Университет ИТМО

Цифровая обработка сигналов
Лабораторная работа №7

Выполнила: Калугина Марина

Группа: P3402

г. Санкт-Петербург

2020 г.

Задание

Задание №1.

1. Используя соответствующие функции пакета Matlab, напишите программу, реализующую алгоритм сокрытия данных в младшем разрядном срезе изображения.
2. Проведите эксперименты по сокрытию данных и их извлечению для различных изображений и типов скрываемых данных (бинарных изображений, данных типа .txt, .dat).
3. Определите количественные характеристики, определяющие соответствие между пустым и заполненным стегоконтйнером.

Задание №2

1. Самостоятельно реализуйте алгоритм сокрытия данных (текста) в псевдо-белых и псевдо-черных пикселах изображения и с использованием средств пакета Matlab напишите программу, реализующую данный метод.
2. Выполните п.п. 2 и 3 задания №1.
3. Сравните емкость стегоконтейнеров для обоих методов.
4. Самостоятельно провести подобные манипуляции с несколькими различными изображениями. Привести вид исходного стегоконтейнера и гистограмму яркости пикселов подлежащего сокрытию бинарного изображения, стегоконтейнера после помещения в него скраваемой информации (с гистограммой) и вид восстановленного бинарного изображения (для каждого из изображений). Проанализировать недостатки и достоинства методов и вариант модификации для помещения скрываемой текстовой (символьной) информации.

Результаты

Сокрытие в младшем разрядном срезе

```
% Считываем изображение-контейнер
input = imread('putin_512.png');
input = input(1:512, 1:512);
% Считываем сообщение
message=fileread('lorem.txt');


% Считаем размер сообщения в битах
len = length(message) * 8;

% Получаем ASCII символы сообщения
ascii_value = uint8(message);

% Конвертируем в биты
bin_message = transpose(dec2bin(ascii_value, 8));

% Преобразуем каждый символ в отдельную битовую строку
bin_message = bin_message(:);

% Длина бинарного сообщения
N = length(bin_message);

% Конвертируем в строковый массив в битовый
bin_num_message=str2num(bin_message);

% Инициализируем результирующее изображение
output = input;

% Считываем размер изображения
height = size(input, 1);
width = size(input, 2);

% Счётчик встроенных битов
embed_counter = 1;

% Обходим изображение
for i = 1 : height
    for j = 1 : width


        if(embed_counter <= len)

            % Получаем наименее значимый бит пикселя
            LSB = mod(double(input(i, j)), 2);
```

```
            % Проверяем на то, совпадают ли биты или их нужно
менять
            temp = double(xor(LSB,
bin_num_message(embed_counter)));

            % Изменяем бит результата to input + temp
            output(i, j) = input(i, j)+temp;


            embed_counter = embed_counter+1;
        end

    end
end

display(embed_counter);
figure, histogram(input); title("Заполненый контейнер.
Гистограмма");
figure, histogram(output); title("Заполненый контейнер.
Гистограмма");
figure, imshow(input); title("Пустой контейнер");
figure, imshow(output); title("Заполненый контейнер");


% Размер в битах
message_length = embed_counter-1;


counter = 1;

for i = 1 : height
    for j = 1 : width


        if (counter <= message_length)

            % Считываем наименее значимый бит
            extracted_bits(counter, 1) = mod(double(output(i, j)),
2);

            counter = counter + 1;
        end
    end
end

% Значения для преобразования в ASCII
binValues = [ 128  64  32  16  8  4  2  1 ];
```

```
% Преобразовываем все биты в таблицу с восмью колонками
% Каждая строка - биты символа сообщения
binMatrix = reshape(extracted_bits, 8,(message_length/8));

% Преобразуем в ASCII
textString = char(binValues*binMatrix);
disp(textString);
```

