МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Навчально-науковий інститут фізико-технічних та комп’ютерних наук

Кафедра комп’ютерних програмного забезпечення комп’ютерних систем

**ЗВІТ  
про виконання лабораторної роботи № 5**

**з дисципліни**

# **«Прикладна інтелектуальна обробка сигналів та зображень»**

**на тему: «“Стиснення зображень”»**

Виконала

студентка 5-го курсу

544 групи

\_Турецька Даяна Вікторівна\_

(П.І.Б)

# Чернівці, 2025

## **1. Мета**

Метою даної роботи є ознайомлення з процедурою стиснення растрових зображень через дискретне косинусне перетворення (DCT), квантування його коефіцієнтів, а також порівняння з простим квантуванням пікселів у просторі.

## **2. Хід роботи**

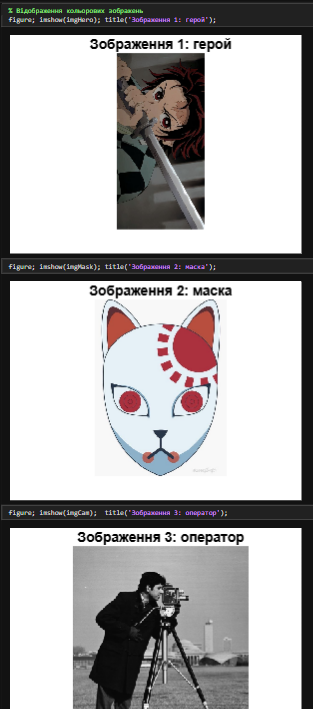
### **2.1 Завантаження вихідних зображень**

Використовуємо imread для трьох файлів:  
imgHero = imread('imgFolder/hero.jpg');

imgMask = imread('imgFolder/mask.jpg');

imgCam = imread('cameraman.tif');

Відображаємо кожне через imshow та підписуємо title.



**Рисунок 1** – Завантаження та відображення вихідних зображень

### **2.2 Перетворення в градації сірого**

Для кольорових зображень застосовуємо rgb2gray:  
grayHero = rgb2gray(imgHero);

grayMask = rgb2gray(imgMask);

grayCam = imgCam; % уже ч/б

Відображаємо отримані чорно-білі зображення.





**Рисунок 2** – Результат перетворення в градації сірого

### **2.3 Дискретне косинусне перетворення (DCT)**

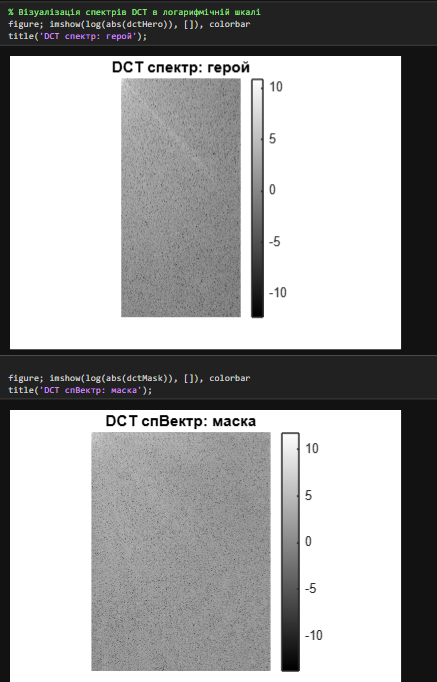
Застосовуємо dct2 до кожного ч/б зображення:  
dctHero = dct2(grayHero);

dctMask = dct2(grayMask);

dctCam = dct2(grayCam);

Візуалізуємо спектри у логарифмічній шкалі:  
imshow(log(abs(dctHero)), []), colorbar

title('DCT спектр: герой');





**Рисунок 3** – DCT-спектри Героя, Маски та Оператора

### **2.4 Відновлення зображень через обернене DCT**

Використовуємо idct2 для кожного спектру:  
reconHero = idct2(dctHero);

reconMask = idct2(dctMask);

reconCam = idct2(dctCam);

Перевіряємо, що при відсутності квантування якість відновлення практично ідентична оригіналу.





