ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____

ПРЕПОДАВАТЕЛЬ

старший преподаватель		Т.А. Суетина
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5

Сжатие изображения алгоритмом JPEG 2000

по курсу: Техника аудиовизуальных средств информации

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4329		Д.С. Шаповалова
	подпись, дата	инициалы, 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:

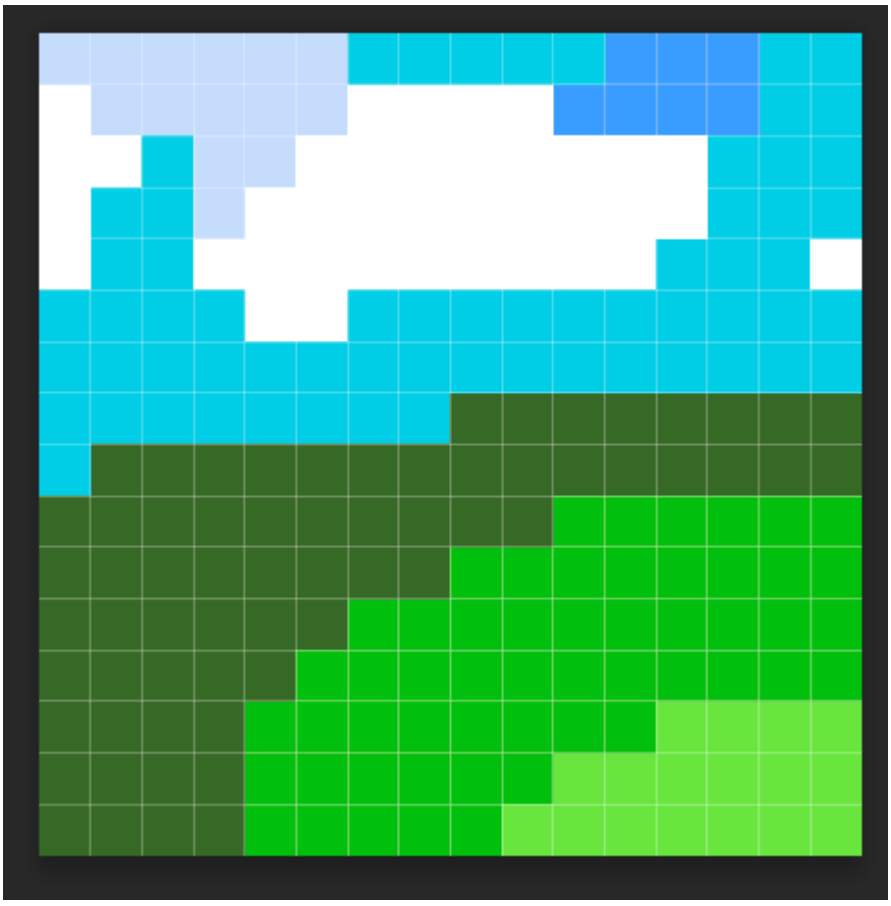

Рисунок 1 – Исходное изображение

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

16x16 cell

	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]	[-29,27,127]	[-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]	[75,91,124]	[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]	[-33,77,100]	[-33,77,100]
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]	[-33,77,100]	[-33,77,100]	[-33,77,100]
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]	[-33,77,100]	[-33,77,100]	[-33,77,100]	[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]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]	[-53,-23,-81]
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]	[-22,62,-80]	[-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]	[-22,62,-80]	[-22,62,-80]	[-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]	[-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]
13	[-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]	[-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]	[25,101,-48]	[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 | U | V

16x16 double

	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	22	22	22	22	47	47
3	127	127	47	90	90	127	127	127	127	127	127	127	127	47	47	47
4	127	47	47	90	127	127	127	127	127	127	127	127	127	47	47	47
5	127	47	47	127	127	127	127	127	127	127	127	127	47	47	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	-39	-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	21	21	21	21	21	21
11	-39	-39	-39	-39	-39	-39	-39	-39	21	21	21	21	21	21	21	21
12	-39	-39	-39	-39	-39	-39	21	21	21	21	21	21	21	21	21	21
13	-39	-39	-39	-39	-39	21	21	21	21	21	21	21	21	21	21	21
14	-39	-39	-39	-39	21	21	21	21	21	21	21	21	61	61	61	61
15	-39	-39	-39	-39	21	21	21	21	21	21	61	61	61	61	61	61
16	-39	-39	-39	-39	21	21	21	21	21	61	61	61	61	61	61	61

