Porovnání komprimačních metod grafických formátů z hlediska míry kvality obrazu

Comparison of compression methods for graphic formats in term of picture quality

Michal Sládek

Bakalářská práce
2010

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2009/2010

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal SLÁDEK**
Osobní číslo:	**A06203**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Téma práce:	**Porovnání komprimačních metod grafických formátů z hlediska míry kvality obrazu**

Zásady pro vypracování:

1. **Seznamte se s používanými komprimačními metodami u obrázků.**
2. **Porovnejte komprimační metody z hlediska zachování kvality obrázku.**
3. **Dále porovnejte metody komprimace s ohledem na velikost komprimovaného souboru.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Vlček, K. : Komprese a kódování zabezpečení, edice BEN, Praha, 2004.**
2. **Prchal, J., Šimák, B.: Digitální zpracování signálu v telekomunikacích. Vydavatelství ČVUT, Praha, 2001.**
3. **Levoy, M.: Polygon-Assisted JPEG and MPEG Compression of Synthetic Images, Center of integrated systems, Stanford University, 1995**
4. **Savakis, A.E.: Evaluation of lossless compression methods for gray scale document images, Department of Computer Engineering, Rochester Institute of Technology [http://www.ce.rit.edu/ásavakis/papers/ICIP00_savakis.pdf]**
5. **Grigorov, L., Abolmaesumi, P. : Region-based method for visually lossless image compression, Queen's University, Kingston, Ontario, Kanada [http://research.cs.queensu.ca/home/grigorov/ICIP04-first.pdf]**

Vedoucí bakalářské práce: **Ing. Karel Perůtka, Ph.D.**
Ústav řízení procesů

Datum zadání bakalářské práce: **5. března 2010**

Termín odevzdání bakalářské práce: **1. června 2010**

Ve Zlíně dne 5. března 2010

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Předložená bakalářská práce se zabývá komprimací obrázků a její nejdůležitější částí a to je porovnání kvality a velikosti výsledného souboru u jednotlivých metod komprimace. V praxi se dnes s komprimací obrázků setkáváme téměř na každém kroku, ať už na internetu, při fotografování nebo ostatních činnostech u kterých by nás to na první pohled nenapadlo.

Přestože existuje mnoho metod komprimace, každá není vhodná na jakýkoliv obrázek. U některých můžeme nést riziko ztráty důležitých informací a u některých je mnoho informací přebytečných. Je potřebné vyčlenit informaci přebytečnou, bez které se můžeme obejít a informaci důležitou, která je stěžejní a obejít bez ní nemůžeme. Je to vlastně činnost, se kterou se setkáváme jak v normálním životě, tak u mnohého dalšího. Mezi jedno z nich patří právě i komprimace obrazových informací.

Klíčová slova:

Komprimace, fotografování, riziko ztráty, obrazové informace

ABSTRACT

The bachelor thesis presented is concerned with compression of pictures and its most important part, which is the comparison of the quality and size of the final file, referring to the differences between the compression methods. In everyday life, we encounter the compression of pictures almost everywhere, either on the Internet, when taking photographs, or with other activities that do not explicitly seem to use this method.

Although there are a high number of compression methods, not all of them are convenient for any picture. With some of the methods, we can face the risk of loss of important information, while with some of them, there is a lot of superfluous information. It is essential to distinguish the superfluous information, that we can manage without, and the important information, that is of major significance and that we cannot manage without. In fact, it is an operation that we encounter in our everyday life, as well as on other occasions. One of these is also the compression of picture information.

Keywords:
Compression, taking photographs, the risk of information loss, picture information

Děkuji vedoucímu své bakalářské práce Ing. Karlu Perůtkovi, Ph.D. za jeho odborné vedení, poskytování rad a čas, který mi věnoval.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….……

podpis diplomanta

OBSAH

ÚVOD

Tato bakalářská práce se zabývá problematikou komprimace grafických formátů. Konkrétně jsou to komprimační metody a jejich porovnání z hlediska zachování kvality obrázku a její vliv na velikost souboru.

V oblasti informatiky se od samých počátků řešil problém s nedostatkem paměťového prostoru a datové propustnosti. Hlavní důvod byl ten, že paměťový prostor byl drahý, a tak se s ním muselo šetřit a přenosová kapacita byla malá. Bylo známo, že obrázky a ostatní data obsahují nepotřebné části a ty je možno v nějakém přesném řádu „vypustit" a obrázek tak zmenšit, neboli ho komprimovat. Daný obrázek nám tedy bude zabírat méně paměťového místa a stejně se neznehodnotí jeho informační hodnota, následně bude možno zpětným dekomprimováním obnovit původní obrázek. Začaly se tedy vymýšlet různé komprimační techniky a algoritmy, které by obrázky a ostatní data co nejvíce zmenšily, tedy zkomprimovaly.

V dnešní době je paměťová kapacita záznamových medií obrovská. Ve srovnání s dobou například před 15 lety a každým rokem stoupá. V přepočtu na peněžní jednotky je jeden bajt čím dál levnější, a tak se může zdát, že komprimační algoritmy již zdaleka nejsou tak potřebné jako v dobách nedávno minulých. Opak je ale pravdou vezmeme-li si například rozlišení digitálních fotoaparátů dnes a před lety. Datové velikosti souborů s obrázky, tak bychom se bez komprimace jen stěží obešli. Užití komprimace obecně je velice široké, komprimovat se dají téměř jakákoliv data, od textu až po nejrůznější složitá videa. V této práci se ale budeme zabývat komprimací obrázků.

S nárůstem výpočetního výkonu přibyla možnost komprimovat a dekomprimovat data takřka v reálném čase, uživatel si tedy ani nemusí uvědomovat, že data, se kterými právě pracuje, nebo je využívá, jsou na paměťovém mediu uloženy v zkomprimované podobě. Příkladem může být klasické fotografování na digitálním fotoaparátu. Obrázky se při ukládání přímo komprimují.

Celé odvětví komprimace je velice rozsáhlé a existuje nepřeberné množství algoritmů využívaných ke komprimaci dat. Každý, kdo vyvíjí komprimační programy, se snaží buďto stávající algoritmy upravit tak, aby byly rychlejší a dosahovaly většího stupně komprimace anebo se snaží vyvinout nové. Takto vznikají další a další verze komprimačních algoritmů. V této práci se zaměřím na algoritmy používané ke komprimaci obrázků případně na jejich předchůdce, ze kterých se vyvinuly.

I. TEORETICKÁ ČÁST

1 DRUHY A ROZDĚLENÍ KOMPRIMACÍ

1.1 Úvod do komprimace, základní rozdělení

Komprimace je speciální druh kódování dat za účelem zmenšení jejich objemu odstraněním nadbytečné informace (redundance). Komprimační algoritmus definuje přesný postup, jak toho docílit. Komprimační algoritmy dokážou určité obrazové informace vypouštět a tím tato data zkomprimovat, tedy výrazně snížit velikost výsledného datového souboru. Datová komprimace má tedy vliv na velikost obrázku, nikoliv však na jeho rozměry. Obecně rozdělujeme komprimaci na bezztrátovou a ztrátovou.

1.1.1 Bezztrátová komprimace

Při bezztrátové komprimaci dokážeme rekonstruovat data do původní podoby, v jaké se nacházely před komprimací. Používá se především tam, kde nemůžeme připustit jakoukoli ztrátu dat. Komprimace pracuje principiálně tak, že program, který ji provádí, se snaží najít v souboru nějaké vztahy nebo posloupnosti, kterých může využít [1]. Například má-li v souboru řadu znaků AAAAAABBBBBBCCCCCC, tak ji zkomprimuje na 6xA+6xB+6xC a tím ušetří 7 znaků. Při komprimaci obrazu obsahujícího geometrické objekty není ukládání prováděno do bitmapy, ale identifikují se jednotlivé objekty. Ukládají se pouze typ objektu a jeho pozice. Příkladem mohou být grafické programy AutoCad nebo Corel. Žádný komprimační program nedokáže bezztrátovou metodou komprimovat všechny soubory tak, aby vždy došlo ke komprimaci dat. Existuje vždy taková kombinace vstupních dat, kterou nebude schopen komprimační algoritmus zkomprimovat ani o jediný bit. Předností bezztrátové komprimace je možnost získání přesného duplikátu původního souboru, nevýhodou pak nízký poměr komprimace. V některých případech se může stát, že výsledný soubor je větší než původní.

1.1.2 Ztrátová komprimace

Při ztrátové komprimaci na rozdíl od bezztrátové nemůžeme přesně rekonstruovat data do původní podoby. Ztrátovou komprimaci nemůžeme využít, potřebujeme-li zachovat přesný obsah souboru. Její využitelnost je v komprimaci obrázků, videí nebo hudby. Přesto že se část původní informace nenávratně ztratí, je tento způsob komprimace velice výhodný. Ztráta některých informací je pak zcela vyvážena velmi výrazným zmenšením velikosti výsledných dat [2]. Příkladem může být nekomprimovaný obrázek, který zabírá na disku

mnohem více místa než stejný komprimovaný obrázek a přitom uživatel na obou obrázcích nevidí žádný rozdíl. Toto je způsobeno tím, že lidské oko má jen omezenou rozlišovací schopnost a na některých obrázcích v digitální formě jsou často „zbytečné informace“, které oko nepostřehne [1]. Na světlé ploše tedy těžko rozezná jeden černý bod, stejně tak, milion barevných odstínů v obraze. Barvy, které leží vedle sebe, oko průměruje [3].

1.2 Logická a fyzická komprimace

Všechny druhy komprimačních algoritmů mají za cíl, převádět data do kompaktnější formy, které mají stejnou informační hodnotu jako data původní. Toto nám zajišťuje logická nebo fyzická komprimace [1].

