
Inżyniera Obrazów
laboratorium numer 4

Autor sprawozdania: Michał Dziedziak 263901
Imię i nazwisko prowadzącego kurs: dr inż. Jan Nikodem
Dzień i godzina zajęć: czwartek, 11:15 - 14:15

Spis treści

1 Temat laboratorium	3
2 Zadanie 1	3
2.1 Treść	3
2.2 Prezentacja wykonanego zadania	3
3 Zadanie 2	4
3.1 Treść	4
3.2 Prezentacja wykonanego zadania	5
4 Zadanie 3	6
4.1 Treść	6
4.2 Prezentacja wykonanego zadania	6
5 Zadanie 4	7
5.1 Treść	7
5.2 Prezentacja wykonanego zadania	7
6 Zadanie 5	9
6.1 Treść	9
6.2 Prezentacja wykonanego zadania	9

Spis rysункów

1 Zdjęcie konsoli podczas kodowania wiadomości	3
2 Zdjęcie z zakodowaną wiadomością	4
3 Zdjęcie konsoli podczas dekodowania wiadomości	4
4 Obrazy z kolejnymi wartościami nbits	5
5 Wykres zależności MSE od wartości nbits	5
6 Zdjęcie logów konsoli podczas procesu kodowania wiadomości	6
7 Obraz przed i po zakodowaniu wiadomości od pozycji 50000	6
8 Zdjęcie logów konsoli podczas procesu dekodowania wiadomości	7

9	Zdjęcie logów konsoli podczas procesu ukrywania obrazu w innym obrazie	7
10	Prezentacja obrazu przed i po zakodowaniu w nim innego obrazu	8
11	Zdjęcie logów konsoli podczas procesu odzyskiwania obrazu z innego obrazu	8
12	Prezentacja obrazu z ukrytym obrazem oraz odkodowanego obrazu	9
13	Zdjęcie logów konsoli podczas procesu odzyskiwania obrazu z innego obrazu	9
14	Prezentacja obrazu z ukrytym obrazem oraz odkodowanego obrazu	10

1 Temat laboratorium

Steganografia jest dziedziną nauki zajmującą się ukrywaniem komunikatów w jawnym medium. W ramach tego laboratorium, eksplorujemy techniki steganograficzne, koncentrując się na metodzie ukrywania informacji w obrazach poprzez modyfikację ich najmniej znaczących bitów (LSB - Least Significant Bit).

W praktyce używamy metody, która zapisuje ukrytą informację w obrazie poprzez modyfikację n-wybranych ostatnich bitów pikseli obrazu.

2 Zadanie 1

2.1 Treść

Pierwsze zadanie polegało o użyciu gotowych funkcji *hide_message* oraz *reveal_message* w celu zakodowania i odkodowania wiadomości w obrazie.

Zadanie zostało rozbudowane o menu, umożliwiające wybór czynności: *zakodowanie wiadomości*, *odkodowanie wiadomości*. Pozwala ono na zapisania obrazu z zakodowaną wiadomością do pliku, oraz odczytanie wiadomości z obrazu z pliku, który może pochodzić z innego programu.

2.2 Prezentacja wykonanego zadania

Kodowanie wiadomości

```
Zadanie 1
[0] - Zakoduj wiadomość
[1] - Odczytaj wiadomość
Wybierz opcje: 0
Podaj ścieżkę do obrazu: img2.jpg
Podaj tekst do ukrycia: To jest testowa wiadomość
Podaj liczbę najmłodszych bitów do użycia do zakodowania obrazka (liczba naturalna): 2
Wiadomość zakodowana pomyślnie. Długość: 216
Podaj ścieżkę do zapisania obrazu (jeżeli ma ukryte bity powinien być w formacie png): tmp.png
```

Rysunek 1: Zdjęcie konsoli podczas kodowania wiadomości



Rysunek 2: Zdjęcie z zakodowaną wiadomością

Dekodowanie wiadomości

```
Zadanie 1
[0] - Zakoduj wiadomość
[1] - Odczytaj wiadomość
Wybierz opcje: 1
Podaj ścieżkę do obrazu: tmp.png
Podaj długość wiadomości do odkodowania: 216
Podaj liczbę najmłodszych bitów do użycia do zakodowania obrazka (liczba naturalna): 2
Wiadomość po odkodowaniu z obrazu:
To jest testowa wiadomość
```

