Стеганография и стегоанализ изображений

Внедрение скрытой информации в изображения может быть осуществлено путем манипуляции с младшими битами значений, кодирующих яркость пикселей. Младшие биты принято обозначать аббревиатурой LSB. Выделяются два метода внедрения: LSB-Replacement и LSB-Matching.

Первый из них (LSB-R) состоит в простой замене LSB на бит внедряемого сообщения, будь то 0 или 1. Покажем, как меняются значения пикселей при последовательном внедрении в них сообщения 1001:

51 80 121 62 → 51 80 120 63.

Второй метод (LSB-M, называемый также ±1-внедрение) немного сложнее. Если бит внедряемого сообщения равен LSB, то ничего не делается. В противном случае с одинаковой вероятностью производится прибавление либо вычитание единицы из значения пикселя (в исключительных случаях, когда это значение равно 0 либо 255, используется метод LSB-R). Пример внедрения сообщения 1001 выглядит так:

51 80 121 62 → 51 80 122 (либо 120) 63 (либо 61).

Как видим, при использовании и того и другого метода биты внедряемого сообщения оказываются в LSB (т.е. метод извлечения информации один и тот же). Если внедрение производится во все пиксели всех цветовых составляющих (RGB), то достигается максимально возможный рейт внедрения R_{max}=3bpp (bit per pixel). Как правило, для повышения скрытности используют более низкие рейты, например 0,25R_{max}, т.е. для внедрения используется лишь четверть пикселей.

Внедряемая информация искажает исходное изображение. Это искажение обычно незаметно для глаза, но может быть обнаружено через изменение степени сжатия. Сжатие изображения можно проводить с помощью стандартных архиваторов или специализированных программ. Обычно после внедрения скрытой информации сжатие ухудшается, т.к. вносимые искажения портят естественные статистические зависимости в матрице изображения. Это может быть использовано для выявления факта наличия скрытых данных.

Если мы внедряем информацию с рейтом ниже R_{max}, то можно минимизировать искажения, применяя коды, исправляющие ошибки. Рассмотрим, например, (15, 11)-код Хемминга, задаваемый следующей проверочной матрицей (она получается путем записи по столбцам всех чисел от 1 до15 в двоичном представлении):

$$H=\begin{pmatrix}1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\\0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\end{pmatrix}.$$

С помощью этого кода можно в группу из 15 значений яркости пикселей вставить 4 бита информации, исказив не более одного значения (при обычном методе внедрения могут исказиться 4 значения яркости). Делается это так. Обозначим через - c = (c_1, c_2,...,c_{15}) младшие биты в группе из 15 значений яркостей пикселей; через m = (m_1,...,m_4) – скрываемое сообщение. Вычислим вектор

$$s^T=(Hc^T)\oplus m^T$$

(знак транспонирования означает, что вектор при умножении на матрицу рассматривается как столбец; операции сложения и умножения производятся по модулю 2). Трактуя полученный вектор как двоичное представление числа, получим индекс:

$$i=8s_4+4s_3+2s_2+s_1.$$

Теперь, чтобы внедрить сообщение m, инвертируем i-й бит в векторе c (если i = 0, все биты остаются без изменения). Обозначим модифицированный таким образом вектор через $\tilde{c}$. Восстановить внедренное сообщение можно по формуле

$$m^T=H\tilde{c}^T$$

Для рассмотренного метода рейт внедрения равен:

$$\frac{4}{15}\cdot R_{\max}\approx 0.27R_{\max}$$

Другие коды Хемминга ((3, 1), (7, 4), (31, 26) и т.д.) могут быть использованы по аналогии.

Замечание. Исходное предназначение кодов Хемминга – исправление одиночной ошибки в блоке. Вектор s называется синдромом. Он показывает место ошибки. По сути дела, мы внедряем сообщение путем воспроизводства ошибки в определенной позиции.

Задание. Написать программы внедрения (и извлечения) скрытой информации в BMP-файлы, используя методы LSB-R и LSB-M, с задаваемым рейтом внедрения. Для заданных файлов (PU24-*.bmp) исследовать зависимости между рейтом внедрения и степенью сжатия, изменение степени сжатия при повторном внедрении при использовании различных ***архиваторов*** или ***графических конверторов***. Затем написать программу внедрения на основе кода Хемминга. Провести для нее аналогичные исследования, сопоставить результаты с обычными методами при (почти) том же рейте внедрения. Сделать выводы о стеганографической стойкости исследованных методов.

BMP-файлы и их загрузка в память

Для простоты используйте цветные изображения с глубиной цвета 24 бита.

```
char *image;
int N;
struct stat st;
fp = fopen ("test.bmp", "rb+");
fstat (fileno(fp), &st);
N = st.st_size;
image = new char [N];
fread(image, 1, N, fp);

// байты изображения идут с image[54] по image[N-1]
// в них можно внедрять скрытые данные, изменяя младшие биты


rewind(fp);
fwrite(image, 1, N, fp);
fclose(fp);
```