Университет ИТМО

Цифровая обработка сигналов

Лабораторная работа №5

Вариант 3

Выполнила: Калугина Марина

Группа: P3402

г. Санкт-Петербург

2020 г.

Задание

1. Создайте функцию *compressRatio*, определяющую степень сжатия изображения;
2. Создайте функцию *directDCT*, осуществляющую прямое ДКП изображения по формуле 1;
3. Создайте функцию *invertDCT,* осуществляющую обратное ДКП изображения по формуле 5;
4. Используя полученное для данного задания изображение и предыдущий пример осуществите 12 итераций, на каждой из которых будет происходить постепенное обнуление коэффициентов ДКП в порядке, обратном зиг-загообразному. (т.е., начиная с высокочастотного). DC-коэффициент обнулять не нужно.
   На каждой итерации а) при помощи функции compressRatio вычислите уровень сжатия полученного изображения, б) при помощи функции corr2 вычислите корреляцию Пирсона между исходным и сжатым изображением.
5. При помощи функции plot постройте графики зависимости а) между количеством задействованных коэффициентов и качеством изображения, б) между уровнем сжатия и качеством изображения. Оси графиков должны быть подписаны.
6. Используя график “а)” определите приблизительное положение высокочастотных, среднечастотных и низкочастотных коэффициентов ДКП. Используя график “б)” определите зависимость между уровнем сжатия и качеством полученного изображения.
7. Анализируя внешний вид изображений, полученных при высоких коэффициентах сжатия, определите главный недостаток формата JPEG.

Выполнение

Рисунок 1 - исходное изображение

Рисунок 2 - сжатое изображени

lab5.m

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


I = imread("putin_512.png");
I = im2double(I);
B = directDCT(I);
```

```
quantized = quantize(B);

k = 0;
elem =0;
x = [];
y = [];
cr =[];
for i = 1:12
    if i<=midPoint
        for j =0:k
            mask(8-j,8-(k-j))=0;
            elem = elem+1;
        end
        k=k+1;
    else

        for j = 1:k-1
           mask(k-j,j)=0;
           elem = elem+1;
        end
        k=k-1;
    end
    B2 = blkproc(B,[8 8],'P1.*x',mask);
    dequantized = dequantize(B2);
    D = inverseDCT(B2);
    imwrite(D, "putin_512_1.jpeg");
    x(i)= 64 - elem;
    y(i)=compressRatio("putin_512.png", "putin_512_1.jpeg");
    cr(i) = corr2(I,D);
end

figure(1)
plot(x,cr, '-o')
title('Зависимость качества изображения от к-ва коэффициентов')
xlabel('К-во коэффициентов')
ylabel('Коэф. корреляции')
for t = 1:numel(x)
  text(x(t)+0.1,cr(t),['(',num2str(x(t)), '; ', num2str(cr(t)),
')'])
end

figure(2)
plot(y,cr, '-o')
title('Зависимость качества изображения от уровня сжатия')
xlabel('Коэф. сжатия')
ylabel('Коэф. корреляции')
for t = 1:numel(x)
```

```
  text(y(t)+0.1,cr(t),['(',num2str(y(t)), '; ', num2str(cr(t)),
')'])
end
```

compressRatio.m

```
function k = compressRatio(original, compressed)
    k = imfinfo(original).FileSize /
imfinfo(compressed).FileSize;
end
```

directDCT.m

```
function dct_domain = directDCT(image)
    DCT_matrix8 = dct(eye(8));
    iDCT_matrix8 = DCT_matrix8';
    [row, coln]= size(image);
    for i1=[1:8:row]
        for i2=[1:8:coln]
            zBLOCK=image(i1:i1+7,i2:i2+7);
            win1=DCT_matrix8*zBLOCK*iDCT_matrix8;
            dct_domain(i1:i1+7,i2:i2+7)=win1;
        end
    end
end
```

inverseDCT.m

```
function dct_restored = inverseDCT(image)
    DCT_matrix8 = dct(eye(8));
    iDCT_matrix8 = DCT_matrix8';
    [row, coln]= size(image);
    for i1=[1:8:row]
        for i2=[1:8:coln]
            win3 = image(i1:i1+7,i2:i2+7);
            win4=iDCT_matrix8*win3*DCT_matrix8;
            dct_restored(i1:i1+7,i2:i2+7)=win4;
        end
    end
end
```

quantize.m

```
function dct_quantized = quantize(dct_domain)
    QX = [3 5 7 9 11 13 15 17;
        5 7 9 11 13 15 17 19;
        7 9 11 13 15 17 19 21;
        9 11 13 15 17 19 21 23;
        11 13 15 17 19 21 23 25;
        13 15 17 19 21 23 25 27;
        15 17 19 21 23 25 27 29;
```