**Рисунок 4** – Відновлені зображення після зворотного DCT

### **2.5 Квантування DCT-коефіцієнтів**

Виконуємо квантування з кроками **5**, **10** та **30** окремо для кожного зображення.

#### **2.5.1 Для фото Hero**

steps = [5, 10, 30];

for q = steps

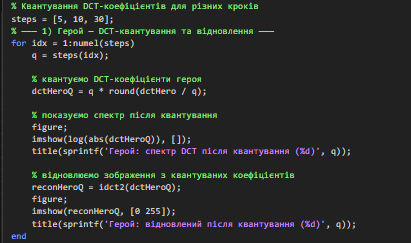
dctHeroQ = q \* round(dctHero / q);

imshow(log(abs(dctHeroQ)), []), title(sprintf('Герой: DCT після квант. (%d)', q));

reconHeroQ = idct2(dctHeroQ);

imshow(reconHeroQ, [0 255]), title(sprintf('Герой: відновлений (%d)', q));

end



**Рисунок 5.1**  – код квантування для фото Герой



**Рисунок 5.2** – Спектри і відновлені зображення Героя для кроків 5



**Рисунок 5.3** – Спектри і відновлені зображення Героя для кроків 10



**Рисунок 5.4** – Спектри і відновлені зображення Героя для кроків 30

#### **2.5.2 Для фото Mask**

for q = steps

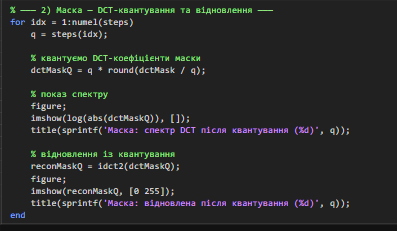
dctMaskQ = q \* round(dctMask / q);

imshow(log(abs(dctMaskQ)), []), title(sprintf('Маска: DCT після квант. (%d)', q));

reconMaskQ = idct2(dctMaskQ);

imshow(reconMaskQ, [0 255]), title(sprintf('Маска: відновлена (%d)', q));

end



**Рисунок 6.1**  – код квантування для фото Mаска



**Рисунок 6.2** – Спектри і відновлені зображення Маска для кроків 5



**Рисунок 6.3** – Спектри і відновлені зображення Маска для кроків 10



**Рисунок 6.4** – Спектри і відновлені зображення Маска для кроків 30

#### **2.5.3 Для фото Cameraman**

for q = steps

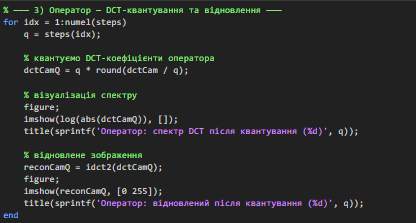
dctCamQ = q \* round(dctCam / q);

imshow(log(abs(dctCamQ)), []), title(sprintf('Оператор: DCT після квант. (%d)', q));

reconCamQ = idct2(dctCamQ);

imshow(reconCamQ, [0 255]), title(sprintf('Оператор: відновлений (%d)', q));

end



**Рисунок 7.1**  – код квантування для фото Оператор



**Рисунок 7.2** – Спектри і відновлені зображення Героя для кроків 5



**Рисунок 7.3** – Спектри і відновлені зображення Героя для кроків 10



**Рисунок 7.4** – Спектри і відновлені зображення Героя для кроків 30

#### **2.6 Пояснення процедури квантування DCT**

**Поясніть, як працює ця процедура, і що отримаємо в результаті.**

Квантування округлює кожен DCT-коефіцієнт до найближчого кратного кроку q.

* **Малі q** (наприклад, 5) зберігають більшість деталей.
* **Великі q** (30 і більше) суттєво знижують обсяг даних за рахунок відкидання високочастотних компонентів, але викликають видимі артефакти (шум, ефект “блоків”).

Квантування зменшує точність результатів ДКП, роблячи їх дискретними, в залежності від величини кроку N. Незначний крок зберігає деталі, великий — збільшує стиснення, але втрачає властивості. Це дає змогу балансувати між стисненням та якістю зображення.