Пустой контейнер

Заполненый контейнер

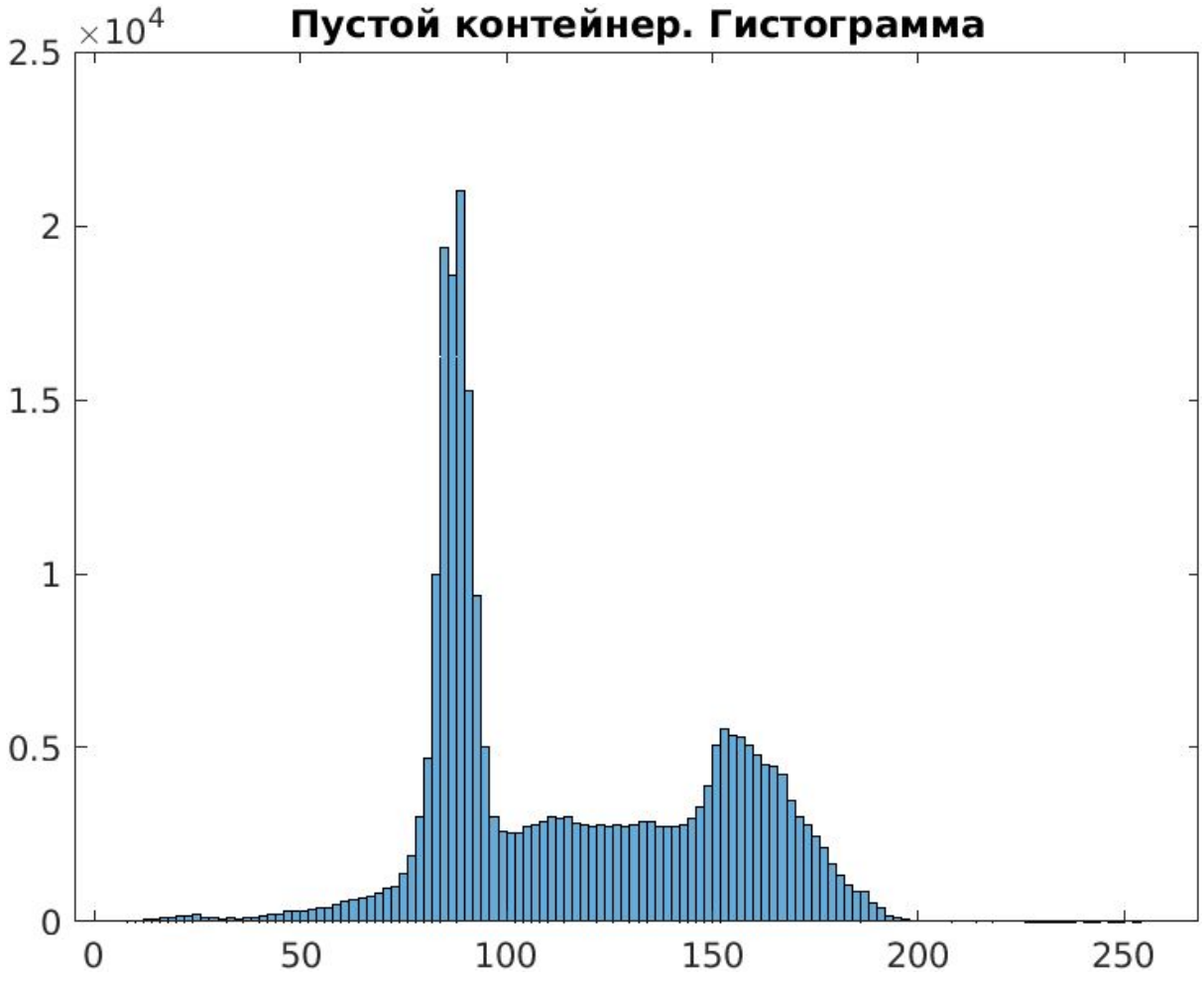

Пустой контейнер. Гистограмма
×10^4
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
50
100
150
200
250

