Politechnika Krakowska
Wydział Inżynierii Elektrycznej i Komputerowej
Katedra Automatyki i Informatyki

Systemy Multimedialne – Projekt

Grupa 12

Autorzy:
Piotr Bujak
Weronika Bober
Chrabąszcz Julia
Mateusz Chorąży

Specyfikacja

Treść zadania:

Należy stworzyć **specyfikację** rastrowego pliku graficznego rejestrującego obraz kolorowy oraz w skali szarości z obsługą trybu **16-bitowego** (**RGB565+dithering**) i **24-bitowego** (**RGB888** oraz **YUV888**), korzystających z:
a) algorytmu **predykcji** (filtracji) typu **1,**
b) kompresji **bezstratnej RLE**,
c) kompresji **stratnej** (**transformata DCT** + **podpróbkowanie składowych**),

W ramach projektu należy także napisać aplikacje, które zgodnie ze stworzoną specyfikacją dokonają konwersji z pliku BMP do nowego rodzaju pliku graficznego oraz z nowego rodzaju pliku do formatu BMP z możliwością wskazania przez użytkownika trybu koloru i przestrzeni barw, algorytmu predykcji oraz rodzaju kompresji (stratnej lub bezstratnej).

Budowa Nagłówka

Nagłówek jest wykorzystywany do identyfikacji pliku oraz do odczytania metadanych dotyczących obrazka. Dodatkowo w nagłówku zamieszczone są dane pozwalające na weryfikację czy obrazek nie jest uszkodzony. Na nagłówek pliku w formacie SM25 składają się:

Nazwa	Offset	Rozmiar	Wartości i przeznaczenie
Identyfikator	00	4 bajty	**'SM25'** służy do identyfikacji pliku
Szerokość obrazka	04	2 bajty	**0-640** Określa szerokość obrazka w pikselach
Wysokość obrazka	06	2 bajty	**0-400** Określa wysokość obrazka w pikselach
Tryb	08	1 bajt	**0-2** 16-bitowy RGB565 z ditheringiem, 24-bitowy RGB888 24-bitowy YUV888
Predykcja	09	1 bajt	**0-1** Bez predykcji Predykcja typu 1
Kompresja	10	1 bajt	**0-1** Bezstratna RLE Stratna DCT + podpróbkowanie
Rozmiar danych	11	4 bajty	**0-4294967296** Określa rozmiar danych obrazu

TRYB KOLORU

Format SM25 obsługuje trzy tryby reprezentacji koloru, które użytkownik może wybrać podczas konwersji z pliku BMP. Każdy tryb określa sposób kodowania informacji o kolorze pojedynczego piksela oraz zastosowane techniki przetwarzania.

Tryb 0: 16-bitowy RGB565 z ditheringiem – W tym trybie każdy piksel obrazu jest kodowany za pomocą 16 bitów (2 bajtów), co pozwala na reprezentację ograniczonej palety kolorów w porównaniu do pełnego 24-bitowego RGB. Rozkład bitów na składowe koloru jest następujący: 5 bitów dla składowej czerwonej (R), 6 bitów dla zielonej (G) i 5 bitów dla niebieskiej (B). Aby zminimalizować widoczne błędy kwantyzacji (posteryzację) wynikające z takiej redukcji głębi koloru, stosowany jest algorytm ditheringu z użyciem stałej matrycy Bayera o rozmiarze 4x4. Wartości z tej matrycy (zdefiniowanej w kodzie projektu) są wykorzystywane do modyfikacji oryginalnych składowych RGB przed ich redukcją do 5/6 bitów, co rozprasza błąd kwantyzacji i tworzy iluzję większej liczby odcieni. Dane pikseli są zapisywane w kolejności wierszy.

Tryb 1: 24-bitowy RGB888 – Jest to standardowy, bezstratny tryb reprezentacji koloru, w którym każdy piksel opisany jest trzema bajtami. Każda składowa – czerwona (R), zielona (G) i niebieska (B) – ma 8-bitową głębię, co daje łącznie 256 możliwych wartości na kanał i teoretycznie około 16,7 miliona możliwych kolorów. Kolejność zapisu bajtów w pliku dla pojedynczego piksela to: najpierw bajt składowej R, potem G, na końcu B. Ten tryb nie stosuje żadnej kompresji ani ditheringu na etapie kodowania koloru; przechowuje oryginalne wartości pikseli z dokładnością do bajta na kanał.