Рисунок 3 – Яркостная компонента Y

Y U V

16x16 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	117	117	117	117	117	117	71	71	71	71	71	92	92	92	71	71
2	128	117	117	117	117	117	128	128	128	128	92	92	92	92	71	71
3	128	128	71	117	117	128	128	128	128	128	128	128	128	71	71	71
4	128	71	71	117	128	128	128	128	128	128	128	128	128	71	71	71
5	128	71	71	128	128	128	128	128	128	128	128	128	71	71	71	128
6	71	71	71	71	128	128	71	71	71	71	71	71	71	71	71	71
7	71	71	71	71	71	71	71	71	71	71	71	71	71	71	71	71
8	71	71	71	71	71	71	71	71	118	118	118	118	118	118	118	118
9	71	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118
10	118	118	118	118	118	118	118	118	118	118	98	98	98	98	98	98
11	118	118	118	118	118	118	118	118	98	98	98	98	98	98	98	98
12	118	118	118	118	118	118	98	98	98	98	98	98	98	98	98	98
13	118	118	118	118	118	98	98	98	98	98	98	98	98	98	98	98
14	118	118	118	118	98	98	98	98	98	98	98	98	102	102	102	102
15	118	118	118	118	98	98	98	98	98	98	102	102	102	102	102	102
16	118	118	118	118	98	98	98	98	98	102	102	102	102	102	102	102

Рисунок 4 – Цветоразностная компонента U

Y U V

16x16 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	173	173	173	173	173	173	147	147	147	147	147	178	178	178	147	147
2	171	173	173	173	173	173	171	171	171	171	171	178	178	178	147	147
3	171	171	147	173	173	171	171	171	171	171	171	171	171	147	147	147
4	171	147	147	173	171	171	171	171	171	171	171	171	171	147	147	147
5	171	147	147	171	171	171	171	171	171	171	171	171	147	147	147	171
6	147	147	147	147	171	171	147	147	147	147	147	147	147	147	147	147
7	147	147	147	147	147	147	147	147	147	147	147	147	147	147	147	147
8	147	147	147	147	147	147	147	147	86	86	86	86	86	86	86	86
9	147	86	86	86	86	86	86	86	86	86	86	86	86	86	86	86
10	86	86	86	86	86	86	86	86	86	86	64	64	64	64	64	64
11	86	86	86	86	86	86	86	86	64	64	64	64	64	64	64	64
12	86	86	86	86	86	86	64	64	64	64	64	64	64	64	64	64
13	86	86	86	86	86	64	64	64	64	64	64	64	64	64	64	64
14	86	86	86	86	64	64	64	64	64	64	64	64	75	75	75	75
15	86	86	86	86	64	64	64	64	64	64	75	75	75	75	75	75
16	86	86	86	86	64	64	64	64	64	75	75	75	75	75	75	75

Рисунок 5 – Цветоразностная компонента V

Дискретное вейвлет преобразование

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

На этом шаге изображение разбивается на квадратные блоки – *тайлы*.

Само преобразование в одномерном случае представляет собой *скалярное произведение* коэффициентов фильтра на строку преобразуемых значений (в нашем случае - на строку изображения)

Полученная матрица аппроксимации LL2 представлена на рисунке 6:

	1	2	3	4	5	6	7	8
1	198.5000	180.0000	180.0000	174.0000	174.0000	56.5000	44.0000	94.0000
2	214	137.0000	235.5000	254.0000	254.0000	254.0000	174	94.0000
3	134	134	254.0000	174.0000	174.0000	174.0000	94.0000	134
4	94.0000	94.0000	94.0000	94.0000	8.0000	8.0000	8.0000	8.0000
5	-35.0000	-78.0000	-78.0000	-78.0000	-78.0000	-18.0000	-18.0000	-18.0000
6	-78.0000	-78.0000	-78.0000	-18.0000	42.0000	42.0000	42.0000	42.0000
7	-78.0000	-78.0000	12.0000	42.0000	42.0000	42.0000	82.0000	82.0000
8	-78.0000	-78.0000	42.0000	42.0000	62	122.0000	122.0000	122.0000
9								

Рисунок 6 – Матрица аппроксимации

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

16x16 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	198	179	179	173	173	51	38	90	55	1	1	238	238	38	1	1
2	214	134	236	255	255	255	173	90	119	1	55	1	1	1	1	1
3	131	131	255	173	173	173	90	131	119	119	1	238	238	238	1	119
4	90	90	90	90	1	1	1	1	1	1	1	1	255	255	255	255
5	28	73	73	73	73	11	11	11	128	1	1	1	1	178	178	178
6	73	73	73	11	36	36	36	36	1	1	1	178	1	1	1	1
7	73	73	5	36	36	36	77	77	1	1	89	1	1	1	119	119
8	73	73	36	36	56	119	119	119	1	1	1	1	60	1	1	1
9	59	1	1	1	1	40	1	1	110	1	1	1	1	75	1	1
10	128	138	59	1	1	1	255	1	238	1	110	1	1	1	1	1
11	128	128	1	1	1	1	1	128	238	238	1	1	1	1	1	238
12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	138	1	1	1	1	1	1	1	255	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	1	1	96	1	1	1	1	1	1	1	178	1	1	1	1	1
16	1	1	1	1	64	1	1	1	1	1	1	1	119	1	1	1
17																

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:

8x8 double

	1	2	3	4	5	6	7	8
1	19	18	18	17	17	5	4	9
2	21	13	23	25	25	25	17	9
3	13	13	25	17	17	17	9	13
4	9	9	9	9	0	0	0	0
5	-3	-7	-7	-7	-7	-1	-1	-1
6	-7	-7	-7	-1	4	4	4	4
7	-7	-7	1	4	4	4	8	8
8	-7	-7	4	4	6	12	12	12
9								