1.2.1 Logická komprimace

Logická komprimace využívá logické informační hodnoty komprimovaných dat. Používá nejčastěji substituce znaků jdoucích za sebou, jinou úspornější řadou [1]. Příkladem můžou být různé zkratky či zkratková slova (např.: Čedok = Česká dopravní kancelář). Metody komprimace na logické úrovni využívají sémantických vědomostí o složení a možné následnosti kódovaných informaci, díky tomu mohou dosahovat vyšších komprimačních poměrů, jsou však časově náročnější a použitelné pouze pro jistý specifický druh dat [4].

1.2.2 Fyzická komprimace

U fyzické komprimace nezáleží na sémantice dat. Pracuje bez zřetele na logiku dat. Vytváříme nové sekvence symbolů (bitů, bytů apod.), pokud možno kratší, podle určitého komprimačního algoritmu bez využití sémantických vědomostí o složení a možné následnosti informací. Fyzickou komprimaci používá většina komprimačních algoritmů [4].

1.3 Symetrická a asymetrická komprimace

1.3.1 Symetrická komprimace

Za symetrickou komprimační metodu se označuje ta, která spotřebuje stejné množství času na komprimaci i na dekomprimaci dat. Příkladem symetrické komprimační metody je Adaptivní Huffmanovo kódování [1].

1.3.2 Asymetrická komprimace

Asymetrickou komprimační metodou označujeme tu, která nespotřebuje stejné množství času pro komprimaci a dekomprimaci dat. U většiny komprimačních algoritmů je čas dekomprimace delší než čas komprimace, ale není to vždy pravidlem. Z toho vyplývá, že používají ke komprimaci a dekomprimaci jiného algoritmu. Příkladem asymetrické komprimační metody je LZ-77.

1.4 Rozdělení podle způsobu sestavování modelu.

Udávají nám, jakým způsobem algoritmus sestavuje pravděpodobnost k jednotlivým symbolům. Komprimace kodérem a dekomprimace dekodérem musí probíhat nad stejným modelem dat.

1.4.1 Statické komprimační metody

Statické komprimační metody používají při komprimaci statický model dat, který se nemění v závislosti na komprimovaných datech. V praxi jde zejména o četnosti výskytu jednotlivých zdrojových jednotek, které jsou na základě obecných pravidel vypočítány a používány pro komprimaci všech dat. Obě strany komprimace (kodér i dekodér) pracují se stejným modelem dat. Tento model se nepřenáší se zakódovanými daty a musí být znám před začátkem komprese. Příkladem statických komprimačních metod jsou: Morseova abeceda a Braillovo písmo

1.4.2 Adaptivní komprimační metody

Adaptivní komprimační metody vytvářejí model dat potřebný při komprimaci v závislosti na komprimovaných datech. Tento model není třeba ukládat společně se zakomprimovanými daty. Při dekomprimaci si dekodér vytváří stejný model dat jako kodér při komprimaci. Kodér při komprimaci vždy nejprve zakóduje další zdrojovou jednotku a teprve následně upraví svůj model podle této jednotky. Tento postup umožní dekodéru vytvářet si svůj identický model. Kdyby kodér nejprve upravil model a následně zakódoval vstupní jednotku, dekodér by přečetl zakódovanou jednotku podle modelu, který nezná. Většina používaných komprimačních metod je adaptivní. Příkladem adaptivních komprimačních metod jsou: Adaptivní aritmetické kódování, Metoda LZW a Metoda PPMC.

1.4.3 Semi-adaptivní komprimační metody

Semi-adaptivní komprimační metody vytvářejí model dat v závislosti na komprimovaných datech, tento model je však nutné přenést spolu se zakomprimovanou zprávou. Na straně dekodéru je nejprve nutné načíst uložený model dat. Teprve s načteným modelem je možné dekomprimovat data. Semi-adaptivní komprimační metody patří většinou mezi dvouprůchodové komprimační metody - potřebují ke komprimaci dvou průchodů komprimovaného textu. První průchod se většinou používá k výpočtu četností výskytu jednotlivých symbolů v komprimovaném textu. Při druhém průchodu se ke komprimaci využívá statistik vypočítaných v průchodu prvním. Příkladem semi-adaptivních komprimačních metod jsou: Shannon-Fanovo kódování, Statické Huffmanovo kódování a Aritmetické kódování.

2 ZÁKLADNÍ UKAZATELÉ UDÁVAJÍCÍ VÝKON KOMPRIMACE

2.1 Komprimační poměr

Je hodnota, která vyjadřuje efektivnost komprimace. Komprimační poměr se vypočítá jako podíl velikosti souboru zkomprimovaného (výstupního) a souboru původního (vstupního). Jestli je vypočítaná hodnota komprimačního poměru menši jak 1, došlo ke komprimaci souboru, čím více se komprimační poměr blíží k hodnotě 0, je komprimace účinnější. Jestli je hodnota komprimačního poměru větší jak 1, dochází k expanzi zkomprimovaného souboru, tedy k zvětšení zkomprimovaného souboru. Nejčastěji se hodnota komprimačního poměru udává v procentech. Výpočet se tedy vypočítá podle vztahu [5]:

$$komprimační\ poměr = \frac{\text{velikost výstupního souboru}}{\text{veliskost vstupního souboru}} \times 100\ [\%]$$

2.2 Faktor komprimace

Faktor komprimace je další hodnota, která vyjadřuje efektivnost komprimace. Jedná se o převrácenou hodnotu komprimačního poměru. Jestli jsou hodnoty faktoru komprimace větší než 1, jedná se nám o komprimaci dat a hodnoty menši jak jedna expanzi dat, tedy negativní komprimaci. Faktor komprimace se vypočítá podle vztahu [5]:

$$faktor\ komprimace = \frac{\text{velikost vstupního souboru}}{\text{veliskost výstupního souboru}}$$

2.3 Rychlost komprimace

Jedná se důležitý údaj zobrazující výkon komprimačních algoritmů. Rychlost komprimace nám udává, kolik času je zapotřebí ke zkomprimování vstupního souboru.

2.4 Rychlost dekomprimace

Rychlost dekomprimace nám udává, kolik času je zapotřebí k dekomprimaci zkomprimovaného souboru do původní podoby. Z hlediska údajů zobrazujících výkon komprimačních algoritmů se jedná o méně důležitý údaj. U většiny algoritmů je rychlost dekomprimace nižší než rychlost komprimace [1].

3 ROZDĚLENÍ KOMPRIMAČNÍCH ALGORITMŮ

Komprimační algoritmy se rozdělují do několika základních kategorií podle toho, jaké používají základní metody ke komprimaci dat.

3.1 Jednoduché komprimační metody

Využívají ke komprimaci dat posloupnost jednotlivých bitů či symbolů. Vyznačují se velkou rychlostí komprimace i dekomprimace, ale dosahují obecně špatného komprimačního poměru, který je značně závislý na typu vstupních dat. Patří mezi ně například metoda potlačení nul, více v [4] nebo metoda proudového kódování (RLE).

3.2 Statistické komprimační metody

Jsou založené na jednotlivých četnostech výskytu symbolů v komprimovaném proudu dat. Většinou jde o dvou průchodové komprimační metody, kdy prvním průchodem proudu je spočítána četnost jednotlivých symbolů v komprimovaném vstupním proudu dat a při druhém se teprve provádí samotná komprimace. Patří sem například Huffmanovo kódování, Shannon-Fanovo kódováni a Aritmetické kódováni.

3.3 Slovníkové komprimační metody

Slovníkové komprimační metody používají ke komprimaci slovník. Tento slovník se na počátku inicializuje, v průběhu komprimace i dekomprimace se upravuje přidáváním nových frází. Proto patří slovníkové metody mezi adaptivní komprimační metody. Komprimovaná data poté obsahují indexy frází ve vytvářeném slovníku. Patří sem například Metoda LZ-77, Metoda LZ-78 nebo Metoda LZW

3.4 Kontextové komprimační metody

Kontextové komprimační metody používají při komprimaci informaci o kontextu komprimovaného symbolu, to znamená, že tvar zkomprimované zprávy záleží na okolních symbolech původní zprávy. Patří sem například metody ACB, DCA a PMM.

3.5 Ostatní metody, které se používají v komprimačních algoritmech

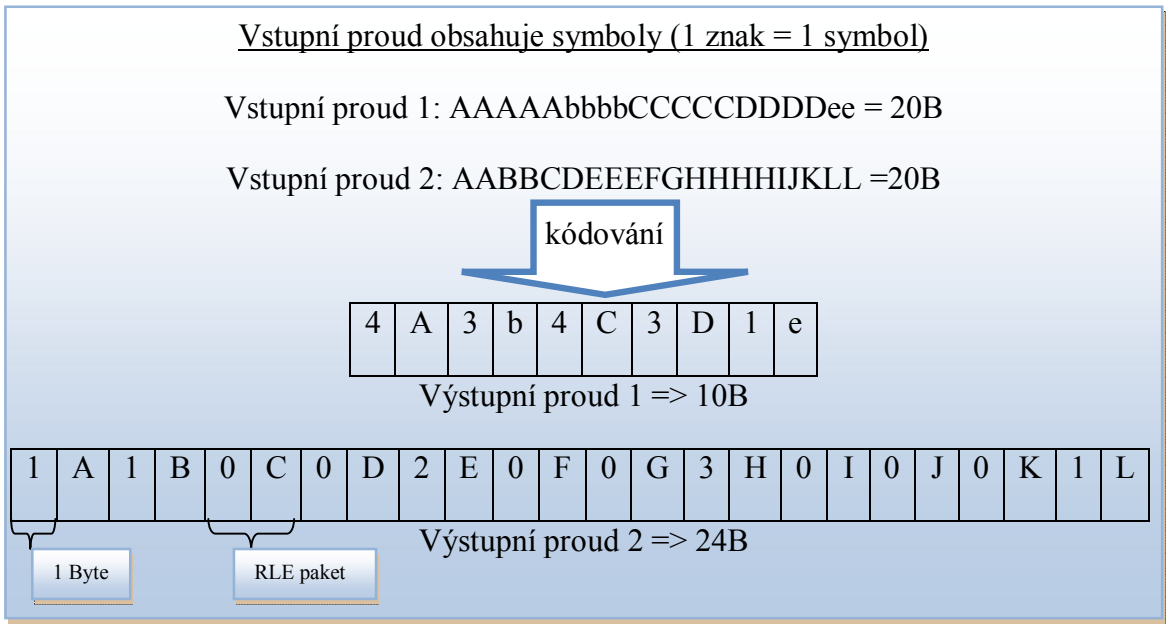
Při použiti těchto metod nedochází k samotné komprimaci dat, ale data jsou pouze

transformována, aby byl dosažen možná co nejlepší komprimační poměr. Patří sem například Burrows-Wheelerova transformace, nebo Move-to-front.

