

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____
ПРЕПОДАВАТЕЛЬ

старший преподаватель _____
должность, уч. степень, звание _____
подпись, дата _____
Т.А. Сутина
ициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5

Сжатие изображения алгоритмом JPEG 2000

по курсу: Техника аудиовизуальных средств информации

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № _____ 4329 _____
подпись, дата _____
Д.С. Шаповалова
ициалы, фамилия

Санкт-Петербург 2025

1. Цель работы:

Получить теоретические знания по особенностям алгоритма сжатия и на практике выполнить все этапы сжатия изображения.

2. Задание:

Для изображения размером 16x16 пикселей в палитре RGB выполнить:

1. Расчет объема входного файла.
2. Яркостной сдвиг изображения.
3. Преобразования цветового пространства
4. Дискретное вейвлет преобразование
5. Квантование
6. Арифметическое кодирование
7. Коэффициент сжатия

Для всех этапов, предоставить исходные матрицы и полученные результаты.

Сделать соответствующие выводы.

Исходное изображение:

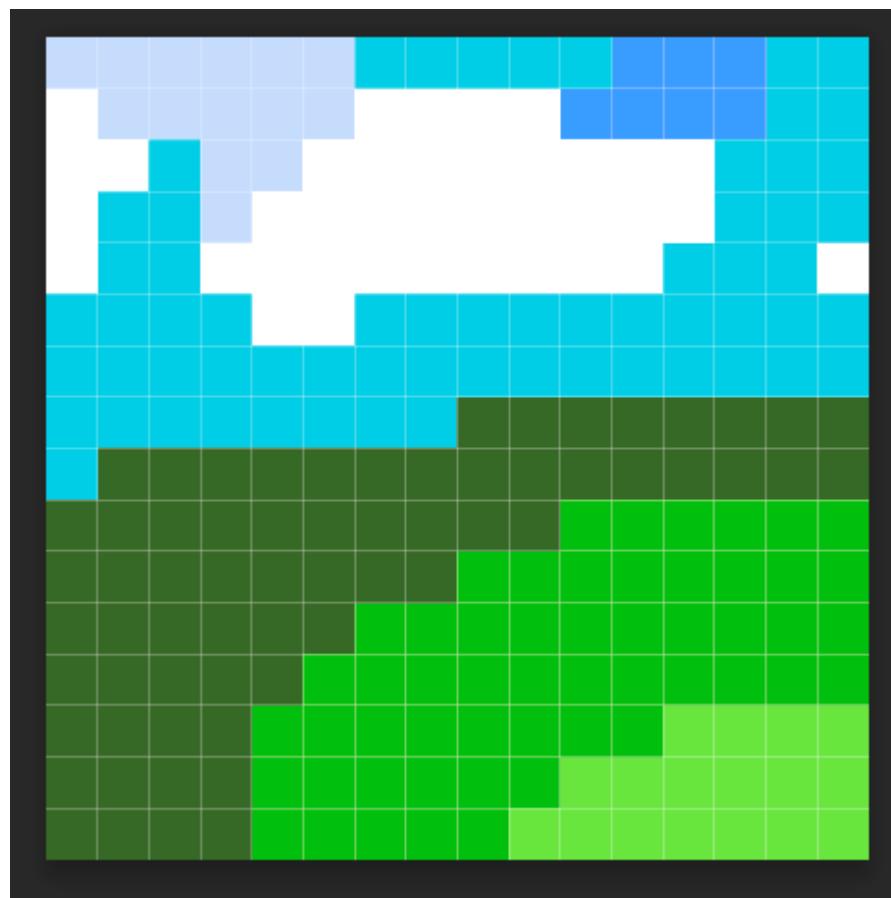


Рисунок 1 – Исходное изображение

3. Ход работы:

Рассчитаем объём исходного изображения: $V = 16 * 16 * 8 * 3 = 6144$ бит

Сдвиг по яркости и преобразование цветового пространства RGB в YUV

Необходимо каждую компоненту RGB изображения подвергнуть перерасчёту по формуле:

$$I'(x, y) = I(x, y) + 2^{ST-1}, \quad (1)$$

где ST – количество бит, которое используется для представления коэффициента, I – изначальное значение яркости компонента, I' – полученное значение после сдвига.