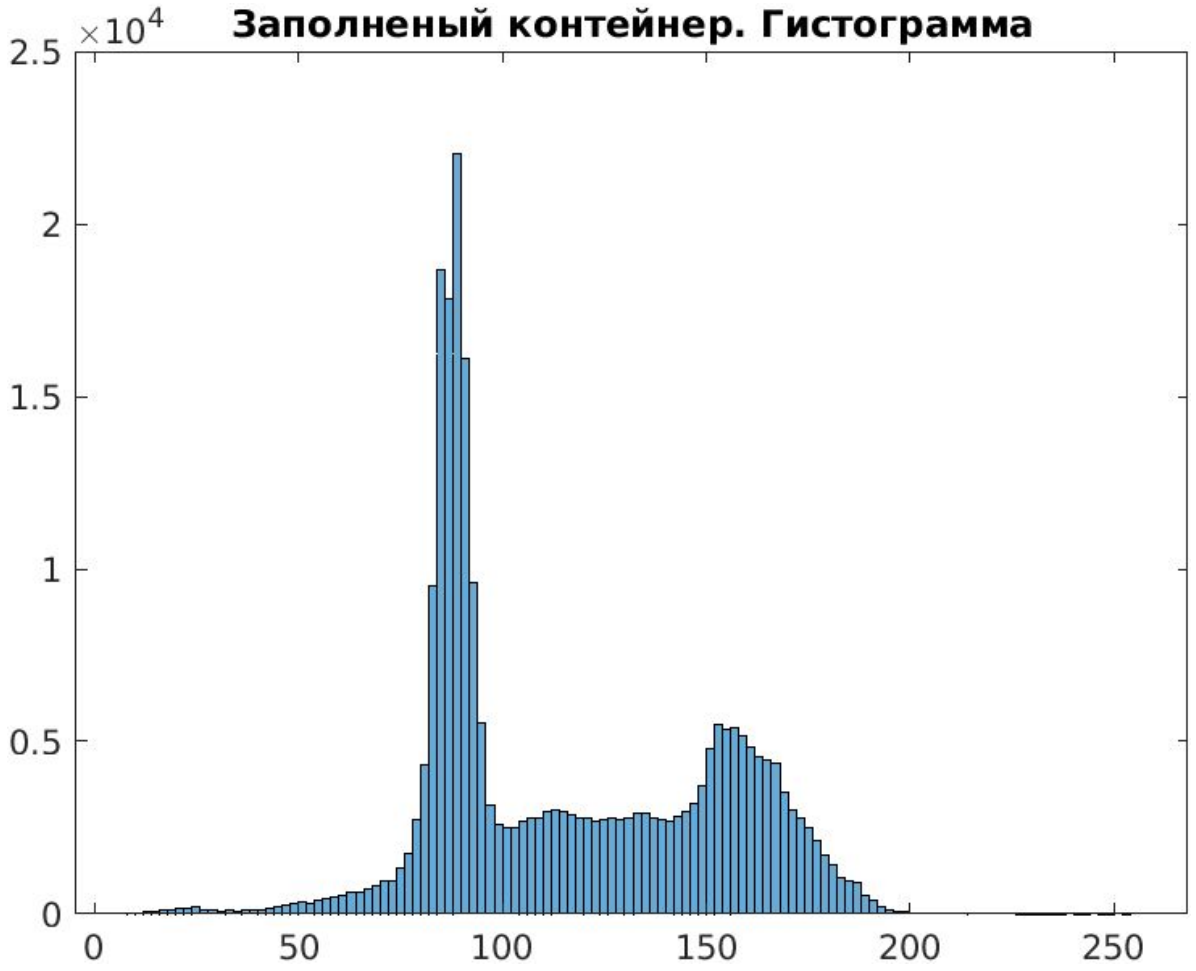

Заполненый контейнер. Гистограмма
×10^4
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
50
100
150
200
250

Пустой контейнер

Заполненый контейнер

Скрываемое изображение


Пустой контейнер. Гистограмма
×10^4
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250