4 KOMPRIMAČNÍ ALGORITMY POUŽÍVANÉ KE KOMPRIMACI OBRÁZKŮ

4.1 RLE (Run-Lenght Encoding)

Run-Lenght Encoding v češtině také někdy nazývána metoda proudového kódování, je bezztrátová komprimace, která se vyznačuje velice jednoduchým algoritmem a velkou rychlostí komprimace i dekomprimace. Obecně se vyznačuje nízkým komprimačním poměrem, výjimku tvoří obrázky s většími jednobarevnými plochami a další. V praxi se využívá například ve formátu PCX [1], více na [6]. Algoritmus je založen na zhuštění po sobě jdoucích symbolů, tyto symboly se nazývají proud. Proud symbolů se komprimuje do tzv. RLE paketů. RLE paket je tvořen proudovým číslem, které udává počet opakujících se symbolů snížený o jedničku a proudovou hodnotou, která nám udává opakující se symbol.



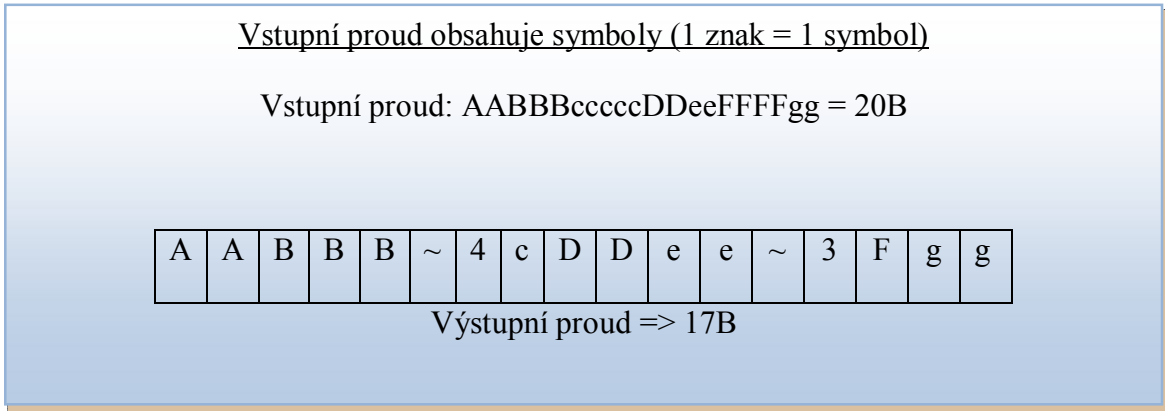
Obrázek 1: Příklad komprimace pomocí základní metody RLE

Tento způsob tvorby RLE paketu není příliš vhodný, protože pokud by v souboru bylo málo po sobě opakujících se symbolů, s největší pravděpodobností by docházelo k záporné komprimaci (viz. Obrázek 1). Proto existuje řada modifikací tvorby RLE paketů, které se snaží vyhnout záporné komprimaci při nízkém počtu opakujících se symbolů.

4.1.1 Tvorba RLE paketu pomocí identifikátoru na bytové úrovni

Tento typ RLE komprimace předchází záporné komprimaci, pomocí identifikátoru který uvozuje začátek RLE paketu. Při tomto druhu komprimace se zakódují pouze ty proudy,

které obsahují více než 3 symboly. Je-li proud dat menší nebo roven 3 symbolům, zapíše se paket přímo. Pokus je tedy větší než 3, zapíše se paket takto: první byte obsahuje identifikační číslo, druhý proudové číslo (počet opakujících se symbolů, mínus 1) a třetí proudovou hodnotu (opakující se symbol).



Obrázek 2: Příklad RLE kódování s využitím identifikátoru na bytové úrovni.

4.1.2 Tvorba RLE paketu pomocí identifikátoru na bitové úrovni

Základní myšlenka této modifikace metody RLE spočívá v tom, že nám nejvyšší bit v bytu velikosti proudu představuje tzv. identifikátor typu proudu. Pokud je nejvyšší bit nastaven na hodnotu 1, jedná se o klasický RLE paket. Nižších 7 bitů nám pak udává proudové číslo a následující bajt představuje proudovou hodnotu. Maximální velikost proudu tedy může byt 128 symbolů. Je-li identifikátor typu proudu nastaven na hodnotu 0, jedná se o tzv. přímý paket. V tomto případě nám proudová hodnota (opět uložena v nižších 7 bitech bajtu identifikátoru) udává přesný proud znaků, které se čtou v té podobě, jako jsou zapsána [3].

4.2 Huffmanovo kódování

Huffmanovo kódování je neznámějším zástupcem algoritmů, které pracují na základě různých četností znaků v kódovaných datech [1]. Algoritmus v roce 1952 poprvé představil David A. Huffman. Algoritmus vytváří Huffmanův kód, tj. kód s minimální délkou a také se jedná o takzvaný prefixový kód. Hlavní myšlenkou je přiřazení zástupného kódu k jednotlivým symbolům vstupní abecedy. Délka zástupného kódu závisí na jednotlivých četnostech výskytu symbolů. Symboly vstupní abecedy s větší četností výskytu se zapisují na výstup s využitím malého množství bitů (nejčetnější symbol může byt zapsán i v podobě jednoho bitu), naopak symboly, které se vyskytují ve vstupní abecedě méně častěji, se zapisují na výstup větším počtem bitů (není výjimkou, že datová

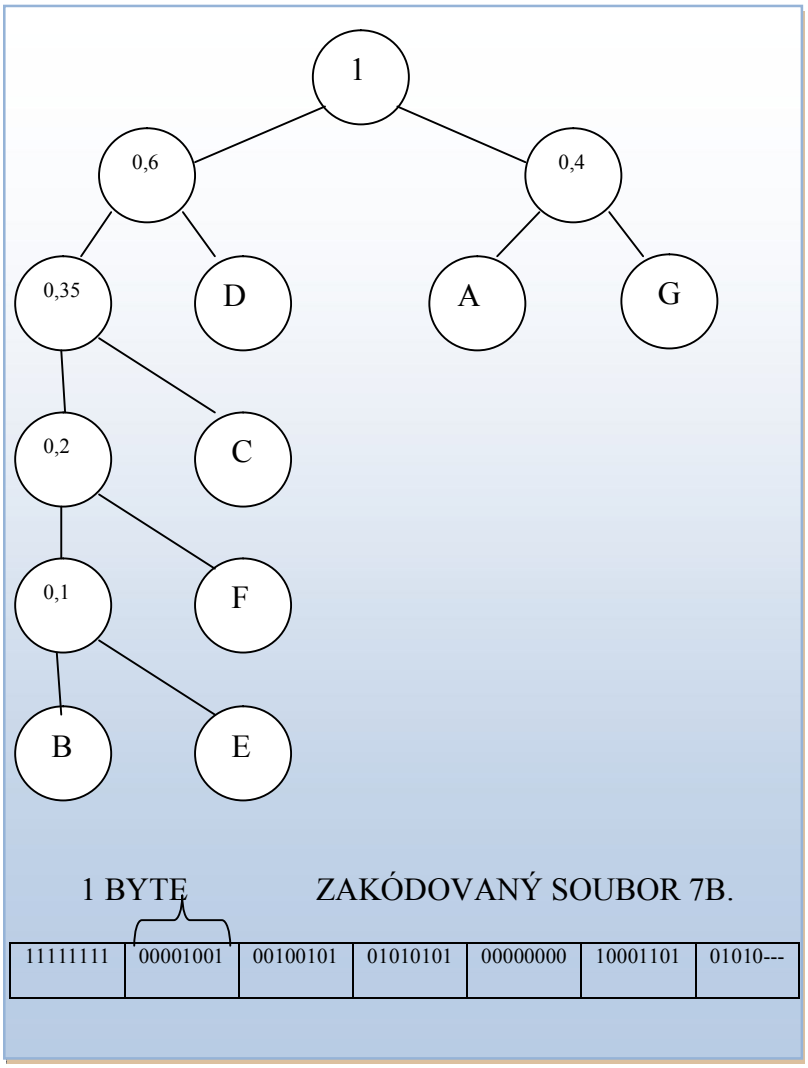
velikost výstupního kódu může přesáhnout i původní datovou velikost vstupního symbolu). K získání zástupného kódu se používá datová struktura binární strom, ve kterém jeho listy představují symboly vstupní abecedy, hrany jsou ohodnoceny symboly 0 (levé větve) a 1 (pravé větve) a všechny uzly stromu (včetně listů) jsou ohodnoceny pravděpodobností výskytu jednotlivých symbolů ve vstupní abecedě. Pravděpodobnost vnějšího uzlu je dána vždy součtem pravděpodobností všech vnitřních uzlů, z toho vyplývá, že kořen stromu má vždy pravděpodobnost rovnu jedné. Zástupný bitový kód vstupního symbolu se pak získá zřetězením hodnot hran, kterými projdeme průchodem stromu od kořene do daného listu [14].