Значения после сдвига представлены на рисунке 2:

RGB_matrix																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
1 [75,91,124]	[75,91,124]	[75,91,124]	[75,91,124]	[75,91,124]	[75,91,124]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-29,27,127]	[-29,27,127]	[-29,27,127]	[-33,77,100]	[-33,77,100]	
2 [127,127,127]	[75,91,124]	[75,91,124]	[75,91,124]	[75,91,124]	[75,91,124]	[127,127,12]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[-29,27,127]	[-29,27,127]	[-29,27,127]	[-33,77,100]	[-33,77,100]	
3 [127,127,127]	[127,127,127]	[-33,77,100]	[75,91,124]	[127,127,127]	[127,127,127]	[127,127,12]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	
4 [127,127,127]	[-33,77,100]	[-33,77,100]	[75,91,124]	[127,127,127]	[127,127,127]	[127,127,12]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	
5 [127,127,127]	[-33,77,100]	[-33,77,100]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,12]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	[127,127,127]	
6 [-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[127,127,127]	[127,127,127]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	
7 [-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	
8 [-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[-33,77,100]	
9 [-33,77,100]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	
10 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	
11 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	
12 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	
13 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	
14 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	
15 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	
16 [-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-22,62,-80]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	[-25,101,-48]	

Рисунок 2 – Матрица RGB после сдвига

Преобразование из цветового пространства RGB в пространство YUV

Осуществляется с помощью матричных преобразований по формуле:

$$\begin{vmatrix} Y \\ U \\ V \end{vmatrix} = \begin{vmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.4187 & -0.0813 \\ 0.1687 & -0.3313 & 0.5 \end{vmatrix} * \begin{vmatrix} R \\ G \\ B \end{vmatrix} + \begin{vmatrix} 0 \\ 128 \\ 128 \end{vmatrix}, \quad (2)$$

Полученные матрицы для яркостной и цветоразностных компонентов представлены на рисунках 3-5:

Y																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
1 90	90	90	90	90	90	47	47	47	47	47	22	22	22	47	47	
2 127	90	90	90	90	90	127	127	127	127	127	22	22	22	47	47	
3 127	127	47	90	90	127	127	127	127	127	127	127	127	127	47	47	
4 127	47	47	90	127	127	127	127	127	127	127	127	127	127	47	47	
5 127	47	47	47	127	127	127	127	127	127	127	127	127	127	47	127	
6 47	47	47	47	127	127	47	47	47	47	47	47	47	47	47	47	
7 47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47	
8 47	47	47	47	47	47	47	47	47	-39	-39	-39	-39	-39	-39	-39	
9 47	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	
10 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
11 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
12 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
13 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
14 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
15 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	
16 -39	-39	-39	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	

Рисунок 3 – Яркостная компонента Y

	Y	X	U	V	
16x16 double					
1	117	117	3	4	5
2	128	117	117	117	6
3	128	128	71	117	7
4	128	71	71	117	8
5	128	71	71	128	9
6	71	71	71	128	10
7	71	71	71	71	11
8	71	71	71	71	12
9	71	118	118	118	13
10	118	118	118	118	14
11	118	118	118	118	15
12	118	118	118	118	16
13	118	118	118	98	
14	118	118	118	98	
15	118	118	118	98	
16	118	118	118	98	

Рисунок 4 – Цветоразностная компонента U

	Y	X	U	V	
16x16 double					
1	173	173	3	4	5
2	171	173	173	173	6
3	171	171	147	173	7
4	171	147	147	173	8
5	171	147	147	171	9
6	147	147	147	171	10
7	147	147	147	171	11
8	147	147	147	147	12
9	147	86	86	86	13
10	86	86	86	86	14
11	86	86	86	86	15
12	86	86	86	86	16
13	86	86	86	64	
14	86	86	86	64	
15	86	86	86	64	
16	86	86	86	64	

Рисунок 5 – Цветоразностная компонента V

Дискретное вейвлет преобразование

Преобразование яркостной компоненты Y осуществляется с помощью вейвлета Хаара в 2 раунда преобразования.

Полученная матрица аппроксимации LL2 представлена на рисунке 6.