Восстановленное изображение

Сокрытие в псевдо-белых и псевдо-чёрных пикселах

```
% Считываем изображение-контейнер
input = imread('putin_512.png');
input = input(1:512, 1:512);
% Считываем сообщение
message=fileread('lorem.txt');

% Считаем размер сообщения в битах
len = length(message) * 8;

% Получаем ASCII символы сообщения
% ascii_value = uint8(message);

% Конвертируем в биты
bin_message = transpose(dec2bin(ascii_value, 8));

% Преобразуем каждый символ в отдельную битовую строку
bin_message = bin_message(:);

% Длина бинарного сообщения
N = length(bin_message);

% Конвертируем в строковый массив в битовый
bin_num_message=str2num(bin_message);

% Инициализируем результирующее изображение
output = input;

% Считываем размер изображения
height = size(input, 1);
width = size(input, 2);

% Счётчик встроенных битов
embed_counter = 1;

% Обходим изображение
for i = 1 : height
    for j = 1 : width


        if(embed_counter <= len)

            if(((input(i, j)>=0) & (input(i, j)<=15)) | ((input(i,
j)>=240) & (input(i, j)<=255)))
                output(i, j) = bitset(  output(i, j), 1,
bin_num_message(embed_counter),'uint8');
                embed_counter = embed_counter+1;
```

```
                output(i, j) = bitset(  output(i, j), 2,
bin_num_message(embed_counter),'uint8');
                embed_counter = embed_counter+1;
                output(i, j) = bitset(  output(i, j), 3,
bin_num_message(embed_counter),'uint8');
                embed_counter = embed_counter+1;
                output(i, j) = bitset(  output(i, j), 4,
bin_num_message(embed_counter),'uint8');
                embed_counter = embed_counter+1;
            end
        end

    end
end
figure, histogram(input)
figure, histogram(output)
figure, imshow(input)
figure, imshow(output)


% Максимальный размер текста в сообщении
chars = 32768;

% Размер в битах
message_length = embed_counter;

message_length=fix(message_length/8)*8;

counter = 1;
extracted_bits=[embed_counter];
for i = 1 : height
    for j = 1 : width

        if(((output(i, j)>=0) && (output(i, j)<=15)) ||
((output(i, j)>=240) && (output(i, j)<=255)))
            if (counter <= message_length)


                extracted_bits(counter, 1) = bitget(output(i,
j),1);
                counter = counter + 1;


                extracted_bits(counter, 1) = bitget(output(i,
j),2);
                counter = counter + 1;
```

```
                extracted_bits(counter, 1) = bitget(output(i,
j),3);
                 counter = counter + 1;


                extracted_bits(counter, 1) = bitget(output(i,
j),4);
                counter = counter + 1;

            end
        end
    end
end

length(extracted_bits)
% Значения для преобразования в ASCII
binValues = [ 128 64 32 16 8 4 2 1 ];

% Преобразовываем все биты в таблицу с восмью колонками
% Каждая строка - биты символа сообщения
binMatrix = reshape(extracted_bits, 8, (message_length/8));

% Преобразуем в ASCII
textString = char(binValues*binMatrix);
disp(textString);
```