```
        17 19 21 23 25 27 29 31];
    QX = double(QX);
    [row, coln]= size(dct_domain);
    for i1=[1:8:row]
        for i2=[1:8:coln]
            win1 = dct_domain(i1:i1+7,i2:i2+7);
            win2 = round(win1./QX);
            dct_quantized(i1:i1+7,i2:i2+7)=win2;
        end
    end
end
```

dequantize.m

```
function dct_dequantized = dequantize(dct_domain)
    QX = [3 5 7 9 11 13 15 17;
        5 7 9 11 13 15 17 19;
        7 9 11 13 15 17 19 21;
        9 11 13 15 17 19 21 23;
        11 13 15 17 19 21 23 25;
        13 15 17 19 21 23 25 27;
        15 17 19 21 23 25 27 29;
        17 19 21 23 25 27 29 31];
    QX = double(QX);
    [row, coln]= size(dct_domain);
    for i1=[1:8:row]
        for i2=[1:8:coln]
            win2 = dct_domain(i1:i1+7,i2:i2+7);
            win3 = win2.*QX;
            dct_dequantized(i1:i1+7,i2:i2+7) = win3;
        end
    end
end
```

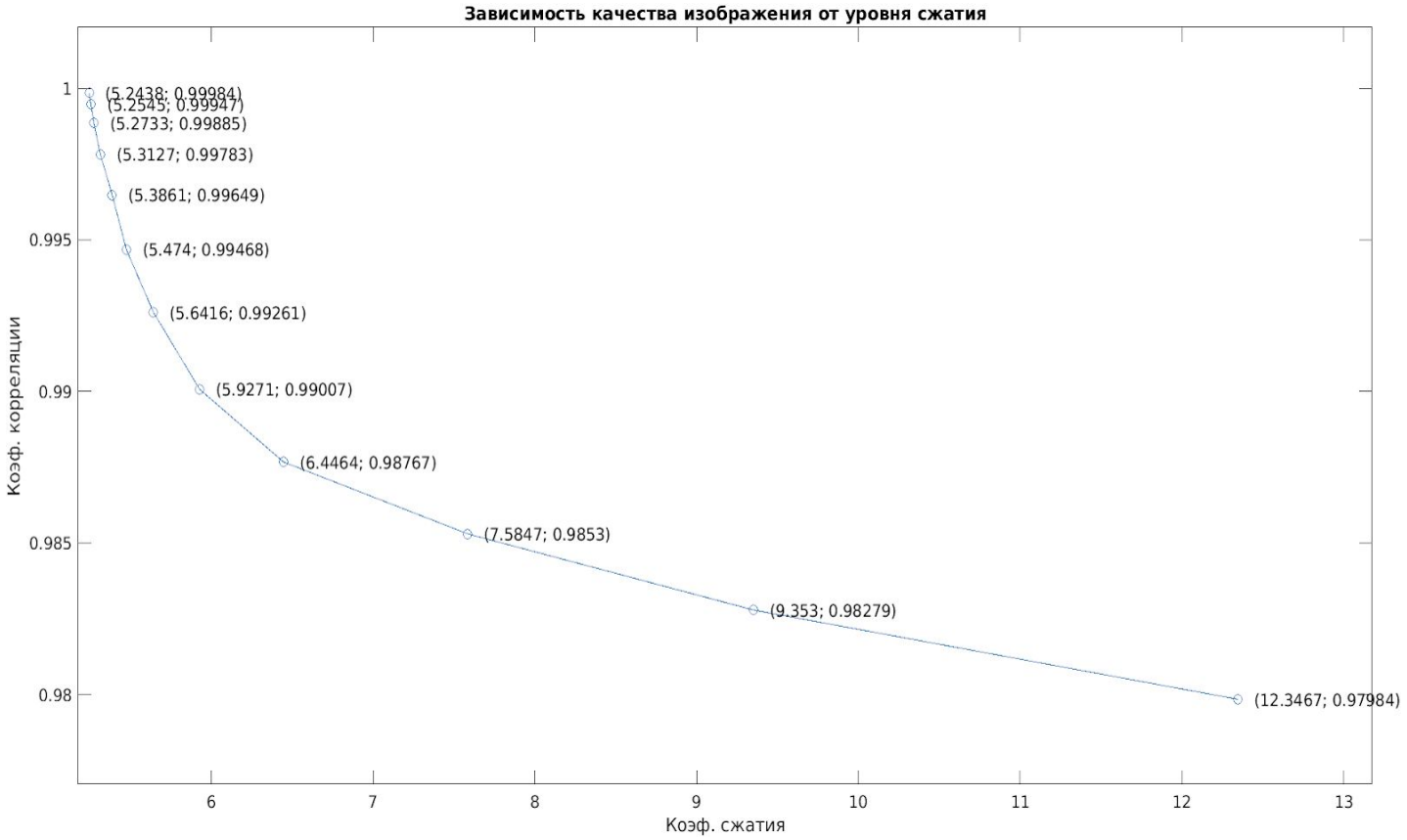