Rysunek 3: Zdjęcie konsoli podczas dekodowania wiadomości

3 Zadanie 2

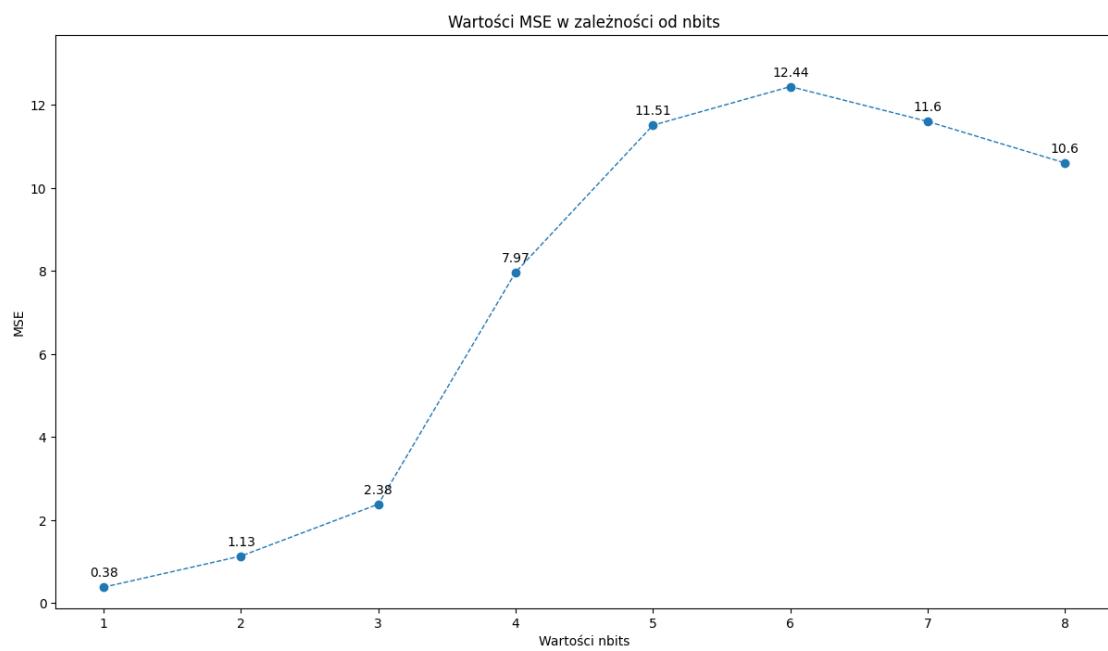
3.1 Treść

W zadaniu drugim należało zakodować w obrazie wiadomość o długości 75% liczby bajtów w obrazie. Następnie należało zmierzyć wartości MSE pomiędzy oryginalnym obrazem o obrazem z zakodowaną wiadomością dla wartości $nbits$ od 1 do 8.

3.2 Prezentacja wykonanego zadania



Rysunek 4: Obrazy z kolejnymi wartościami nbits



Rysunek 5: Wykres zależności MSE od wartości nbits

4 Zadanie 3

4.1 Treść

W zadaniu trzecim należało zmodyfikować funkcję `hide_message` i `reveal_message` tak, aby można było wybrać od którego miejsca ma być zapisywana wiadomość w obrazie.

Zadanie zostało rozbudowane o menu analogiczne do tego z zadania pierwszego.

4.2 Prezentacja wykonanego zadania

Kodowanie wiadomości

```
Zadanie 3
[0] - Zakoduj wiadomość
[1] - Odczytaj wiadomość
Wybierz opcje: 0
Podaj ścieżkę do obrazu: img2.jpg
Podaj pozycję od której ma być kodowana wiadomość (liczba naturalna): 50000
Podaj tekst do ukrycia: Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lor
Podaj liczbę najmłodszych bitów do użycia do zakodowania obrazka (liczba naturalna): 8
Wiadomość zakodowana pomyślnie. Długość: 4592
Podaj ścieżkę do zapisania obrazu (jeżeli ma ukryte bity powinien być w formacie png): tmp.png
```

Rysunek 6: Zdjęcie logów konsoli podczas procesu kodowania wiadomości



Rysunek 7: Obraz przed i po zakodowaniu wiadomości od pozycji 50000

Dekodowanie wiadomości

```
Zadanie 3
[0] - Zakoduj wiadomość
[1] - Odczytaj wiadomość
Wybierz opcje: 1
Podaj ścieżkę do obrazu: tmp.png
Podaj pozycję od której ma być dekodowana wiadomość (liczba naturalna): 50000
Podaj długość wiadomości do odkodowania: 4592
Podaj liczbę najmłodszych bitów użytych do zakodowania obrazka (liczba naturalna): 8
Wiadomość po odkodowaniu z obrazu:
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has bee
```

Rysunek 8: Zdjęcie logów konsoli podczas procesu dekodowania wiadomości

5 Zadanie 4

5.1 Treść

Zadanie czwarte polegało na zaimplementowaniu funkcjonalności ukrywania (i odzyskiwania) obrazu w innym obrazie.

W praktyce oznaczało to dodanie kroku, który spłaszczał piksele w obrazie do jednego wymiaru, a następnie ukrywał kolejne bity w najmniej znaczących bitach obrazu docelowego.

Analogicznie proces odzyskiwania obrazu polegał na odczytaniu ukrytych bitów i odpowiednią ich transformację do kanałów RGB.

