Dokumentácia k projektu do predmetu Kódovanie a kompresia dát: Konverzia obrazového formátu GIF na BMP

Autor: Filip Gulán (xgulan00)

1 Úvod

Úlohou tohoto projektu bolo vytvoriť v jazyku C/C++ knižnicu a aplikáciu pre prevod súboru grafického formátu GIF (Graphics Interchange Format) na súbor grafického formátu BMP (Microsoft Windows Bitmap). Pre súbor formátu GIF sa predpokladalo zakódovanie pomocou metódy LZW (Lempel-Ziv-Welch). Výstupný súbor vo formáte BMP malo byť možné zobraziť napríklad v prehliadači IfranView či iným podobným.

2 Popis riešenia

Samotný popis riešenia je rozdelený na niekoľko celkov, ktoré sa podrobnejšie venujú danej časti riešenej úlohy. Najprv sa popisuje samotná aplikácia, potom knižnica, na ktorej aplikácia stojí a následne metódy dekódovania a kódovania.

2.1 Popis aplikácie

Aplikácia, ktorá využíva knižnicu *gif2bmp,* slúži predovšetkým na spracovanie argumentov príkazového riadku a na spustenie samotnej konverzie podľa týchto parameterov. Konverzia sa uskutočňuje pomocou knižnej funkcie *gif2bmp()*. Vstupný GIF obrázok je možné špecifikovať prepínačom *-i <obrázok>*. V prípade chýbajúceho parametra sa berie vstup zo *stdin*. Umiestnenie a meno výstupného BMP obrázku je možné špecifikovať prepínačom *-o <obrázok>*, a v prípade, že prepínač chýba, tak výstup konverzie je zobrazený na *stdout*. Prepínač *-l <súbor>* slúži na špecifikovanie umiestnenia a mena log súboru, ktorý obsahuje login riešiteľa projektu a veľkosť obrazových dát formátu GIF a BMP. Veľkosť obrazových dát je v prípade GIF počítaná iba podľa dát zakódovaných pomocou LZW (bez LZW minimálnej veľkosti kódu a hlavičiek GIF). V prípade BMP je veľkosť obrázkových dát počítaná podľa jednotlivých pixelov a zarovnania, teda bez hlavičky *BMP* a hlavičky *DIB*. Všetky súbory, či už pre vstupný/výstupný obrázok, alebo log sa otvárajú práve v aplikácií a do funkcie sa predávajú už iba ukazatele. *Ifstream** v prípade, že ide o vstupný súbor, *ofstream** v prípade, že ide o výstupný. Okrem toho, vstupný obrázok pre korektné fungovanie knižnice, musí byť otvorený v móde *ios::binary*. Posledný prepínač *-h* slúži na zobrazenie nápovedy. Prepínač nesmie byť kombinovaný s ostatnými prepínačmi.

2.1 Popis knižnice *gif2bmp*

Knižnica *gif2bmp* slúži na konverziu obrázkov z formátu GIF do formátu BMP. Obsahuje jedinú funkciu viditeľnú pre iné knižnice alebo programy, a to funkciu *gif2bmp()*. Za parametre na svoj vstup berie štruktúru typu *tGIF2BMP*, ktorá je v tejto funkcii naplnená veľkosťou obrazových dát GIF a BMP. Parametre typu *ifstream** a *ofstream** slúžia pre čítanie dát z GIF súboru a pre zapísanie dát do BMP súboru. Funkcia v prvom rade získa hexadecimálne dáta obrázku GIF, ktoré následne predá funkcii *decodeGIF()* a tá vráti vektor pixelov (vektor reťazcov, každý reťazec reprezentuje

farbu s farebnými kanálmi v poradí ako to vyžaduje BMP t.j. : modrá, zelená, červená). Vektor pixelov je následne predaný funkcii *makeBMP()*, ktorá vytvorí BMP formát. BMP dáta sú následne podľa potreby zobrazené na výstup, alebo uložené do súboru.

2.1 Dekódovanie GIF obrázku

Dekódovanie a získanie obrázkových dát, ako už bolo spomenuté, zabezpečuje funkcia *decodeGIF()*. Funkcia postupne spracováva jednotlivé hlavičky a bloky GIF súboru. V prvom rade sa zisťuje, či sa jedná o GIF súbor, testuje sa prítomnosť na GIF signatúru. Hneď po GIF signatúre sa spracuje verzia GIF súboru. Následne pokračuje popis logickej obrazovky, ktorý sa nielen spracuje, ale získajú sa aj dôležité dáta, ako je príznak, ktorý určuje či je použitá globálna tabuľka farieb a veľkosť tejto tabuľky. Pokiaľ je príznak globálnej tabuľky nastavený na *true*, tak ihneď po popise logickej obrazovky nasleduje globálna tabuľka farieb. Globálna tabuľka je spracovaná a farby sú následne uložené do vektoru reťazcov. Po globálnej tabuľke nasleduje spracovávanie ostatných blokov, ktoré sa deje v cykle. Tento cyklus beží, dokiaľ sa nenarazí na hlavičku trailer, ktorá reprezentuje koniec GIF súboru. V prípade že sa narazí v tomto cykle na bloky Graphic control extension, Application extension, Comment extension alebo Plain Text extension, tak sa tieto bloky jednoducho iba spracujú, pretože pre konverziu neobsahujú žiadne užitočné dáta. Dôležité sú bloky popis obrázku, lokálna tabuľka farieb a samotné dáta obrázku. Popis obrázku je spracovaný a sú z neho získané taktiež dôležité dáta, ako to, či je použitá lokálna tabuľka farieb, či je obrázok prekladaný alebo šírka a výška obrázku. Lokálna tabuľka farieb je prítomná iba v prípade nastavenia príznaku získaného z popisu obrázku. Samotné dáta sú potom kódované pomocou LZW a dekódujú sa vo funkcii *decodeLZW()*. Táto funkcia vracia vektor pixelov (reťazcov). Nakoniec, sa v prípade nastaveného príznaku prekladania vykoná funkcia *interlacedToNoninterlaced()*, ktorá prekladané riadky obrázku konvertuje do ich neprekladanej podoby. [1]