Пустой контейнер

Заполненый контейнер

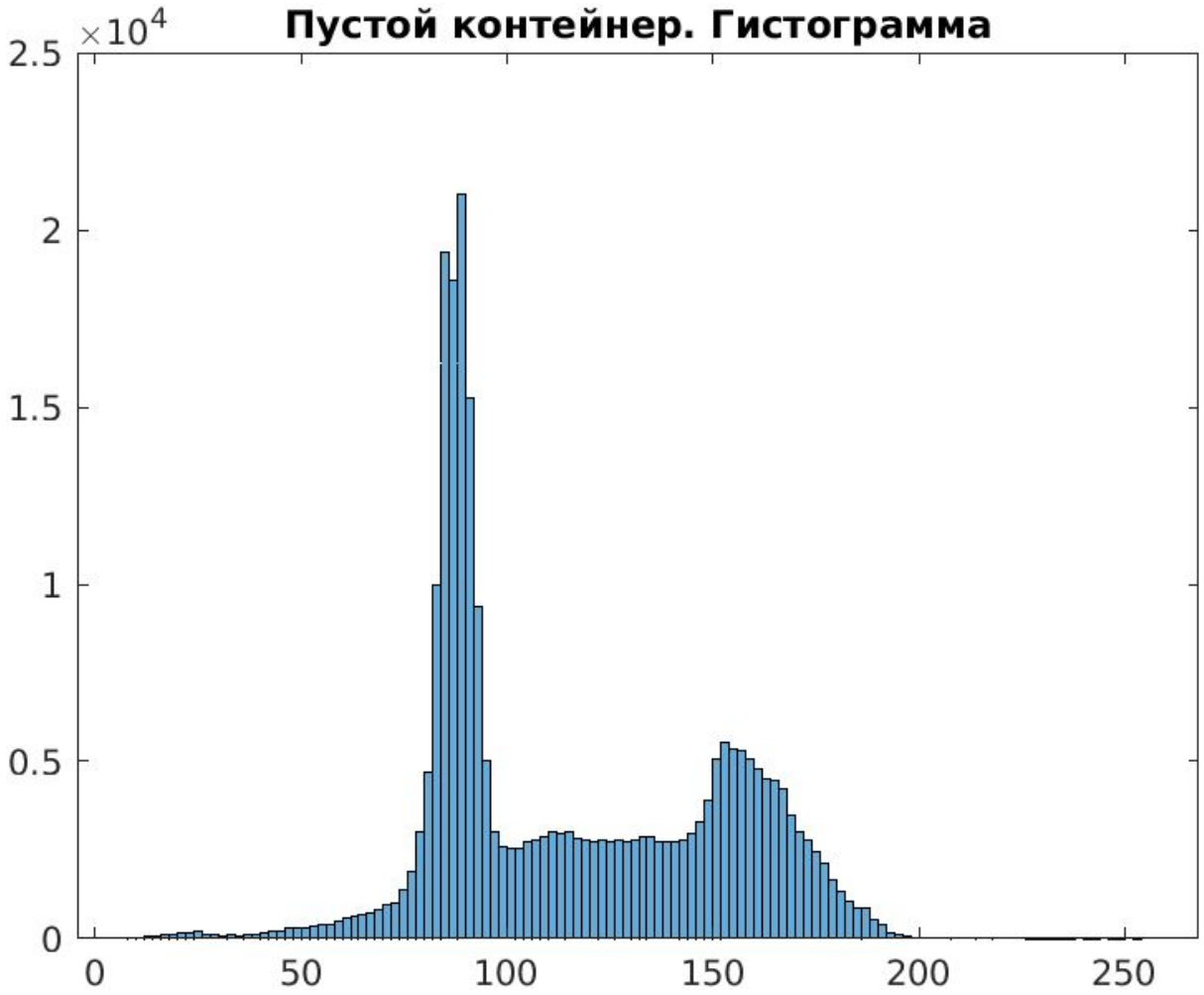

Пустой контейнер. Гистограмма
×10^4
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
50
100
150
200
250