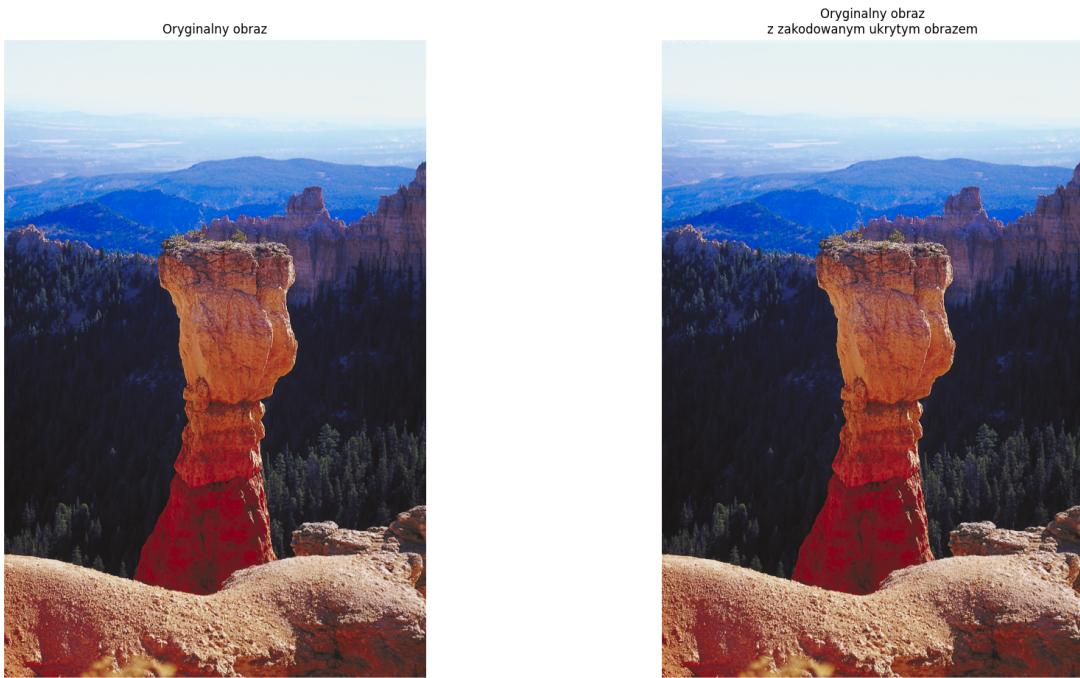
Zadanie zostało rozbudowane o menu analogiczne do tego z zadania pierwszego.

5.2 Prezentacja wykonanego zadania

Kodowanie obrazu

```
Zadanie 4
[0] - Zakoduj obraz
[1] - Odczytaj obraz
Wybierz opcje: 0
Podaj ścieżkę do głównego obrazu: img1.jpg
Podaj ścieżkę do pliku, który ma zostać ukryty: img2.jpg
Znaleziono plik img2.jpg
Podaj liczbę najmłodszych bitów do użycia do zakodowania obrazka (liczba naturalna): 4
Zakodowano obraz. Jego długość to: 777912
Podaj ścieżkę do zapisania obrazu (jeżeli ma ukryte bity powinien być w formacie png): tmp.png
```

Rysunek 9: Zdjęcie logów konsoli podczas procesu ukrywania obrazu w innym obrazie



Rysunek 10: Prezentacja obrazu przed i po zakodowaniu w nim innego obrazu

Dekodowanie obrazu

```
Zadanie 4
[0] - Zakoduj obraz
[1] - Odczytaj obraz
Wybierz opcje: 1
Podaj ścieżkę do obrazu z ukrytym obrazem: tmp.png
Podaj liczbę najmłodszych bitów użyty do zakodowania obrazka (liczba naturalna): 4
Podaj liczbę bajtów zakodowanego obrazu: 777912
```

Rysunek 11: Zdjęcie logów konsoli podczas procesu odzyskiwania obrazu z innego obrazu



Rysunek 12: Prezentacja obrazu z ukrytym obrazem oraz odkodowanego obrazu

6 Zadanie 5

6.1 Treść

W zadaniu piątym należało rozbudować odczytywanie obrazu z pliku o automatyczne wykrywanie końca wiadomości. W ramach zadania zaimplementowałem rozpoznawanie końca wiadomości poprzez sprawdzanie czy aktualnie przetwarzane bajty pokrywają się ze stopką końca pliku jpg:

$$\text{kod stopki} = 111111111011001_2 = FFD9_{16}$$

6.2 Prezentacja wykonanego zadania

Zadanie 5

Podaj scieżkę do obrazu z ukrytym obrazem. Ukryty obraz musi mieć format jpg: **tmp.png**
Podaj liczbę najmłodszych bitów użytą do zakodowania obrazka (liczba naturalna): **5**

Rysunek 13: Zdjęcie logów konsoli podczas procesu odzyskiwania obrazu z innego obrazu

Oryginalny obraz
z zakodowanym ukrytym obrazem



Ukryty obraz po odkodowaniu



Rysunek 14: Prezentacja obrazu z ukrytym obrazem oraz odkodowanego obrazu