2.2 Dekódovanie LZW dát

Dekódovanie LZW dát prebieha vo funkcii *decodeLZW()*. Algoritmus potrebuje pre svoje vykonávanie poznať minimálnu počiatočnú veľkosť LZW kódu. Fungovanie algoritmu môže byť popísané nasledovne: Inicializuje sa LZW tabuľka, t.j. vložia sa do nej farby z globálnej alebo lokálnej tabuľky farieb, a k tomu *Clear code* a *End of Information*. Súčasne sa veľkosť LZW kódu nastaví na počiatočnú veľkosť LZW +1. Zoberie sa prvý kód (prvý, ktorý ale nie je *Clear code*). Kód reprezentuje index do LZW tabuľky. Indexom sa teda pristúpi k dátam LZW tabuľky a získa sa pixel, ktorý je následne vložený do vektoru pixelov. Následne pokračuje úsek algoritmu vykonávaný v cykle, ktorý pracuje pokiaľ sa nenarazí na index *End of Information*. V každej iterácii cyklu sa zoberie podľa veľkosti kódu index a podľa toho, či je index v tabuľke alebo nie je, sa vykonajú dané operácie. V prípade, že sa index v tabuľke nachádza, pridá sa do LZW tabuľky predchádzajúca hodnota kódu + prvý *byte* aktuálnej hodnoty, a následne sa aktuálna hodnota kódu/pixelu uloží do vektoru pixelov. V prípade, že sa index v LZW tabuľke nenachádza, tak sa do nej pridá hodnota prechádzajúceho kódu + prvý *byte* predchádzajúcej hodnoty, a rovnako aj do tabuľky pixelov. V prípade, že sa narazí na index *Clear code*, tak je LZW tabuľka znovu inicializovaná a LZW kód je nastavený znovu na počiatočnú hodnotu LZW +1. V prípade, že index vkladanej hodnoty do tabuľky LZW je rovný maximu pre veľkosť kódu a zároveň veľkosť kódu je menšia ako 12, tak sa veľkosť kódu zväčší o jedna. [2]

2.3 Kódovanie obrázku BMP

Kódovanie pixelov do formátu BMP verzia 3 sa deje vo funkcii *makeBMP()*, ktorej je potrebné predať výšku a šírku obrázka a vektor pixelov. Funkcia následne tvorí postupne reťazec. Najprv sa pridávajú hlavičky, presnejšie hlavička *BMP* a hlavička *DIB*, ktorá obsahuje informáciu práve o šírke a výške obrázku a na akom *offsete* začínajú samotné dáta obrázku. Po týchto hlavičkách už nasleduje pridávanie samotných pixelov. Okrem pixelov sa musí vkladať po každom riadku ešte zarovnanie. [3]

3 Záver

Odovzdaný zip archív obsahuje okrem požadovaných zdrojových súborov a dokumentácie aj obrázky, na ktorých bola celá funkcionalita testovaná. Preklad projektu je možný pomocou priloženého *Makefile* súboru. Taktiež pomocou príkazu *make tests* je možné spustiť testovaciu konverziu všetkých obrázkov. Obrázky vo formáte BMP spolu s log súbormi je potom možné nájsť v zložke *out*.

Program dokáže konvertovať GIF obrázky vo verzií 87a a 89a. V prípade animácií konvertuje do BMP súboru iba prvý snímok. Program bol riadne otestovaný na operačnom systéme Ubuntu 16.04 LTS a na operačnom systéme CentOS. Behom testovania sa nevyskytli žiadne chyby a všetky testy dopadli úspešne.

Referencie

[1] *Graphics Interchange Format Programming Reference: Cover Sheet for the GIF89a Specification* [online]. Columbus, Ohio: CompuServe Incorporated, 1990 [cit. 2017-03-04]. Dostupné z: https://www.w3.org/Graphics/GIF/spec-gif89a.txt

[2] GIF File Format Specification. *Daub* [online]. [cit. 2017-03-04]. Dostupné z: https://www.daubnet.com/en/file-format-gif

[3] Microsoft Windows Bitmap File Format Summary. *File Format Info* [online]. [cit. 2017-03-04]. Dostupné z: http://www.fileformat.info/format/bmp/egff.htm