Tryb 2: 24-bitowy YUV888 – W tym trybu kolor jest reprezentowany w przestrzeni barw YUV, która oddziela informację o jasności (luminancji Y) od informacji o kolorze (chrominancji U i V). Każda z trzech składowych (Y, U, V) kodowana jest na 8 bitach, więc łączny rozmiar na piksel to także 3 bajty. Konwersja z oryginalnego RGB do YUV odbywa się według standardowych wzorów implementowanych w funkcjach projektu (getYUV w SM2025-Barwy.cpp). Kolejność zapisu bajtów w pliku to: Y, U, V. Przestrzeń YUV jest często używana w kompresji stratnej, ponieważ ludzkie oko jest mniej czułe na zmiany koloru niż na zmiany jasności, co pozwala na późniejsze podpróbkowanie składowych U i V bez znacznej utraty jakości wizualnej. W trybie pełnym 888 nie stosuje się jednak takiego podpróbkowania – jest ono realizowane dopiero w połączeniu z kompresją stratną DCT.

DANE OBRAZU

Dane obrazu w formacie SM25 zaczynają się od offsetu 15, bezpośrednio po nagłówku. Ich struktura zależy od trybu koloru i kompresji, ale zawsze zapisywane są wierszami, od lewego górnego rogu do prawego dolnego.

Jeśli wybrano typ predykcji 0 (brak), dane są zapisywane w swojej oryginalnej postaci po konwersji do wybranego trybu koloru. Jeśli wybrano typ predykcji 1, dane są najpierw przetwarzane tym filtrem przed zapisem. Filtr ten działa osobno na każdej składowej koloru (R,G,B lub Y,U,V). Dla każdego piksela (oprócz pierwszego w wierszu) obliczana jest różnica między jego wartością a wartością piksela bezpośrednio po lewej stronie (w tym samym wierszu). Do pliku zapisywana jest właśnie ta różnica. Pierwszy piksel w każdym wierszu zapisywany jest w postaci nieprzetworzonej (jako wartość bezwzględna), ponieważ nie ma lewego sąsiada.

Dla trybu 0 (16-bit RGB565 z ditheringiem), każdy piksel po przetworzeniu (i ewentualnej filtracji) zajmuje 2 bajty. Format to 5 bitów R, 6 bitów G, 5 bitów B. Przed redukcją bitów stosowany jest dithering z matrycą Bayera 4x4.

- Dla trybu 1 (24-bit RGB888), każdy piksel to 3 bajty w kolejności R, G, B.
- Dla trybu 2 (24-bit YUV888), każdy piksel to 3 bajty w kolejności Y, U, V, po konwersji z RGB.

Jeśli zastosowano kompresję RLE (bezstratną), to na danych po ewentualnej filtracji działa algorytm RLE. Szuka on sekwencji identycznych pikseli. Gdy znajdzie dwa lub więcej takich samych, zapisuje liczbę powtórzeń, a potem tylko jedną wartość tego piksela. Gdy trafi na sekwencję różnych pikseli, zapisuje znacznik 0, liczbę pikseli w sekwencji, a potem ich wszystkie wartości.

Jeśli zastosowano kompresję DCT (stratną), obraz (po konwersji do YUV, jeśli był w RGB) jest dzielony na bloki 16x16 pikseli. Na każdym bloku wykonywana jest transformata DCT, a jej współczynniki są kwantyzowane (np. niektóre zerowane). Do pliku zapisywane są te skwantyzowane współczynniki DCT dla każdego bloku, najpierw wszystkie bloki składowej Y, potem U, na końcu V.

KOLEJNOŚĆ ZBIERANIA DANYCH

Kolejność zbierania i zapisu danych obrazu w formacie SM25 jest ściśle określona i przebiega według następującej sekwencji, która musi być identycznie przestrzegana zarówno podczas kodowania (zapisu), jak i dekodowania (odczytu) pliku:

Dane są zbierane i przetwarzane wiersz po wierszu, zaczynając od górnego lewego rogu obrazu. W obrębie każdego wiersza piksele są przetwarzane od lewej do prawej strony. Po zakończeniu przetwarzania ostatniego piksela w wierszu, algorytm przechodzi do pierwszego piksela kolejnego wiersza poniżej, kontynuując aż do osiągnięcia prawego dolnego rogu obrazu.