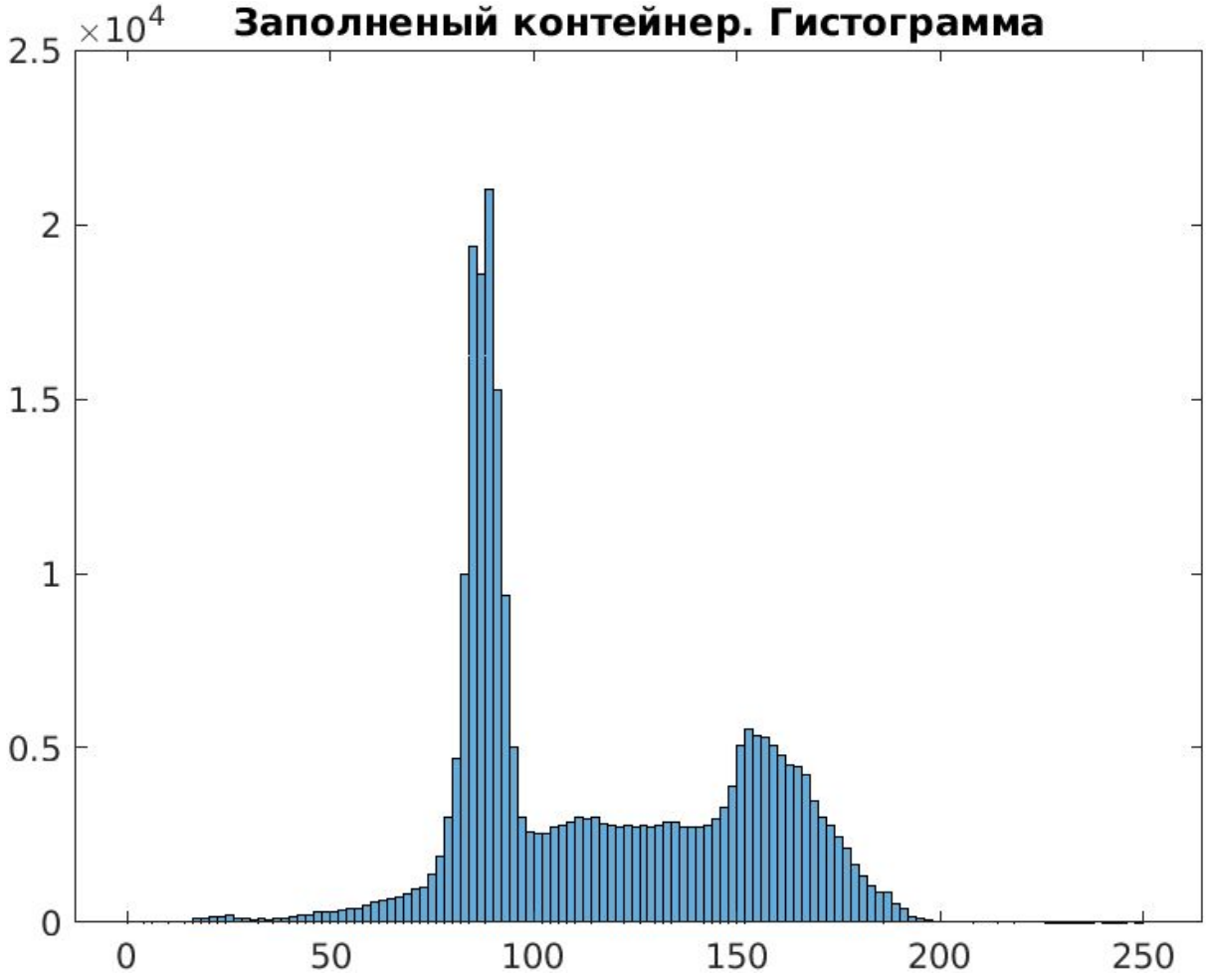

Заполненый контейнер. Гистограмма
15000
10000
5000
0
0
50
100
150
200
250

Пустой контейнер

Заполненый контейнер

Скрываемое изображение


Пустой контейнер. Гистограмма
×10^4
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250



Восстановленное изображение

Вывод

В ходе данной лабораторной работы были реализованы алгоритмы сокрытия данных в младшем разрядном срезе изображения и в псевдо-белых и псевдо-чёрных пикселах. Первый подход позволяет в любом изображении скрывать количество бит, равному произведению его измерений. Второй подход, потенциально, позволяет скрывать большие объёмы данных, поскольку в одном байте можно скрыть 4 бита информации. Однако данный подход требует изображения со значительным

количество псевдо-белых и псевдо-чёрных пикселов, в противном случае объём скрываемой информации будет незначительным.