4.2.1 Komprimace

Vlastní algoritmus pracuje tak, že nejdříve projde vstupní soubor a vytvoří statistiku četnosti jednotlivých symbolů obsažených ve vstupním souboru. Dále setřídíme symboly vzestupně podle pravděpodobnosti výskytu a poté vytváříme binární strom. Podle zásad Huffmanova algoritmu se binární strom vytváří od listů a od znaků s nejmenší pravděpodobností výskytu. Do datové struktury prioritní fronta budou uloženy všechny pravděpodobnosti výskytu jednotlivých symbolů, kde nejvyšší prioritu (bude odebrán jako první) má prvek s nejnižší pravděpodobností [14]. V dalším kroku odebereme vždy z fronty dva prvky (s nejnižší pravděpodobností) a spojíme je v jeden uzel, který bude mít ohodnocení dané součtem pravděpodobností svých synů (odebraných záznamů). Do fronty poté vložíme nový prvek (námi vytvořený uzel). Tento postup opakujeme až do té doby, než nám ve frontě zůstane pouze jediný záznam, který nám představuje kořen stromu s pravděpodobností rovné 1. Vstupní proud o velikosti 20 B obsahuje následující symboly (velikost symbolu je jeden znak): *AAAABCCCDDDDDEFF*GGGG. Vytvořený binární strom a zakódovaný vstupní proud.

Tabulka 1: Četnost jednotlivých znaků ve vstupním proudu

Znak	Četnost	Pravděpodobnost	Kód
D	5	0,25	01
A	4	0,2	10
G	4	0,2	11
C	3	0,15	001
F	2	0,1	0001
B	1	0,05	00001
E	1	0,05	00000
Suma	20	1	



Obrázek 3: Příklad Huffmanova binárního stromu

Po vytvoření binárního stromu máme definované kódy, které jsou přiřazené podle četnosti výskytu k jednotlivým symbolům. Kódování probíhá tak, že ke vstupnímu symbolu vyhledáme příslušný bitový kód a ten zapíšeme na výstup. Bitový kód odpovídá zřetězení hodnot hran, kterými projdeme při cestě z kořene do daného listu, který odpovídá pravě kódovanému symbolu. K dekódování vstupního proudu dat potřebujeme stejný binární strom, jako jsme měli v případě kódováni. Z toho vyplývá, že se jedná o metodu semi-adaptivní. Proto dekodéru musíme předat strukturu binárního stromu, nejčastěji ji zapisujeme na začátek výstupního souboru při kódováni. Po znovu sestavení binárního stromu, čteme vstupní proud dat bit po bitu a postupně od kořene procházíme stromem do té doby, než se dostaneme k listu, který nám představuje zakódovaný symbol. Symbol zapíšeme na výstup a strom začneme procházet opět od kořene. Toto opakujeme do té doby, dokud není konec vstupního proudu [7].

4.2.2 Adaptivní Huffmanovo kódování

Algoritmus je založen na původním Huffmanově kódování. Jedná se o nejstarší adaptivní algoritmus, byl nezávisle publikován Fallerem (1973), později Gallagerem (1978). V roce 1985 provedl další obměnu tohoto algoritmu Knuth. Tento algoritmus dostal zkrácený název FGK . Binární strom má sourozeneckou vlastnost, jestliže každý uzel (s výjimkou kořene) má sourozence a pokud lze uzly seřadit do monotónní posloupnosti (podle četnosti daného uzlu) tak, že má každý uzel v posloupnosti za souseda svého sourozence. Gallager potom dokázal, že binární prefixový kód je Huffmanovým kódem pravě tehdy, když má odpovídající kódovací strom sourozeneckou vlastnost. Algoritmus potom funguje na principu vkládání symbolů do stromu a v případě, že se podaří detekovat porušení sourozenecké vlastnosti, se provede přeuspořádání stromu tak, aby byla sourozenecká vlastnost zachována. Algoritmus má dvě varianty přístupu k abecedám, v prvním případě je strom na začátku nastaven tak, že obsahuje všechny znaky se zvolenou pravděpodobností. Druhy případ používá tzv. uzel zero. V tomto případě obsahuje počáteční strom pouze jediný uzel zero. Při načtení znaku, který dosud nebyl zakódován, se do výstupu vypíše kód uzlu zero a uzel se rozdělí na nový uzel zero a list s novým znakem. Další adaptivní verzí Huffmanova kódovani je Vitterův algoritmus [7]. Jedná se o podobný algoritmus jako FGK, ale používá trochu jiný způsob upravování stromu a v určitých případech tak dává lepší výsledky.

4.3 Shannon-Fanovo kódování

Tato metoda pochází z roku 1949. Publikovali ji nezávisle na sobě Claude-Elwood Shannon s Warrenem Weaverem a Robert Mario Fano. Jedná se o statistickou, semi-adaptivni metodu. Shannon-Fanovo kódování je velice podobné Huffmanovu kódování, Huffman při tvorbě algoritmu z tohoto kódování vycházel. Jediný podstatný rozdíl mezi oběma metodami je při tvorbě binárního stromu [1]. U Huffmanova kódování vytváříme binární strom od listů s nejmenší pravděpodobností, až ke kořenu, Shannon-Fanovo kódování má opačný přistup. Vytváří tedy binární strom od kořene až ke koncovým listům, tvorba binárního stromu je tedy jednodušší [14].

4.3.1 Komprimace a dekomprimace

Komprimace a dekomprimace pomoci algoritmu probíhá stejně jako u výše popsaného Huffmanova kódování (semi-adaptivního), proto ji zde nebudu podrobněji popisovat a

zaměřím se pouze na rozdílnou konstrukci binárního stromu. Ze začátku musíme opět získat pravděpodobnost jednotlivých symbolů vyskytujících se ve vstupním proudu dat. Vytváření binárního stromu, jak již bylo řečeno, probíhá od kořene k listům. V prvním kroku rozdělíme soubor vstupní abecedy na dvě skupiny se stejnou nebo co nejbližší pravděpodobností. První skupinu, která se nachází vlevo, označíme binární nulou a druhou jedničkou. Můžeme říci, že celá vytvořená skupina je nový list stromu. Toto dělení na skupiny se stejnou nebo nejvíce podobnou pravděpodobností opakujeme tak dlouho, dokud skupina nebude obsahovat pouze jediný symbol vstupní abecedy. V praxi se Shannon Fanovo kódování většinou nepoužívá, pro komprimaci se raději volí Huffmanovo kódování, které může dosahovat lepších výsledků, nikdy však nedosáhne horších výsledků než Shannon-Fanovo kódování [1].

4.4 Aritmetické kódování

Aritmetické kódování reprezentuje zprávu jako číslo z intervalu <0,1). Na začátku uvažujeme celý tento interval. Jak se zpráva prodlužuje, zpřesňuje se i výsledný interval a jeho horní a dolní mez se k sobě přibližují. Čím je kódovaný symbol pravděpodobnější, tím se interval zúží méně a k zápisu nám tedy stačí méně bitů. Nakonec stačí zapsat libovolné číslo z výsledného intervalu, které nám samo o sobě reprezentuje celou zprávu [8].

4.4.1 Komprimace

Algoritmus kódování můžeme jednoduše zapsat takto:

1. Inicializuj hodnoty:

Dolní interval =0, *Horní interval* =1

2. Opakuj tak dlouho, dokud není vstup prázdný:

(a) přečti vstupní symbol (s)

(b) *Rozsah= Horní interval - Dolní interval*

(c) *Horní interval = Dolní interval + Rozsah * Horní interval symbolu(s)*

(d) *Dolní interval = Dolní interval + Rozsah * Dolní interval symbolu(s)*

3. Výstupem je libovolná hodnota z intervalu (*Dolní interval, Horní interval*> a hodnota udávající délku vstupních dat, nebo speciální ukončovací symbol [9]. Pro lepší vysvětlení tohoto kódování uvedu příklad: Vstupní soubor obsahuje znaky: *abac*. Vstupní abeceda je

tedy: *a, b, c*. Pravděpodobnost znaků je a=0,5; b=0,25; c=0,25; nakonec symbolům přiřadíme intervaly a(0; 0,5>, b(0,5; 0,75>, c(0,75; 1> Průběh kódování můžeme vidět v tabulce 2.

Tabulka 2: Příklad kódování pomocí aritmetického kódování

Krok	Výpočet	Symbol	Výpočet intervalu
1	H=1,0 D=0,0 R=1,0	H=0,5 a D=0,0	H= 0 + 1 * 0,5 = 0,5 D= 0 + 1 * 0,0 = 0,0
2	H=0,5 D=0,0 R=0,5	H=0,75 b D=0,5	H= 0 + 0,5 * 0,75 = 0,375 D= 0 + 0,5 * 0,50 = 0,25
3	H=0,375 D=0,25 R=0,125	H=0,5 a D=0,0	H= 0,25 + 0,125 * 0,5 = 0,3125 D= 0,25 + 0,125 * 0,0 = 0,25
4	H=0,3125 D=0,25 R=0,0625	H=1,0 c D=0,75	H= 0,25 + 0,0625 * 1,0 = 0,3125 D= 0,25 + 0,0625 * 0,75 = 0,296875
5	H=0,3125 D=0,296875	Libovolné číslo z výsledného intervalu <0,296875; 0,3125) nám reprezentuje danou zprávu. Vybereme, pokud možno, nejkratší reprezentaci čísla v našem případě to může byt hodnota 0,3.	