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 в таблице 1:

Таблица 1 – Результат арифметического кодирования

Коэф.	Вероятность	Нижн.гран.	Верхн.гран.	Расчит.н.гр.	Расчит.верх.гр.
19	0.0156	0.8906	0.9062	0.890625	0.906250
21	0.0156	0.9062	0.9219	0.904785	0.905029

13	0.0625	0.7031	0.7656	0.904957	0.904972
9	0.1094	0.5469	0.6562	0.904965	0.904967
-3	0.0156	0.1719	0.1875	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
18	0.0312	0.8594	0.8906	0.904965	0.904965
13	0.0625	0.7031	0.7656	0.904965	0.904965
13	0.0625	0.7031	0.7656	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
18	0.0312	0.8594	0.8906	0.904965	0.904965
23	0.0156	0.9219	0.9375	0.904965	0.904965
25	0.0625	0.9375	1.0000	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
1	0.0156	0.3125	0.3281	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965
25	0.0625	0.9375	1.0000	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
-1	0.0625	0.1875	0.2500	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965

25	0.0625	0.9375	1.0000	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965
0	0.0625	0.2500	0.3125	0.904965	0.904965
-7	0.1719	0.0000	0.1719	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
6	0.0156	0.5000	0.5156	0.904965	0.904965
5	0.0156	0.4844	0.5000	0.904965	0.904965
25	0.0625	0.9375	1.0000	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965
0	0.0625	0.2500	0.3125	0.904965	0.904965
-1	0.0625	0.1875	0.2500	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
12	0.0469	0.6562	0.7031	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
17	0.0938	0.7656	0.8594	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
0	0.0625	0.2500	0.3125	0.904965	0.904965
-1	0.0625	0.1875	0.2500	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
8	0.0312	0.5156	0.5469	0.904965	0.904965
12	0.0469	0.6562	0.7031	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
9	0.1094	0.5469	0.6562	0.904965	0.904965
13	0.0625	0.7031	0.7656	0.904965	0.904965
0	0.0625	0.2500	0.3125	0.904965	0.904965
-1	0.0625	0.1875	0.2500	0.904965	0.904965
4	0.1562	0.3281	0.4844	0.904965	0.904965
8	0.0312	0.5156	0.5469	0.904965	0.904965
12	0.0469	0.6562	0.7031	0.904965	0.904965

Закодированное значение: 0.904965449241

Итоговый диапазон: [0.904965449241 , 0.904965449241]

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.

$V = 72$ бит

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

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);
[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));
img_shifted(:,:,1) = img(:,:,1) - shift_vals(1);
img_shifted(:,:,2) = img(:,:,2) - shift_vals(2);
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
        RGB_matrix{i,j} = squeeze(img_shifted(i,j,:))';
    end
end

%% Шаг 2: Преобразование цветового пространства
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}';
        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 — ТОЛЬКО 1 УРОВЕНЬ (первый проход)
wname = 'haar';
[C, S] = wavedec2(Y, 1, wname);  % ← 1 уровень = первый проход

% Извлечение компонент 1-го уровня
LL1 = appcoef2(C, S, wname, 1);   % аппроксимация
[HL1, LH1, HH1] = detcoef2('all', C, S, 1);  % детали

% Покажем размеры
fprintf('Исходный Y: %d x %d\n', size(Y));
fprintf('После 1-го прохода (LL1, HL1, LH1, HH1): %d x %d\n', size(LL1));

% Собираем все компоненты в единую матрицу C1 для визуализации
LL1_vis = wcodemat(LL1, 255);
HL1_vis = wcodemat(HL1, 255);
LH1_vis = wcodemat(LH1, 255);
HH1_vis = wcodemat(HH1, 255);
```

```
C1 = [LL1_vis, HL1_vis; ...
      LH1_vis, HH1_vis];

% Отображаем (опционально — можно закомментировать)
figure;
imshow(uint8(C1), []);
title('Результат первого прохода DWT: [LL1 HL1; LH1 HH1]');

%% Шаг 4: Квантование (применим к LL1, как вы делали с LL2)
Q = 10;
deadzone_quant = @(x) sign(x) .* floor((abs(x))/Q) .* (abs(x) >= Q);

LL1_q = deadzone_quant(LL1);
% (можно также заквантовать HL1, LH1, HH1, но вы кодируете только LL)

%% Шаг 5: Арифметическое кодирование (теперь для LL1_q)
Xq = LL1_q(:);
code = arithmetic_encode_verbose(Xq);

%% Функция арифметического кодирования (без изменений)
function code = arithmetic_encode_verbose(symbols)
    alphabet = unique(symbols);
    counts = histc(symbols, alphabet);
    prob = counts / sum(counts);
    prob = prob(:);
    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
');

    for k = 1:n
        sym = symbols(k);
        idx = find(alphabet == sym);
        c_low  = cum_prob(idx);
        c_high = cum_prob(idx + 1);
        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
```