Лабораторна робота 6:

“Блокова обробка. Реалізація алгоритму JPEG”

Виконав:

Білобрицький Денис Анатолійович

544 група

Мета: Метою даної лабораторної роботи є набуття знань про існуючи методи стиснення зображень та ознайомитися з основними з них.

Хід роботи:

Спочатку завантажуємо зображення для роботи:

```
I1 = imread('Images/cat.jpg');
I2 = imread('Images/dog.jpg');
I3 = imread('cameraman.tif');
```

```
imshow(I1);
title('Зображення 1');
```

Рисунок 1 – Завантаження зображень ти вивід одного з них

З використанням функції **rgb2gray** перетворюємо кольорові зображення в чорно-білі.

```
I1_gray = rgb2gray(I1)
```

```
I1_gray = 199×200 uint8 matrix
   206   211   216   221   225   231   237   240   240   240   240   240   240   241   239   237   238   237   237   237   236  ...
   208   212   217   221   226   232   237   239   241   241   241   240   241   241   239   237   236   236   237   238   237
   207   211   216   221   227   232   237   238   243   243   242   242   241   241   240   238   236   237   238   239   238
   204   209   215   220   225   231   235   238   243   244   243   242   242   241   240   238   238   238   238   238   238
   201   207   215   220   223   228   234   237   242   243   243   242   241   241   240   238   239   238   238   237   237
   201   208   216   220   223   227   232   236   240   242   242   241   240   240   239   239   238   238   236   235   236
   204   210   217   221   223   226   231   234   238   240   241   240   239   239   239   238   238   237   236   235   235
   206   212   218   221   223   226   229   231   236   238   240   239   238   238   238   238   238   238   237   235   235
   209   214   219   222   223   224   227   230   237   239   240   239   238   238   238   237   238   237   236   236   235
   214   217   220   222   223   225   228   231   236   238   239   238   238   238   238   237   237   236   235   235   235
     ⋮
```

```
imshow(I1_gray);
title('Зображення 1 (чорно-біле)');
```

Рисунок 2 – Перетворення кольорового зображення

З використанням функції **dctmtx** створюємо матрицю коефіцієнтів 8x8 і задаємо процедуру **dct = @(block_struct) T * block_struct.data * T'**, далі використовуємо функцію **blockproc.**

```
% задаємо розмір блока і створюємо матрицю
blockSize = 8;
T = dctmtx(blockSize);

dct_fun = @(block_struct) T * block_struct.data * T';

% поблочне ДКП для кожного зображення
DCT1 = blockproc(I1_gray, [blockSize blockSize], dct_fun);
DCT2 = blockproc(I2_gray, [blockSize blockSize], dct_fun);
DCT3 = blockproc(I3_gray, [blockSize blockSize], dct_fun);
```

```
imshow(log(abs(DCT1) + 1), []);
title('DCT спектр зображення 1 (лог масштаб)');
```

Рисунок 3 – Поблокове ДКП у вигляді зображення

Поясніть вигляд отриманого зображення.

Кожний 8×8 блок представляє локальні частотні компоненти, де у лівому верхньому куті сконцентровані низькі частоти (основна структура), а у правому нижньому — високі (деталі, шуми); сітка з повторюваних патернів вказує на блокову обробку, найяскравіші точки — домінуючі частоти, а темні або рівномірні області свідчать про однорідні ділянки зображення.

Далі за допомогою оберненої процедури **invdct = @(block_struct) T' * block_struct.data * T** перетворюємо отримані спектри назад в зображення:

```
invdct = @(block_struct) T' * block_struct.data * T;

I1_restored = blockproc(DCT1, [blockSize blockSize], invdct);
I2_restored = blockproc(DCT2, [blockSize blockSize], invdct);
I3_restored = blockproc(DCT3, [blockSize blockSize], invdct);
```

```
imshow(I1_restored); title('Відновлене зображення 1');
```

Рисунок 4 – Вигляд відновленого зображення

Далі виконаємо квантування ДКП спектру з кроками 0.1, 0.5, 1, 2, 4 за допомогою коду:

```
N_values = [0.1, 0.5, 1, 2, 4];

for i = 1:length(N_values)
    N = N_values(i);
    DCT1_q = N * round(DCT3 / N);

    figure;
    imshow(log(abs(DCT1_q) + 1), []); colormap(jet); colorbar;
    title(['Квантований спектр DCT (N = ', num2str(N), ')']);
end
```

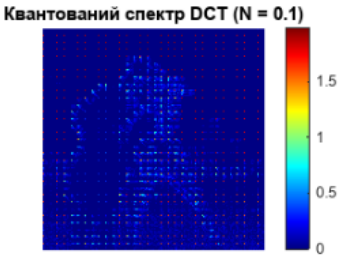

Рисунок 4 – Квантування ДКП спектру зображення з кроком 0.1

Так як і раніше все повторюємо для кожного зображення. Після чого відновлюємо зображення.

Поясніть, як працює ця процедура, і що отримується в її результаті.

Це дуже проста і ефективна форма уніформного квантування, яка призводить до того, що всі значення "прив'язуються" до найближчого кратного числа N.

В результаті ми отримуємо зменшення роздільної здатності коефіцієнтів ДКП згладжуються дрібні варіації, зменшується кількість унікальних значень (що сприяє стисканню, як у JPEG), але призводить до втрати точності й деталей при великому N. Досить добре видно при відновленні зображення з високим кроком квантування N:

Рисунок 5 – Відновлені зображення з різними кроками квантування N

Як результат чим більший крок тим більше стиснення і втрата якості.

Отже останнім кроком буде використання **маски** і відновлення зображення після неї.

```
mask = [
    1 1 1 1 0 0 0 0;
    1 1 1 0 0 0 0 0;
    1 1 0 0 0 0 0 0;
    1 0 0 0 0 0 0 0;
    0 0 0 0 0 0 0 0;
    0 0 0 0 0 0 0 0;
    0 0 0 0 0 0 0 0;
    0 0 0 0 0 0 0 0
];

% застосування маски до кожного блока
B = blockproc(DCT3, [8 8], @(block_struct) mask .* block_struct.data);

% зворотне перетворення ДКП
B_restored = blockproc(B, [8 8], invdct);

imshow(log(1 + abs(B_restored)), []);
title('Використання маски');
```

Рисунок 6 – Використання маски і відновлення зображення з спектру

Поясніть, отриманий результат і яка мета досягається квантуванням коефіцієнтів ДКП?

Після відновлення зображення з квантованого ДКП-спектру спостерігається зменшення якості: втрачаються дрібні деталі, з'являються артефакти. Це відбувається через втрату точності при квантуванні — менш значущі (високочастотні) коефіцієнти обнуляються або набувають грубих значень.

Мета квантування — зменшити обсяг даних, зберігаючи при цьому загальну структуру зображення. Квантування дозволяє сильно стиснути зображення з допустимими втратами, саме тому воно лежить в основі JPEG-компресії.