4.4.2 Dekomprimace

Dekomprimace zakódované zprávy probíhá obdobným způsobem. Příklad kódováni je uveden na statickém modelu, proto je třeba do zkomprimovaného souboru zapsat informace, aby dekodér dokázal znovu sestavit intervaly jednotlivých symbolů. Můžeme například uložit pravděpodobnost výskytu jednotlivých symbolů a před výpočtem intervalu seřadit symboly abecedně, to platí i v případě kódování, aby nám nedošlo např. k záměně intervalu symbolů se stejnou pravděpodobností. Dále uvádím popis dekódovacího algoritmu:

1. Načti zakódovanou hodnotu (*hodnota*) a zrekonstruuj intervaly symbolů.

2. Inicializuj hodnoty: *Dolní interval* =0.0 *Horní interval* =1.0

3. Opakuj tak dlouho, dokud se počet dekódovaných symbolů nerovná počtu zakódovaných symbolů, či dokud nenarazíš na konstantu „eof“:

(a) urči, do jakého intervalu patří (*hodnota)*

(b) zapiš na výstup symbol (*s*), kterému náleží daný interval

(c) *hodnota = [hodnota - Dolní interval symbolu(s)] / Pravděpodobnost symbolu(s)*

Dekódování hodnoty 0,3 z minulého přikladu ukazuje tabulka 3.

Tabulka 3: Příklad dekódování pomocí aritmetického kódování

Krok	Hodnota	Aktuální interval	Výstup	Výpočet
1	0,3	a <0; 0,5)	a	hodnota = (0,3 – 0) / 0,5 = 0,6
2	0,6	b <0,5; 0,75)	b	hodnota = (0,6– 0,5) / 0,25 = 0,4
3	0,4	a <0; 0,5)	a	hodnota = (0,4 – 0) / 0,5 = 0,8
4	0,8	c <0,75; 1)	c	----------
Výstup abac				

4.5 LZ-77

Tuto komprimační metodu uvedli poprvé v roce 1977 Abraham Lempel a Jacob Ziv. Jedná se o metodu slovníkovou, která položila základ mnoha jiným slovníkovým metodám, které jsou více či méně od této metody odvozeny. Slovníkové metody mají několik výhod oproti metodám statistickým, které jsme popsali výše. Statistické metody nahrazují symbol kratším vyjádřením hlavně podle pravděpodobnosti výskytu, které reprezentuje model. Algoritmus LZ-77 kóduje vstup jako posloupnost trojic. Základní myšlenkou je, že pokud kódujeme vstup od nějakého místa, pak můžeme využít předcházející vstup. Pokud najdeme nějakou shodnou sekvenci, je dán pouze odkaz na tuto sekvenci. Metoda je založena na principu posuvného okna, které je rozděleno na dvě časti. První část mnohonásobně větší, nám reprezentuje slovník a při dalším popisu této metody budeme tuto část nazývat jako prohlížecí okno. Do prohlížecího okna jsou postupně ukládány již zpracované symboly. Druhá kratší část posuvného okna, nazývejme ji aktuální okno,

obsahuje přednačtené, ještě nezpracované symboly. Velikost prohlížecího okna se nejčastěji pohybuje od 2^{12} do 2^{16}bitů. Metoda je vhodná pro jednorázové zakódování a četná dekódováni. Používá se v algoritmu zvaném Deflate, což je kombinace metody LZ-77 a Huffmanova kódování. Nejdříve je vstupní proud zakódovaný metodou LZ-77 a poté se pro zakódování pozic znaků a délek použije Huffmanovo kódování. Metodu LZ-77 používá formát PNG.

4.6 LZSS

Tento algoritmus v roce 1982 publikoval James Storer a Thomasem Szymanski. Vychází z algoritmu LZ-77, základem je opět posuvné okno, které je opět rozděleno na prohlížecí a aktuální část. Hlavní rozdíl oproti LZ-77 je v podobě výstupu, který je reprezentován buď symbolem, nebo dvojicí <*i,j*>, kde *i* nám udává pozici prvního symbolu ve slovníku a *j* nám představuje počet shodných symbolů. Oproti LZ-77 se neshodný symbol nezapisuje. Pokud se nám nevyplatí kódování symbolů, zapisuje se na výstup přímo. Aby bylo možné poté data zpět dekomprimovat, musíme poznat, zda se jedná o symbol, či odkaz do slovníku, proto musíme uvádět před kódovaným blokem vždy identifikační bit [5].

4.7 LZ-78

Tento algoritmus publikovali v roce 1978 opět Abraham Lempel a Jacob Ziv. Na rozdíl od předchozí metody zde nepoužíváme žádné posuvné okno, ale vytváříme si dynamický slovník z načtených (zakódovaných) symbolů [5]. Velikost slovníku je omezena pouze velikostí paměti. Výstupem kodéru je dvojice <*i, S*>, kde *i* nám představuje ukazatel na pozici ve slovníku a *S* symbol, který se již ve slovníku nenachází. Komprimaci začínáme s prázdným slovníkem, ze vstupu načteme symbol. Jelikož slovník je prázdný symbol, uložíme na pozici 1 další nenalezený symbol, uložíme na pozici 2 atd. Symboly samozřejmě zakódované zapisujeme do výstupního proudu. Nyní načteme symbol b, prohledáváme slovník, pokud není nalezen, uložíme symbol do slovníku na pozici 3 a na výstup zapíšeme <0, b>. Jiná situace nastane, pokud symbol b je ve slovníku nalezen, například na pozici 2. Poté načteme další symbol ze vstupního proudu, kterým bude například p, opět prohledáme slovník a hledáme, zda neobsahuje záznam s hodnotou bp. Hodnotu bp nenalezneme, a tak na novou pozici – v našem případě konkrétně tři, uložíme danou hodnotu bp a na výstup zapíšeme <2,p>. Příklad vytváření slovníku je

uveden v tabulce 4. Budeme kódovat vstupní proud obsahující následující symboly: emamamaso.

Tabulka 4: Příklad vytváření slovníku u metody LZ-78

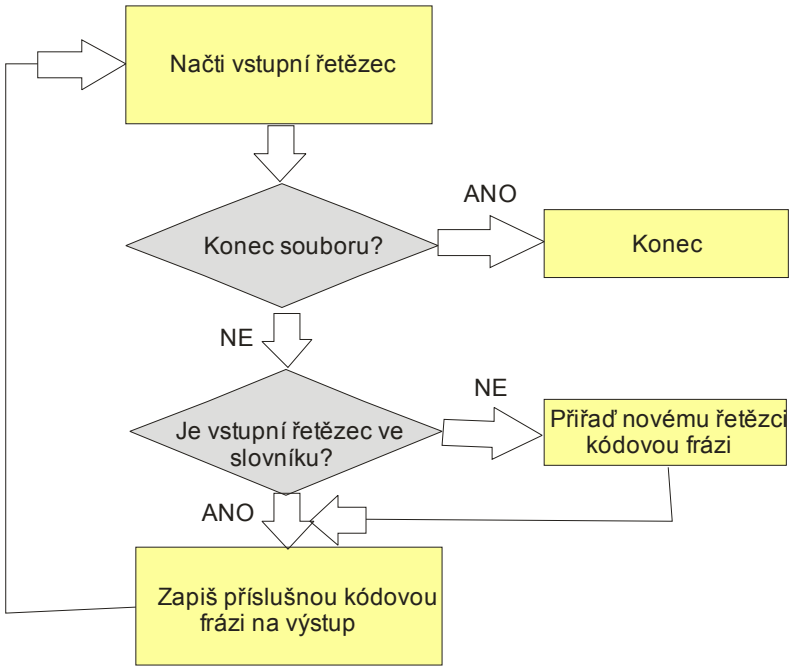
Slovník		Výstup
Pozice	Hodnota	
1	e	<0,“e“>
2	m	<0,“m“>
3	a	<0,“a“>
4	ma	<2,“a“>
5	mas	<4,“s“>
6	o	<0,“o“>

Dekomprimace probíhá analogicky, znovu vytváříme, respektive rekonstruujeme slovník, načteme dvojici dat, podíváme se na pozici ve slovníku, na výstup zapíšeme její hodnotu plus druhou položku z načtené dvojice (tedy nenalezený symbol při komprimaci). Nakonec do slovníku vložíme novou položku, jejíž hodnota bude reprezentována námi zapsanými daty na výstup (tedy hodnotou, na kterou ukazuje první z dvojice plus symbol S).