Рисунок 3 - график зависимости качества изображения от уровня сжатия

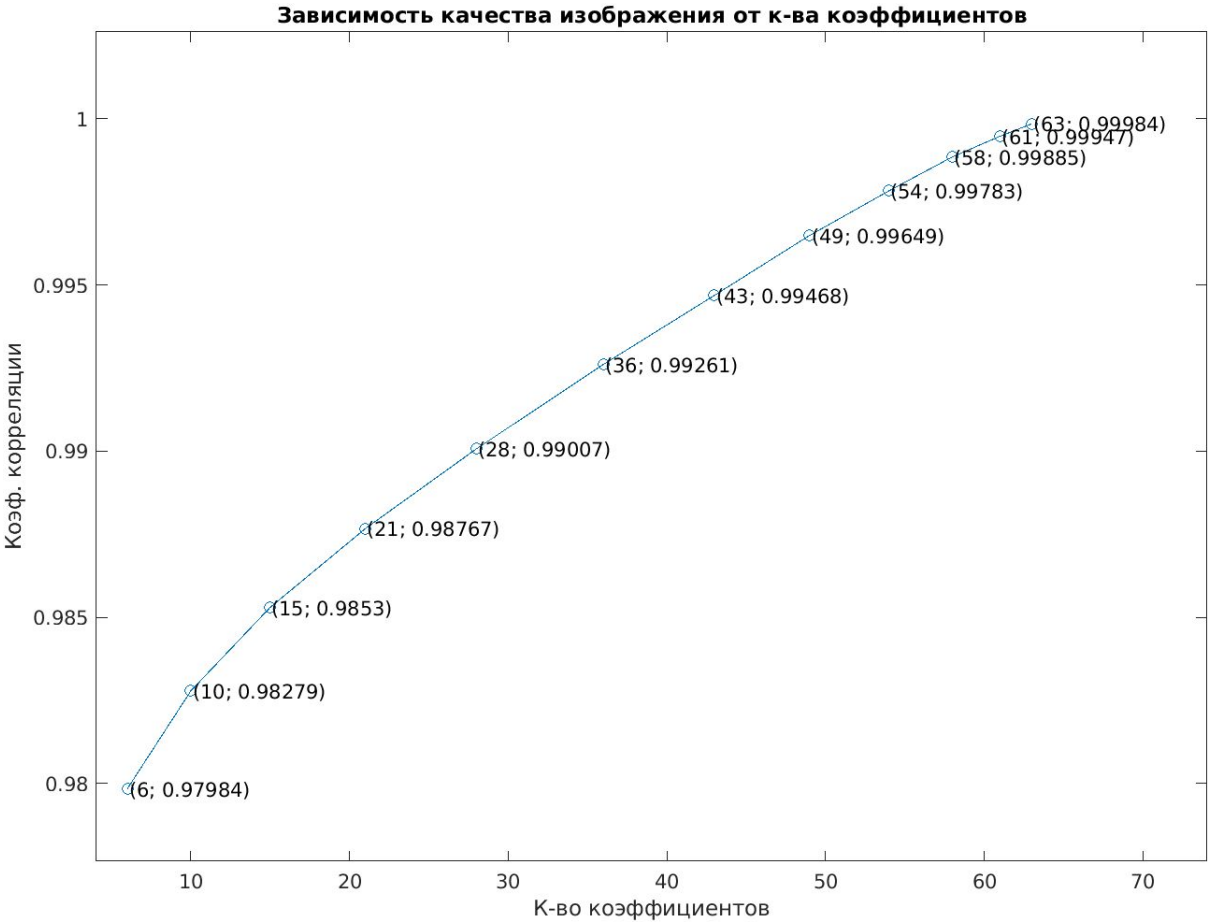

Рисунок 4 - график зависимости качества изображения от количества коэффициентов

Вывод

В ходе данной работы были реализованы алгоритмы прямого и обратного дискретного косинусного преобразования. При помощи данных алгоритмов было произведено сжатие изображений по алгоритму JPEG. В ходе данного сжатия происходит сильная потеря качества изображения и чем выше степень сжатия, тем хуже качество изображения. При применении ДКП к изображению получется матрица, в которой коэффициенты в левом верхнем углу соответствуют низкочастотной составляющей изображения, а в правом нижнем – высокочастотной. Понятие частоты следует из рассмотрения изображения как двумерного сигнала (аналогично рассмотрению звука как сигнала). Плавное изменение цвета соответствует низкочастотной составляющей, а резкие скачки – высокочастотной.