W przypadku trybów 24-bitowych (RGB888 i YUV888) dane dla każdego pojedynczego piksela są zbierane i zapisywane w postaci trzech kolejnych bajtów. Dla trybu RGB888 kolejność tych bajtów jest zawsze: najpierw składowa czerwona (R), następnie zielona (G), na końcu niebieska (B). Dla trybu YUV888 kolejność to: luminancja (Y), chrominancja U, chrominancja V.

Jeśli w nagłówku wybrano predykcję typu 1, proces zbierania danych jest modyfikowany. Dla każdego przetwarzanego piksela (oprócz pierwszego w każdym wierszu) nie zapisuje się jego bezwzględnej wartości, lecz różnicę między tą wartością a wartością piksela znajdującego się bezpośrednio po jego lewej stronie, który został już wcześniej przetworzony w tym samym wierszu. Pierwszy piksel w każdym wierszu nie ma lewego sąsiada, więc jego wartość jest zapisywana w formie nieprzetworzonej (bezwzględnej). Predykcja jest stosowana osobno i niezależnie do każdej z trzech składowych koloru (R, G, B lub Y, U, V), co oznacza, że algorytm przechodzi trzykrotnie przez cały obraz – raz dla każdej płaszczyzny składowej.

Gdy zastosowano kompresję bezstratną RLE, zbieranie danych do kompresji następuje po ewentualnej fazie predykcji. Algorytm RLE analizuje ciąg pikseli dokładnie w kolejności, w jakiej zostały zebrane (wiersz po wierszu, od lewej do prawej). Poszukuje on ciągłych sekwencji identycznych pikseli. Gdy taka sekwencja zostanie znaleziona, zamiast zapisywać każdy piksel osobno, algorytm zbiera informację o długości tej sekwencji (liczbie powtórzeń) i pojedynczej wartości reprezentatywnego piksela. W przypadku napotkania sekwencji nieidentycznych pikseli, algorytm zbiera długość tej sekwencji, a następnie wartości wszystkich pikseli w niej zawartych. Liczniki i wartości kompresji są zapisywane do pliku w dokładnie tej samej kolejności, w jakiej zostały zebrane podczas skanowania obrazu.

W przypadku kompresji stratnej DCT, kolejność zbierania danych jest bardziej złożona i wielowarstwowa. W implementacji referencyjnej, algorytm DCT działa wyłącznie na skali szarości (wartościach luminancji). Dane wejściowe dla DCT są pobierane bezpośrednio jako wartości jasności pikseli. Proces zbierania danych dla DCT rozpoczyna się od podziału obrazu (lub jego fragmentu w skali szarości) na bloki o stałym rozmiarze 16×16 pikseli. Bloki te są zbierane i przetwarzane w kolejności wierszowej bloków. Oznacza to, że algorytm zaczyna od bloku w lewym górnym rogu obrazu, następnie przechodzi do bloku bezpośrednio po prawej stronie, i kontynuuje w prawo aż do końca pierwszego poziomego rzędu bloków. Po zakończeniu pierwszego rzędu, algorytm przechodzi do bloku znajdującego się najbardziej po lewej stronie, w rzędzie bezpośrednio poniżej, i kontynuuje w prawo. Ten proces powtarza się, aż wszystkie bloki zostaną zebrane. Wewnątrz każdego pojedynczego bloku 16×16, wartości pikseli są zbierane w standardowej, przestrzennej kolejności – wiersz po wierszu od góry do dołu, a w każdym wierszu od lewej do prawej strony. Ta sekwencja 256 wartości (16×16) tworzy dokładnie tę dwuwymiarową macierz.

Podczas zapisu do pliku, wszystkie zebrane i przetworzone dane są zapisywane w dokładnie tej samej sekwencji, w jakiej zostały zebrane. Ta deterministyczna i niezmienna kolejność jest kluczowa dla poprawnego działania dekodera, który podczas odczytu pliku musi odtworzyć identyczny proces, tyle że w odwrotnej kolejności operacji: najpierw dekompresuje dane, następnie odwraca predykcję, a na końcu konwertuje wartości do finalnej przestrzeni kolorów wyświetlanego obrazu.