4.8 LZW

Metodu LZW (Lempel-Ziv-Welch) publikoval v roce 1984 Terry Welch, jedná se o vylepšení metody LZ-78. Na výstup zapisujeme pouze číselné označení daného listu, což je hlavni rozdíl oproti LZ-78 [5]. Před začátkem komprimace musíme inicializovat slovník, který bude obsahovat celou vstupní abecedu. Tímto je zajištěno, že každý symbol bude ve slovníku nalezen a odpadá nám tím zapisování nenalezeného symbolu na výstup, jak tomu je v případě algoritmu LZ-78. Postup komprimace je dále obdobný jako v předchozím případě, jen s malými obměnami. Jak již bylo řečeno, slovník musí být zpočátku inicializovaný, pokud velikost vstupního symbolu je 8 bitů a budeme uvažovat opět pro reprezentaci slovníku datovou strukturu k-cestný strom, kořen stromu bude mít na počátku komprimace 256 synů. Při samotné komprimaci načteme symbol ze vstupního proudu dat a

prohledáme slovník, zda se v něm nachází načtený symbol – pokud ano, načteme další symbol a prohledáváme následující symboly (v rámci struktury k-cestny strom, jsou to synové nalezeného symbolu), pokud symbol není nalezen, zapíše na výstup číselnou hodnotu posledního nalezeného symbolu. Poté prohledáváme strom znovu od začátku, od nenalezeného symbolu. Průběh komprimace můžeme vidět na obrázku 4. Pro dekomprimaci stačí pouze zkomprimovaný řetězec, slovník se vytváří opět postupně během dekomprimace. Na vstupu se může objevit i odkaz na frázi, která ještě ve slovníku není. Tuto frázi nám představuje poslední dekódovaná fráze a její první symbol. Takto novou frázi přidáme do slovníku a zapíšeme do výstupního proudu. Metodu LZW využívají ke komprimaci grafické formáty GIF a TIFF, komprimační program WinZip, nebo dokumenty PS a PDF. Podobně jako RLE funguje dobře u obrázků s jednobarevnými plochami. Lépe zvládá barevné přechody, ale těch nesmí být mnoho [5].



Obrázek 4: Princip LZW komprese

4.9 JPEG (JFIF)

JPEG je komprimační metoda určená ke komprimaci obrázků, vyvinula ji skupina expertů, která se nazývala Joint Photographics Experts Group (JPEG), protože byla složena z členů komise ISO a členů skupiny Photographics Experts Group (PEG). Skupina se sešla v roce 1982 a do roku 1987 bylo navrženo dvanáct různých metod komprimace rastrových obrázků. Z těchto metod bylo v roce 1989 vyčleněno základní kompresní schéma JPEGu,

které se skládá z barvové transformace, diskrétní kosinové transformace a Huffmannova kódování. Finální revize kompresního schématu byla provedena v roce 1990 a následně, konkrétně v roce 1992 došlo k definitivnímu schválení schématu. Od tohoto data se začínají objevovat nové a nové aplikace i elektronické výrobky, které přímo či nepřímo JPEG používají [13].

JPEG je ztrátová komprimační metoda, která využívá faktu, že selektivním zanedbáním určité informace obsažené v obrázku je možné dosáhnout mnohem lepšího komprimačního poměru než při bezeztrátové komprimaci. Mohou se však zanedbat pouze ty informace, které při prohlížení nezpůsobí viditelnou ztrátu kvality obrázku. Ve finále může být obrázek uložený do JPEGu velmi odlišný od originálu, zatímco člověk v mnoha případech žádnou ztrátu informace nepozná. U JPEGu je možné dosáhnout komprimačního poměru 1:20 až 1:30 při zanedbatelné ztrátě informace. Je vhodné zdůraznit, že JPEG není a ani nikdy nebyl určen pro ukládání obrázků obsahujících malé množství barev, kontrastní barevné přechody, ostré hrany, písmo apod. V těchto případech bude obrázek rozmazaný anebo se značným způsobem zhorší komprimační poměr [13].

4.9.1 Komprimace

Kompresní algoritmus JPEG se skládá z následujících částí:

1. **Transformace barev** - Snímky (pokud používají jiný barevný model, např. RGB) se převedou do barevného modelu YUV (nebo YC_BC_R), kde je samostatně uchovávána jasová (Y) a dvě barevné složky (U a V). Tento mód je pro další zpracování mnohem vhodnější než např. často používaný RGB. Důvodem je vlastnost lidského oka lépe rozpoznávat změny v intenzitě (jasu) a nikoliv v malé změny v barvě.

   RGB → YC_BC_R (1 byt na složku - tři bajty na pixel, zatím žádná ztráta dat)

$$\begin{pmatrix} Y \\ C_B \\ C_R \end{pmatrix} = \begin{pmatrix} 0{,}2990 & 0{,}5870 & 0{,}1140 \\ -0{,}1687 & -0{,}3313 & 0{,}5000 \\ 0{,}5000 & -0{,}4187 & 0{,}0813 \end{pmatrix} \times \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

   nebo

$$C_B = 0{,}5643 \times (B - Y)$$

$$C_R = 0{,}7133 \times (R - Y)$$

2. **Redukce barev** - týká se pouze barevných složek C_B, C_R. Jasová složka Y zůstává v původním stavu. Existují dvě varianty. První je průměrování sousedních dvojic pixelů (původně 3+3=6 bajtů se redukuje na 4 bajty - 2 bajty pro nezměněné složky

Y, 2 bajty pro společné složky C_B, C_R obou pixelů). Druhá je průměrování čtyř sousedních pixelů do čtverce (původních 4×3 bajtů se redukuje na 6 bajtů - 4 na Y, 2 na C_B, C_R)dochází tak k nenávratné ztrátě dat (která je ovšem většinou vizuálně málo patrná)[15]

3. **Diskrétní kosinová transformace** - Obraz se rozdělí do čtvercových oblastí 8x8 bodů a na každou takovou oblast se aplikuje diskrétní kosinová transformace. Tato transformace v podstatě převede amplitudovou informaci na informaci frekvenční, u které lze snadno rozlišit, které části informace jsou pro danou oblast obrazu dominantní (nižší frekvence) a které části jsou pouze jemné detaily (vyšší frekvence). Získáme tak opět matici 8x8 bodů. Na rozdíl od původních matic, kde byla obrazová informace rozložena rovnoměrně (pro každý pixel jsme znali jeho barevnou hodnotu), po aplikování transformace se informace s největším vlivem dostaly do levého horního rohu a se stoupající vzdáleností od něj důležitost informace klesá [14]. Tím se dominantní a nejdůležitější složky obrazu oddělí od drobných detailů a obraz se tím připraví k další redukci, tj. vypuštění těchto detailů. Vzorec pro diskrétní kosinovu transformaci:

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)cos\frac{(2x+1)u\pi}{16}cos\frac{(2y+1)v\pi}{16}\right]$$

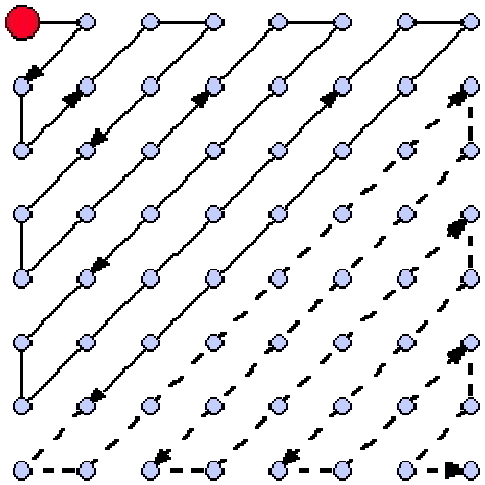
$$C(u) = C(v) =$$

$$= \frac{1}{\sqrt{2}} pro\ u = v = 0$$

$$= 1\ jinde$$

4. **Kvantifikace** - Až do této fáze se v úpravách obrazu jednalo o vratné operace a nedocházelo (kromě podvzorkování barev) ke ztrátě informací. Nyní přichází onen fundamentální krok: provedením kvantifikace dochází k nezvratným změnám. Každá z 64 hodnot jednotlivých bloků se po diskrétní kosinové transformaci vydělí kvantifikačními koeficienty (Q=1-100), které jsou součástí předem experimentálně připravených kvantifikačních matic a zaokrouhlí se na celá čísla. Podle volby kvantifikační matice můžeme ovlivnit redukci dat, ke které dojde a tak i výslednou velikost a kvalitu obrázku [16].
5. **Kódování** - Po provedení kvantifikace dostaneme řídkou matici, která kromě několika koeficientů v levém horním rohu obsahuje jen samé nuly. Tuto matici převedeme do posloupností čísel, přičemž postupujeme po diagonálách od levého

horního rohu k pravému dolnímu rohu. Takto získaná posloupnost se zakóduje Huffmanovým nebo aritmetickým kódováním. Již tedy nedochází k žádné ztrátě informací.



Obrázek 5: Kódování do výstupního souboru

6. **Konečná fáze** - Přidání hlavičky souboru ke komprimovaným datům. Aby dekompresor mohl správně rekonstruovat grafická data je nutno připojit také kvantifikační tabulku a tabulku Huffmanových kódů, či poměry pravděpodobností v případě, že bylo použito aritmetické kódování.

4.9.2 Dekomprimace

Postup dekomprimace probíhá přesně obráceně. Zpětným dekódováním se získají příslušné matice 8×8 hodnot pro jednotlivé složky Y,C_B,C_R a provede se zpětné násobení s členy z kvantizační tabulky (dekvantování). Dále se provede zpětná diskrétní kosinová transformace: $f(x,y) = \frac{1}{4}\left[\sum_{x=0}^{7}\sum_{y=0}^{7} C(u)C(v)F(u,v)cos\frac{(2x+1)u\pi}{16}cos\frac{(2y+1)v\pi}{16}\right]$ a nakonec zpětná transformace barev $YC_BC_R \rightarrow$ RGB. Jelikož se jedná o ztrátovou komprimaci, přesně nezrekonstruujeme data z původní podoby[16].

4.9.3 Bezztrátový JPEG

Joint Photographic Experts Group se zabývala i vývojem standardu, jež se nazýval JPEG-LS, který je bezztrátový. Pracuje velmi dobře se složitými barevnými předlohami, nikdy však nemůže dosahovat stejných kompresních poměrů jako ztrátový JPEG [16]. Vzhledem k velice malému využívání tohoto schématu, Joint Photographic Exprerts Group zastavila před časem veškeré vývojová práce na projektu JPEG-LS.

