Zaawansowane Przetwarzanie Obrazów
Czwartek 18:55-20:35
Kodek

Wojciech Adamek
226337

1 Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z metodami kodowania i dekodowania obrazów, a następnie próba napisania własnego kodeka. Metodyka zbliżona jest do zmodyfikowanych kodeków JPEG oraz JPEG2000. Ćwiczenie zostało wykonane na obrazie:

Rysunek 1: Obraz pierwotny

2 Przebieg ćwiczenia

2.1 Struktura Kodeka

Ćwiczenie składa się z dwóch skryptów - skrypt kodujący obraz rastrowy wejściowy o rozszerzeniu .tiff i wielkości 57380 KB oraz skrypt dekodujący, wykonujący operacje odwrotne do kodowania. Po zakodowaniu i zdekodowaniu obrazu powinien on nie utracić lub utracić bardzo mało informacji.

Kodek został zrealizowany w następujących krokach:

- Odczytanie obrazu rastrowego i przekonwertowanie kolorów na schemat YCbCr
- Podział obrazu na bloki [8x8] i zastosowanie transformacji falkowej kosinusowej na każdym z nich
- Kwantyzacja współczynników każdego z bloków za pomocą maski:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- Konwersja wyników z liczb zmiennoprzecinkowych na całkowite

2.2 Wyjaśnienie

Dobrana maska powoduje utratę dużej części informacji - zachowuje tylko 10 współczynników z każdego bloku 64. Wiążę się to z gorszym efektem końcowym, ale bardzo dużym spadkiem wagi pliku.

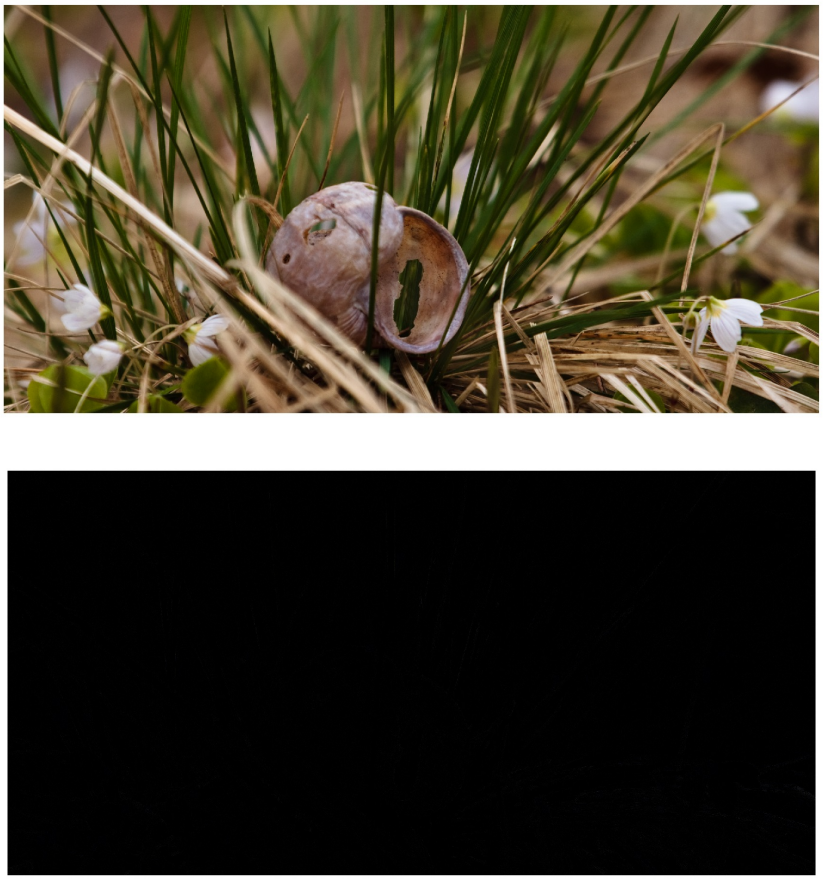
Transformacja kosinusowa wymaga dokładności liczb zmiennoprzecinkowych z podwójna precyzją (typ double). Zapis liczb tego typu jest jednak nieoptymalny, ponieważ zajmuje dużo pamięci. Żeby zminimalizować zużycie pamięci, przed zapisem pliku wybierany jest typ konwersji - 8-bitowa, 16-bitowa, 32-bitowa lub 64-bitowa liczba całkowita ze znakiem(int), ponieważ wyniki transformaty przyjmują też wartości ujemne. W przypadku tego obrazu przyjmują one wartości od około 6 do -3, a większość z nich oscyluje wokół zera. Dlatego nieefektywna jest bezpośrednia konwersja poprzez przybliżanie, a lepszym rozwiązaniem jest rozpięcie przybieranych wartości na cały zakres wybranego typu integer. Im więcej bitów w wybranym typie, tym większa dokładność odtwarzania tych liczb podczas dekodowania obrazu.

3 Wyniki

Dla porównania otrzymanych wyników: Obraz zakodowany automatycznie w formacie .jpg waży 483 KB, a jego błąd między obrazem rastrowym wynosi 0,24% (Błąd ten może być niepoprawny, ponieważ nastąpiła konieczność dwukrotnego pomniejszenia obrazu rastrowego, aby móc je ze sobą porównać).

3.1 Konwersja do typu 64-bitowego (int64)

- Wielkość pliku: 32298 KB, Poprawa około 177%.
- Błąd wynosi 1,15% (Różnica między obrazem zdekodowanym a wejściowym obrazem rastrowym)



Rysunek 2: Obraz zakodowany w dokładności 64-bitowej oraz różnica od obrazu rastrowego

3.2 Konwersja do typu 32-bitowego (int32)

- Wielkość pliku: 15910 KB, Poprawa około 361%.
- Błąd wynosi 1,15%
- Efekt końcowy identyczny co z konwersją do liczb 64-bitowych.

3.3 Konwersja do typu 16-bitowego (int16)

- Wielkość pliku: 7088 KB, Poprawa około 809%.
- Błąd wynosi 1,15%
- Również brak różnic. Efekt taki sam jak dla poprzednich konwersji.

3.4 Konwersja do typu 8-bitowego (int8)

- Wielkość pliku: 984 KB, Poprawa około 5831%.
- Błąd wynosi 1,43%
- Efekt końcowy bardzo zbliżony poprzednim konwersjom.

4 Wnioski

Efektem końcowym jest plik o formacie .jwa, o wielkości 984 KB i 1,43% błędu czyli około 1,43% informacji obrazu wejściowego zostało utraconych lub zmodyfikowanych. Jest to waga około 204% większa i około 596% więcej informacji utraconych od formatu .jpg. Wiążę się to z kwantyzacją wybraną maską gdzie tracone są informacje oraz brakiem dalszego kodowania np. Huffmana. Nieudane próby ich implementacji zakończyły się gorszymi efektami niż opisana wyżej metoda.

Warto wspomnieć, że plik .jpg(dostarczony w materiałach do ćwiczenia), do którego został porównany efekt końcowy jest dwa razy mniejszy. Po zmniejszeniu dwukrotnie pliku .jwa otrzymano wagę 868 KB poprawa około 13%.

Rysunek 3: Obraz zakodowany w dokładności 8-bitowej, Efekt końcowy