LL2			
4x4 double			
1	2	3	4
1	364.7500	421.7500	369.2500
2	228.0000	308.0000	182.0000
3	-134.5000	-126.0000	-6
4	-156.0000	69.0000	134.0000
			204.0000

Рисунок 6 – Матрица аппроксимации

Матрица всех составляющих (LL, HL, LH, HH) в диапазоне 0-255 после 2 раунда вейвлет преобразования представлена на рисунке 7.

	1	2	3	4	5	6	7	8
1	221	255	223	121	22	105	214	100
2	137	186	108	72	62	185	255	163
3	79	74	1	12	34	47	139	93
4	93	39	79	122	1	24	77	62
5	208	28	255	66	115	49	231	255
6	1	174	1	87	1	157	1	79
7	94	131	131	1	85	118	118	1
8	1	66	131	1	1	59	118	1

Рисунок 7 – Матрица всех составляющих

Квантование

Квантование частотных коэффициентов в матрице LL2 происходит за счёт квантования с мёртвой зоной с шагом $\Delta = 10$. Значения меньше Δ по модулю приравниваются к 0. Больше – принимают значения уровня, в диапазон которого попали.

Матрица проквантованных коэффициентов представлена на рисунке 8:

	1	2	3	4
1	36	42	36	20
2	22	30	18	12
3	-13	-12	0	2
4	-15	6	13	20

Рисунок 8 – Матрица LL2 после квантования

Арифметическое кодирование

Для арифметического кодирования квантованной матрицы LL2 рассчитывается частота каждого коэффициента, распределяется по прямой, проводится непосредственное арифметическое кодирование.

Этапы уточнения диапазона и итоговый диапазон представлены на рисунке 9:

Коэф.	Вероятность	Нижн.гран.	Верхн.гран.	Расчит.н.гр.	Расчит.верх.гр.
36	0.1250	0.8125	0.9375	0.812500	0.937500
22	0.0625	0.6875	0.7500	0.898438	0.906250
-13	0.0625	0.0625	0.1250	0.898926	0.899414
-15	0.0625	0.0000	0.0625	0.898926	0.898956
42	0.0625	0.9375	1.0000	0.898954	0.898956
30	0.0625	0.7500	0.8125	0.898956	0.898956
-12	0.0625	0.1250	0.1875	0.898956	0.898956
6	0.0625	0.3125	0.3750	0.898956	0.898956
36	0.1250	0.8125	0.9375	0.898956	0.898956
18	0.0625	0.5000	0.5625	0.898956	0.898956
0	0.0625	0.1875	0.2500	0.898956	0.898956
13	0.0625	0.4375	0.5000	0.898956	0.898956
20	0.1250	0.5625	0.6875	0.898956	0.898956
12	0.0625	0.3750	0.4375	0.898956	0.898956
2	0.0625	0.2500	0.3125	0.898956	0.898956
20	0.1250	0.5625	0.6875	0.898956	0.898956

Закодированное значение: 0.898955839629

Итоговый диапазон: [0.898955839629 , 0.898955839629]

Рисунок 9 – Результат арифметического кодирования

Расчёт коэффициента сжатия

Расчёт размера преобразованного файла проводился по формуле 3:

$$V = -n * \sum_{i=1}^m p_i \log_2 p_i, \quad (3)$$

где p_i – вероятность появления i -го символа, n – число символов, m – мощность алфавита.

$$V = -16 * (4 * 0,1250 * \log(0,1250) + 12 * 0,0625 * \log(0,0625)) = 72 \text{ бита}$$

Столько бит требуется для точного восстановления всех коэффициентов, что определяется шириной итогового интервала кодирования.

$$\text{Коэффициент сжатия } K = \frac{6144}{72} \approx 85,333$$

ВЫВОД

В ходе лабораторной работы были выполнены ключевые этапы сжатия изображения по алгоритму JPEG 2000: сдвиг яркости, преобразование цветового пространства RGB в YUV, двухуровневое вейвлет-преобразование Хаара, квантование с мёртвой зоной и арифметическое кодирование.

Было показано, что за счёт вейвлет-анализа и эффективного кодирования объём данных значительно сокращается – после арифметического кодирования размер преобразованного файла составил 72 бита, что соответствует коэффициенту сжатия 85,333.

Это подтверждает высокую эффективность JPEG 2000 при сохранении визуального качества изображения.