4.9.4 JPEG 2000

V roce 1995 byla vytvořena nová specifikace JPEG, která dostala název JPEG 2000 a to podle roku, kdy bylo naplánováno její uvedení. Na jeho vývoji pracuje kromě Joint Picture Expert Group a ISO JPEG Commitee i Digital Imaging Group. V roce 2000 byla zveřejněna první díl specifikace dílu jpeg2000, která obsahovala 90% algoritmu tohoto formátu. JPEG 2000 používá místo komprimace DCT (Diskrétní kosinová transformace), která redukuje zobrazovací body do čtvercových bloků o hraně 8 pixelů, komprimaci waveletovou neboli vlnkovou. JPEG 2000 pracuje s obrázkem jako s celkem a převádí je na popisy pomocí vlnkových funkcí. Převod je víceprůchodový, počet průchodů určuje kompresní poměr a kvalitu dekomprimovaného obrázku (méně průchodů=vyšší kompresní poměr=nižší kvalita obrazu). Každému průchodu odpovídá zvláštní datový blok komprimovaného souboru. Vlnková komprese se objevila v polovině 80. let. Původně jako čistě matematická metoda. JPEG 2000 má lepší, rychlejší a kvalitnější kompresi, zvýšení komprimačního poměru je zhruba 20 až 30 procent oproti JPEG při ztrátové kompresi. Poměr bezztrátové komprese je ½ [16]. U JPEG 2000 máme možnost zpracovávat obrázky větší než 64000 krát 64000 pixelů, těmito hodnotami byl omezen klasický JPEG. Dále může kromě obrazové informace obsahovat i metadata, udávající např. název, datum a čas pořízení, autora, popis, vlastníka autorských práv atd [12].

4.10 Fraktální komprese

Fraktální kódování je matematický postup používaný ke kódování bitmapových obrazů reálného světa jako množiny matematických dat, která vyjadřují fraktální vlastnosti obrazu. Fraktální kódování vychází ze skutečnosti, že všechny přirozené a většina umělých obrazů obsahují nadbytečné informace ve formě podobných, opakujících se vzorů, tzv. fraktálů. Fraktální komprese je ztrátová a asymetrická. Fraktální komprese se hodí zejména na kompresi přírodních obrazů. Velmi špatných výsledků je dosahováno při práci s obrazy, které obsahují text a kresby. Tyto části obrazu jsou po kompresi deformovány, někdy dokonce úplně zmizí, to je typické hlavně pro přímé čáry, které směřují diagonálně k hranám řadových bloků. Zajímavou vlastností je také tzv. fraktální interpolace to znamená, že dekomprimovaný obraz má znaky fraktálu. Je možné ho zvětšovat prakticky donekonečna a stále se objevují nové detaily, které nejsou obsažené v původním komprimovaném obraze.

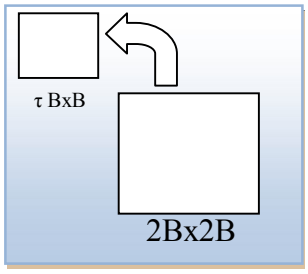
4.10.1 Komprimace

1. Segmentace - Obrázek se rovnoměrně rozdělí na segmenty o rozměrech BxB do dvourozměrného pole. Obvykle jsou rozměry obrázku 2^l x2^l , např. pokud jsou rozměry obrázku 512 x 512 nebo 256x256 , potom range bloky volíme 4x4, 8x8, 16x16, 32x32. Tyto bloky jsou zpravidla zpracovávány postupně řádek po řádku.

Tabulka 5: Rozdělení obrázku na range bloky

R_{11}	R_{12}	….	R_{1n}
R_{21}	R_{22}	….	R_{2n}
….	….	….	….
R_{m1}	R_{m2}	….	R_{mn}

2. Doménové bloky - Pro každý range blok hledáme doménový blok a vhodnou transformaci tak, aby oba bloky byly po transformaci co nejvíce podobné. Vhodné doménové bloky hledáme v „zásobníku“ doménových bloků. Zásobník doménových bloků se skládá z bloků o rozměrech 2Bx2B z původního obrázku. Získáme ho posouváním 2Bx2B okénka po původním obrázku s krokem δ. Pokud má obrázek rozměry M x M, pak existuje právě $(\frac{M-2B}{\delta}+1)\times(\frac{M-2B}{\delta}+1)$ takových bloků. Například pokud je rozměr původního obrázku 256x256, range bloky jsou 4x4 a velikost kroku δ=4, potom existuje 63x63 doménových bloků. Při kompresi porovnáváme každý range blok se všemi doménovými bloky a hledáme nejlepší dvojici.
3. Afinní transformace - Afinní transformace je mapování doménového bloku na range blok viz obrázek 6. Je to složené zobrazení $\tau_j = T_j x S_j$ kde S_j je operátor průměru a T_j je lineární mapa. Díky této minimalizaci poté nalezneme nejlepší pár.



Obrázek 6: Mapování doménového bloku na range blok

4.10.2 Dekomprimace

Komprimační proces je velice časově náročný, proto se většina výzkumníků snaží o jeho zrychlení. Naopak dekomprese je proces jednoduchý a velice rychlý. Stačí pouze iterovat získané transformace z libovolného počátečního obrázku. Pro dobrou kvalitu výsledného obrázku většinou postačí 8 iterací.

5 FORMÁTY OBRÁZKŮ

Formáty obrázků se dělí na obrázky rastrové a obrázky vektorové. U rastrových obrázků je základem rastr pixelů s informacemi o jejich barvě. Při změně velikosti obrázků dochází k problémům (viditelné kostičky), mají dobré možnosti práce s barvou. U vektorových formátů jsou základem čáry, které lze matematicky popsat, mají určenou velikost, tloušťku, barvu a další atributy. U těchto formátů není problém s jejich zvětšováním, ale mají horší možnosti popisu barev. Mezi nejznámější vektorové formáty patří DXF (Autocad) nebo CDR (Corel).

5.1 BMP

Microsoft Windows Bitmap používá bitovou mapu, to znamená, že každému pixelu je přiřazena barva pomocí RGB složek. Bez ohledu na obsah obrázku je velikost souboru vždy stejná (výška × šířka × počet bitů na pixel + hlavička; v hlavičce je uvedena velikost obrázku, způsob komprese a další údaje). Formát většinou nepoužívá žádnou komprimaci a je tedy vhodný pro snímky, které mají být v maximální dosažené kvalitě, případně může využít metodu RLE, ovšem jen pro omezenou 8-bitovou barevnou paletu.

5.2 GIF

Graphics Interchange Format používá barevnou paletu o velikosti maximálně 256 barev. Formát používá komprimační metodu LZW. Umožňuje uložit více obrázků (i s různými paletami) do jednoho souboru. Takto lze vytvářet jednoduché animace (podporuje i cyklickou animaci, různou délku prodlev mezi snímky). S počtem snímků a barevných palet ovšem narůstá velikost souboru. U GIFů lze také nastavit pro jednu barvu z palety průhlednost a prokládání řádků (interlaced), to znamená, že jednotlivé řádky obrazu nejsou ukládány za sebou, toto uložení je vhodné pro obrázky přenášené po internetu, kdy postupné vykreslování obrázku vzbuzuje dojem jeho zaostřování (v neprokládaném režimu je obrázek vykreslován po řádcích odshora dolů) [10].

5.3 PNG

Portable Network Graphics byl vyvinut jako zdokonalení a náhrada formátu GIF, který byl patentově chráněný, dnes jsou patenty prošlé. Je vhodný především na přenos kvalitních obrázků po internetu. Používá komprimační metodu LZ-77 nebo LZW tak jako GIF. Podporuje barevnou hloubku až 32 bitů (3x8 bitů pro paletu RGB a 8 bitů pro alfa-kanál

průhlednosti). Podporuje dvojrozměrný prokládaný režim v 7 krocích, po řádcích i sloupcích. Na rozdíl od GIFů nemůže ukládat více obrázků do souboru, nelze ho tedy užít na vytváření animací. To je možné až u inovovaného APNG. Většina zařízení, které jsou schopny zobrazit PNG obrázek, jsou schopny pracovat s APNG s tím, že zobrazí pouze první obrázek animace [10].

5.4 JFIF (JPEG)

JPEG File Interchange Format používá ztrátovou JPEG komprimaci. Je to formát vhodný pro fotografie a obrázky s mnoha barevnými přechody a nevhodná pro obrázky s výraznými hranami jako jsou texty atd. Nejrozšířenější příponou tohoto formátu je .jpg, .jpeg, .jfif, .jpe. Formát JFIF je nejrozšířenějším formátem pro ukládání obrázků vůbec [10].

5.5 TIFF

Tag Image File Format je určen pro ukládání statických obrazů všeho druhu. Umožňuje možnost ukládání více bitmap do jednoho souboru, a proto se často používá například pro ukládání přijatých faxů. Podporuje až 24 bitovou hloubku. Byly u něj používány různé metody komprimace. U verze 3.0 to bylo Huffmanovo kódování, podporovala pouze barvy ve stupních šedi. U verze 4.0 bylo použito RLE kódování a podpora RBG palety barev. Verze 5.0 užívala LZW kódování a u 6.0 to byla metoda JPEG, zde již byla i podpora CMYK palety barev [10].

5.6 PCX

PC Paintbrush File Format umožňuje ukládat obrázky s barevnou hloubkou 1-24 bitů. Používá se buď bez komprese nebo s metodou RLE. Původně byl formát PCX navržen k ukládání obrázků v aplikaci PC Paintbrush, postupem času i přes malou otevřenost dokumentace se rozšířila jeho podpora i na jiné aplikace. Dnes již se od tohoto formátu upouští. Licenci na používání tohoto formátu drží firma Microsoft [6].

