そのお互いのピクセルデータが等しいこと同等である。また、Ｒ，Ｇ，Ｂが０になるということは黒を表す。

成分差がある場合は、その成分の値を２５５にする。こうすることで、黒以外の色データをもつピクセルデータをつくる。以上のことより、結果的に差分画像をみたとき、黒以外の色の部分に成分差があることを示すことになる。

成分による色の種類

	黒	青	水色	白	緑	黄	赤	紫
R成分	0	0	0	255	0	255	255	255
G成分	0	0	255	255	255	255	0	0
B成分	0	255	255	255	0	0	0	255

図 4.1: ２つの画像の差分をとる補足

4.2　結果

4.2.1　差分画像において

●まず、元画像と隠蔽後画像を目で見て比べてみた。
　もちろん、どちらの画像に秘密データが隠蔽されているのか、人の目で見ただけでは判断できない。
　以下にあるのが、実際に使用した元画像と隠蔽後画像である。

元画像

隠蔽後画像

図 4.2: 元画像と隠蔽後画像

●次に、その二つの画像から得られる差分画像を求めた。
　下の画像が差分画像である。

差分画像

図 4.3: 元画像と隠蔽後画像から得られた差分画像

4.2.2　考察

図 4.3 の差分画像の大部分の色は黒い。これは、それぞれ対応する元画像と隠蔽後画像の画像データが等しいこと、つまり、元画像の画像データが秘密情報の隠蔽によって、変更されていない部分があることを表している。

図 4.3 の差分画像の下の部分には黒以外の色が表示されている。これは、秘密情報の隠蔽によって、画像データが変更されていることを示している。だだし、元々画像データが黒である場合もある。画像の下の部分に、これが表れているのは、ビットマップではイメージを下から格納する方式が一般的であるためである。このことより、秘密情報が画像の下の部分に隠蔽されていることがわかる。

第5章　終論

画像に秘密情報を隠蔽することで、画像データが変更されていることが確認できた。ただし、本研究では Steganography の一方向からの検証にすぎず、そのほかの方法との比較・検証が必要である。なぜなら、本研究で利用した方法は Steganography の基本的なものであり、複数の方法を用いることで、その理解が深まることが明白だからである。

また、ビットマップだけでなく、JPEG、GIF 等の様々な画像に対象を広げ、それぞれを比較することで、用途が多様化し、利便性を高めることができる。

実際に、画像を人の目から見ての判断といった色々なパターンの実験を行ない、幅広い見地から Steganography を考察する必要がある。

引用文献・ホームページ・資料

松井　甲子雄　著　「電子透かしの基礎」　森北出版　235pp　1998

松井　甲子雄　著　「画像深層暗号」　森北出版　185pp　1993

小野　束　著　「電子透かしのコンテンツ」　オーム社　260pp　2001

http:/www.geocities.co.jp/SiliconValley-Oakland/3676/

謝辞

本研究において最後まで暖かく丁寧な指導、また助言をしてくださった田中章司郎先生に深く感謝の意をここに表します。

そして、田中研究室の藤井宏行さん、森本誠人さん、江野本隆行さん、、森下旭さん、平田純一さん、辰巳圭介さん、
貫目洋一さん、荒木俊太郎くん、駅範之くん、高木明くん、中村吉郎くん、森岡慶彦くんには研究に関して数々の助言を
いただきました。厚く御礼申し上げます。

さらに、本論文と本研究で作成したプログラムなどのすべての著作権は田中章司郎教授に譲渡いたします。