# Raport z Laboratorium 5
# Steganografia: Metoda Kocha-Zhao

Grzegorz Golonka

Nr Albumu: 156742

## Temat:

Ukrywanie danych w obrazach metodą Kocha-Zhao z wykorzystaniem dyskretnej transformacji kosinusowej (DCT).

## Cel laboratorium:

Celem laboratorium było zapoznanie się z jedną z klasycznych metod steganografii cyfrowej, czyli ukrywania danych w obrazach w sposób odporny na kompresję JPEG.
Metoda, którą realizowaliśmy, bazuje na modyfikacji wybranych współczynników DCT w blokach 8×8 pikseli.

## Wprowadzenie teoretyczne:

Steganografia polega na ukrywaniu informacji w taki sposób, aby jej obecność nie była łatwa do wykrycia. W przypadku obrazów, często wykorzystywana jest transformacja DCT, ponieważ dzięki niej możemy manipulować współczynnikami reprezentującymi różne częstotliwości obrazu.

W metodzie Kocha-Zhao:

- Najpierw dzielimy obraz na bloki 8x8 pikseli.
- Następnie wykonujemy na każdym bloku transformację DCT.
- W każdym bloku wybieramy dwie współczynniki (najczęściej średnich częstotliwości), które modyfikujemy w zależności od bitu wiadomości, który chcemy ukryć:
    - Jeśli chcemy zapisać bit 1, zwiększamy pierwszy współczynnik względem drugiego.
    - Jeśli chcemy zapisać bit 0, zmniejszamy pierwszy współczynnik względem drugiego.

- Po ukryciu wiadomości wykonujemy odwrotną transformację DCT (IDCT), aby odzyskać obraz w przestrzeni pikseli.

Taka metoda jest stosunkowo odporna na stratną kompresję JPEG, ponieważ średnie częstotliwości nie są tak mocno kwantyzowane jak wysokie.

# Wykonanie ćwiczenia:

## 1. Przygotowanie obrazu

Wczytaliśmy przykładowy obraz w skali szarości i dopasowaliśmy jego rozmiar tak, aby jego wysokość i szerokość były podzielne przez 8.

## Użyte funkcje w skrypcie:

Podczas implementacji stworzyliśmy kilka dedykowanych funkcji, które dzielą cały proces na przejrzyste etapy:

- `load_image_grayscale_aligned(path)`
  Wczytuje obraz w skali szarości i przycina go tak, by jego rozmiary były podzielne przez 8 (wymagane przez DCT).
- `apply_dct_blocks(image)`
  Dzieli obraz na bloki 8x8 i wykonuje transformację DCT na każdym z nich. Zwraca współczynniki częstotliwościowe.
- `embed_message_koch_zhao(dct_image, bits, coeff_pair=(3, 4))`
  Osadza binarną wiadomość w obrazie modyfikując współczynniki DCT w każdej parze bloków zgodnie z metodą Kocha-Zhao.
- `apply_idct_blocks(dct_image)`
  Wykonuje odwrotną transformację DCT (IDCT) na blokach obrazu, przekształcając dane częstotliwościowe z powrotem na piksele.
- `extract_embedded_bits(dct_image, bit_count, coeff_pair=(3, 4))`
  Ekstrahuje wiadomość z obrazu, porównując odpowiednie pary współczynników i odczytując zakodowane bity.

### 2. Podział na bloki i transformacja DCT

Obraz został podzielony na bloki 8x8 pikseli. Dla każdego bloku wykonano dyskretną transformację kosinusową (DCT).

### 3. Wbudowanie wiadomości

Przygotowaliśmy komunikat binarny o przykładowej zawartości `1011001110`.
Następnie:

- Dla każdego bitu wybieraliśmy osobny blok obrazu.
- W każdej parze współczynników (na pozycjach 3 i 4 w tablicy blokowej) modyfikowaliśmy wartości zgodnie z metodą Kocha-Zhao.

Dzięki dodanym logom na konsoli było dokładnie widoczne, w którym bloku i jak zmieniały się współczynniki.

### 4. Odzyskiwanie obrazu

Wykonaliśmy odwrotną transformację DCT (IDCT) na zmodyfikowanych blokach, dzięki czemu otrzymaliśmy finalny obraz z ukrytą wiadomością.

### 5. Ekstrakcja wiadomości

Ponownie podzieliliśmy obraz na bloki i wykonaliśmy DCT. Następnie analizowaliśmy pary współczynników w każdym bloku, aby odczytać zakodowane bity.

Porównaliśmy pierwotną wiadomość z odzyskaną — rezultat był bardzo dobry, z drobnymi różnicami wynikającymi z operacji zaokrąglania i przybliżeń podczas transformacji.

## Podsumowanie:

W laboratorium zrealizowaliśmy pełny system steganograficzny oparty na metodzie Kocha-Zhao.
Zaimplementowaliśmy wszystkie kroki:

- przygotowanie obrazu,
- osadzanie danych,
- odczytanie ukrytej wiadomości.

Dzięki zastosowanym logom mogliśmy na bieżąco obserwować proces ukrywania i odczytu informacji.
Metoda okazała się stosunkowo prosta w implementacji, ale wymagała precyzyjnej manipulacji współczynnikami DCT.

## Użyte technologie:

- Python 3.11
- OpenCV (opencv-python)

## Efekty końcowe:

- Plik wejściowy: `house_bw.png`
- Plik wynikowy z ukrytą wiadomością: `stego_image.png`
- Konsolowe logi potwierdzające poprawne działanie algorytmu.