ПРИЛОЖЕНИЕ А

Листинг Программы

```
img = imread("img5.bmp");
img = double(img);
% Размер изображения
[m, n, ~] = size(img);

%% Шаг 1: Сдвиг по яркости
ST = [8, 8, 8];
shift_vals = 2.^(ST - 1); % = 128

% Сдвиг к каждому из 3 каналов
img_shifted = zeros(size(img));

% R channel
img_shifted(:,:1) = img(:,:1) - shift_vals(1);
% G channel
img_shifted(:,:2) = img(:,:2) - shift_vals(2);
% B channel
img_shifted(:,:3) = img(:,:3) - shift_vals(3);

% Матрица RGB компонентов после сдвига
RGB_matrix = cell(m, n);

for i = 1:m
    for j = 1:n
        % вектор [R G B]
        RGB_matrix{i,j} = squeeze(img_shifted(i,j,:))';
    end
end

%% Шаг 2: Преобразование цветового пространства
% Матрица преобразования
T = [ 0.299  0.587  0.114;
      0.5     -0.4187 -0.0813;
      0.1687 -0.3313  0.5 ];
offset = [0; 128; 128];

Y = zeros(m,n);
U = zeros(m,n);
V = zeros(m,n);

for i = 1:m
    for j = 1:n
        RGB = RGB_matrix{i,j}';
        % Преобразование
        YCbCr = T * RGB + offset;
        Y(i,j) = round(YCbCr(1));
        U(i,j) = round(YCbCr(2));
        V(i,j) = round(YCbCr(3));
    end
end

%% Шаг 3: Дискретное wavelet преобразование
[C,S] = wavedec2(Y, 2, 'haar');

m1 = S(1,1); n1 = S(1,2);

% Аппроксимация 2 уровня
LL2 = reshape(C(1:m1*n1), m1, n1);
```

```

LL1 = appcoef2(C, S, 'haar', 1);
m2 = S(2,1); n2 = S(2,2);

% Детали 2 уровня
HL2 = reshape(C(m1*n1+1:m1*n1+m2*n2), m2, n2);
LH2 = reshape(C(m1*n1+m2*n2+1:m1*n1+2*m2*n2), m2, n2);
HH2 = reshape(C(m1*n1+2*m2*n2+1:m1*n1+3*m2*n2), m2, n2);

C2=[wcodemat(LL2,255), wcodemat(HL2,255); wcodemat(LH2,255), wcodemat(HH2,255)];

%% Шаг 4: Квантование
Q = 10; % шаг квантования

% Функция для квантования с мёртвой зоной
deadzone_quant = @(x) sign(x) .* floor((abs(x))/Q) .* (abs(x) >= Q);

LL2_q = deadzone_quant(LL2);
HL2_q = deadzone_quant(HL2);
LH2_q = deadzone_quant(LH2);
HH2_q = deadzone_quant(HH2);

%% Шаг 5: Арифметическое кодирование
Xq = LL2_q(:);
code = arithmetic_encode_verbose(Xq);
function code = arithmetic_encode_verbose(symbols)

% Алфавит
alphabet = unique(symbols);

% Частоты
counts = histc(symbols, alphabet);
prob = counts / sum(counts);
prob = prob(:);

% Кумулятивное распределение
cum_prob = [0; cumsum(prob)];

n = length(symbols);
low = 0;
high = 1;

fprintf('Коэф.\tВероятность\tНижн.гран.\tВерхн.гран.\tРасчит.н.гр.\tРасчит.верх.гр.\n');
');

% Кодирование
for k = 1:n
    sym = symbols(k);
    idx = find(alphabet == sym);

    % Границы символа
    c_low = cum_prob(idx);
    c_high = cum_prob(idx + 1);

    % Новые границы диапазона
    range = high - low;

    new_low = low + range * c_low;
    new_high = low + range * c_high;

    fprintf('%d\t%.4f\t%.4f\t%.4f\t%.6f\t%.6f\n', ...

```

```
    sym, prob(idx), c_low, c_high, new_low, new_high);  
  
    low = new_low;  
    high = new_high;  
end  
code = (low + high)/2;  
fprintf('\nЗакодированное значение: %.12f\n', code);  
fprintf('Итоговый диапазон: [% .12f , %.12f]\n', low, high);  
end
```