## **3. Просте квантування у просторі**

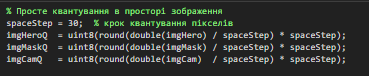
**Чи можливо добитися аналогічної мети й результату, квантуючи вихідне зображення, а не коефіцієнти його ДКП?**

Реалізація цього за допомогою коду нижче:

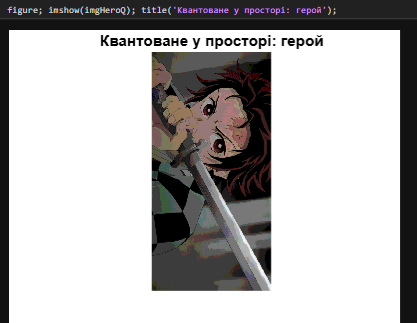
Квантуємо пікселі вихідних зображень із кроком spaceStep = 30:  
  
imgHeroQ = uint8(round(double(imgHero)/spaceStep)\*spaceStep);

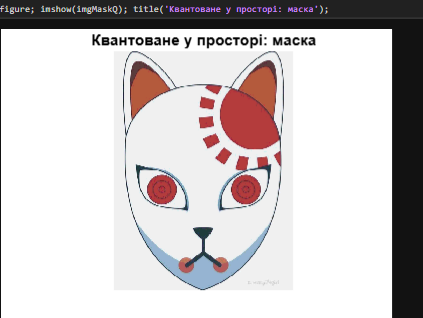
imgMaskQ = uint8(round(double(imgMask)/spaceStep)\*spaceStep);

imgCamQ = uint8(round(double(imgCam)/spaceStep)\*spaceStep);



**Рисунок 8** – Код простого квантування у просторі

Відображаємо результати.  


**Рисунок 9.1** – Квантування простору для Героя

**Рисунок 9.2** – Квантування простору для Маски



**Рисунок 9.3**  – Квантування простору для Героя, Маски та Оператора

## **4. Пояснення результатів**

### **4.1 Вплив кроку квантування DCT**

Для кожного зображення ми виконали квантування DCT-коефіцієнтів із трьома різними кроками **N = 5, 10, 30**

**N = 5**: коефіцієнти округлюються помірно, тому більшість частот залишається. Різниця з оригіналом практично не помітна людському оку.

**N = 10**: блочність стає видно на однорідних ділянках, втрачається легка текстурна інформація.

**N = 30**: значне відкидання високочастотних гармонік призводить до «заблокованості» зображення і сильного зменшення кількості градацій.

**Висновок:** із збільшенням кроку N ступінь стиснення зростає, але якість відновлення падає експоненціально.

### **4.2 Порівняння з простим квантуванням у просторі**

* **Просте квантування пікселів (N = 30)** Відкидання дрібних градацій кольору рівномірне по всьому зображенню:  
  + Контраст переходів (плавні градієнти) перетворюється на великі рівномірні блоки одного тону.
  + Зображення стає «пластиковим», без текстурних деталей.
* **DCT + квантування (N = 30)** Блочність концентрується в межах DCT-блоків (8×8); при цьому в одних ділянках (низькі частоти) збережено більше інформації, ніж у простому квантуванні.

**Висновок:**

**Просте квантування** рівномірно зменшує кількість тонів в усьому просторі, **DCT-квантування** в першу чергу відкидає високочастотні деталі (зріз шуму й дрібних текстур), краще зберігаючи основні контури.

**Які недоліки ви бачите в стисненні зображень із використанням його ДКП і квантування коефіцієнтів ДКП?**

* + «Ефект блоків» при сильному стисненні (видно 8×8 блоки).
  + З’являється шум і артефакти на межах блоків.
  + Менш ефективно для зображень із різкими горизонталями/вертикалями.

## **5. Висновки**

У роботі показано, що поєднання DCT і адаптивного квантування є дієвим методом стискання зображень, що дозволяє балансувати між ступенем стиснення та якістю. На відміну від простого квантування у просторі, цей підхід краще зберігає візуально важливі деталі, хоча й вимагає складніших обчислень та може спричиняти блокові артефакти при жорсткому стисненні.