5.7 SVG

Scalable Vector Graphics je značkovací jazyk a formát souboru, který popisuje dvojrozměrnou vektorovou grafiku pomocí XML. Formát SVG by se měl v budoucnu stát základním otevřeným formátem pro vektorovou grafiku na Internetu. SVG definuje tři

základní typy grafických objektů: vektorové tvary, rastrové obrazy a textové objekty. Tyto objekty mohou být různě seskupeny, formátovány pomocí atributů nebo stylů CSS a polohovány pomocí obecných prostorových transformací [11].

II. PRAKTICKÁ ČÁST

6 POROVNÁNÍ KOMPRIMAČNÍCH METOD Z HLEDISKA KVALITY VÝSLEDNÉHO OBRAZU

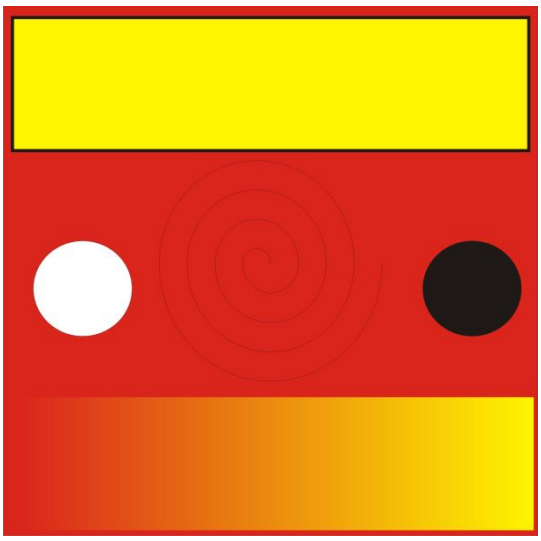
V této části se budu zabývat porovnáváním výsledků komprimace pomocí různých metod. Testovány budou různé typy obrázků např. fotografie vytvořená fotoaparátem, jednoduchý obrázek vytvořený v programu Corel Draw a rozměrové malý obrázek obsahující mnoho detailů a text. Ukládány budou do různých typů souborů s různými kompresemi.

6.1 Jednoduchý obrázek vytvořený programem Corel Draw

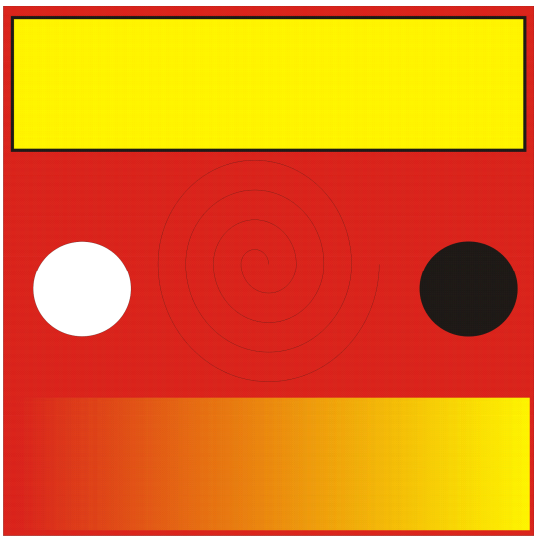
Obrázek má velikost 2000x2000 pixelů, rozlišení 300DPI a skládá se ze sytě oranžového pozadí, v barevném modelu CMYK (0,100,100,0). Dále se na obrázku nacházejí dva kvádry, jeden se silným obrysem a jednotnou výplní, druhý pak bez obrysu a s přechodovou výplní. Vprostřed obrázku je tenká spirála a dva kruhy, jeden s výplní bílou, druhý s černou. Obrázek byl s vektorového formátu převeden na rastrový ve velikosti 2000x2000 pixelů s rozlišením 300 DPI, v režimu barev RGB (24 bitů), základní velikost obrázku je 11,4 MB (12 000 054 bytů) a režimu barev paletovém (8 bitů) kde základní velikost obrázku je 3,81 MB (4 000 998 bytů).

6.1.1 BMP bez komprimace

Po exportu do formátu BMP bez jakékoliv komprimace zůstala kvalita obrazu (24 bitového i 8 bitového) stoprocentní bez jakýchkoliv změn oproti převedenému základu. Budeme tedy považovat tyto obrázky za základní s nejvyšší možnou kvalitou.



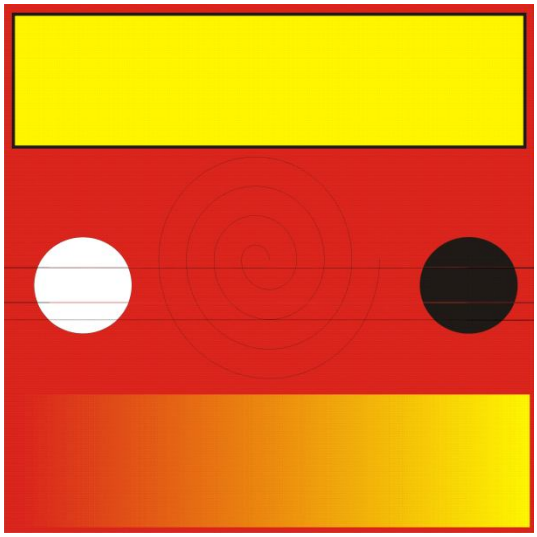
Obrázek 7: BMP bez komprimace (24 bitů)



Obrázek 8: BMP bez komprimace (8 bitů)

6.1.2 BMP s komprimační metodou RLE

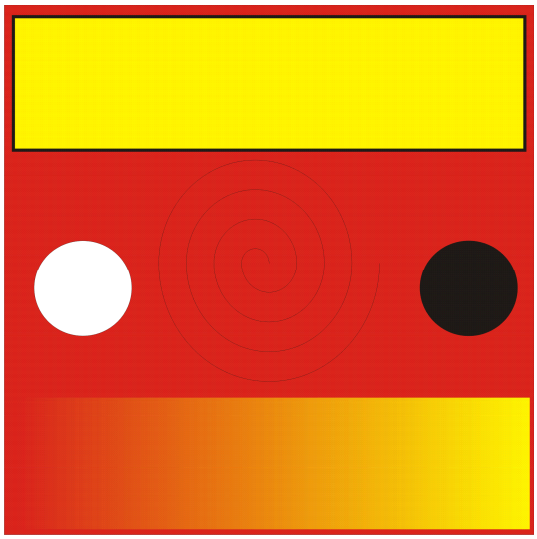
Jelikož RLE komprimaci lze využít pouze u bitmap s 4 nebo 8 bitovou barevnou hloubkou, použijeme zde obrázek s 8 bity na pixel. U jednobitových a TrueColor bitmap jsou obrázky vždy uloženy v nekomprimované podobě. U tohoto druhu komprimace se na obrázku objevilo několik pruhů, které obrázek znehodnotily. V tomto případě je výhodnější obrázek uložit v nekomprimované formě jelikož komprimační poměr metodou RLE je velice malý. Výsledný soubor má velikost 3,32 MB (3 482 014 bytů).



Obrázek 9: BMP komprimované metodou RLE

6.1.3 GIF

GIF jak již bylo zmíněno, používá metodu komprimace LZW s barevnou paletou max. 256 barev. Takže využijeme znovu základní obrázek o 8 bitové barevné hloubce. Pro tento typ obrázku a ostatní čárovou grafiku je volba formátu GIF velice vhodná, jelikož užívá bezztrátové komprimace, tak zůstávají okraje ostré a celková kvalita obrazu je dobrá při výborném komprimačním poměru. Výsledný obrázek se tedy viditelně neliší od původního, velikost komprimovaného obrazu je 257 kB (263 652 bytů).



Obrázek 10: Obrázek ve formátu GIF

6.1.4 PNG

U typu souboru PNG můžeme použít 24 bitový obrázek, oproti GIFu tento formát podporuje až 32 bitovou hloubku. Komprimační metodu využijeme LZ-77. Výsledek komprimace tohoto obrázku je výborný, tak jako u GIFu nejsou viditelné žádné odlišnosti od originálu. Je vidět, že tento formát je také velice vhodný pro typy souborů s čárovou grafikou. Komprimační poměr je ještě lepší než u GIFu.

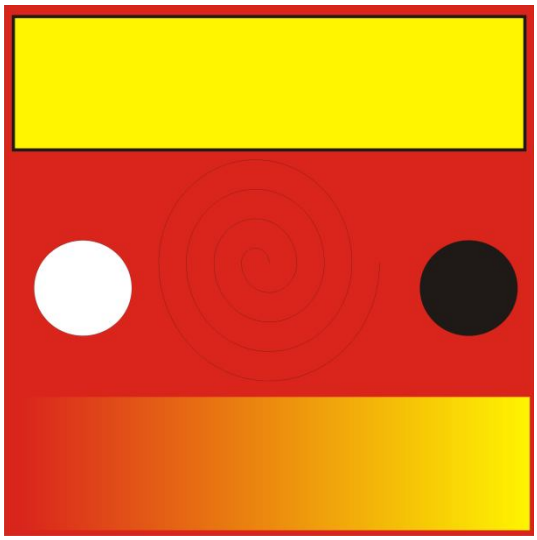
6.1.5 PCX

Formát PCX užívá ke komprimaci metodu RLE, na rozdíl od BMP, umí použít i 24bitovou paletu barev. Výsledný soubor má na rozdíl od BMP komprimovaného metodou RLE vysokou kvalitu, nejsou na něm viditelné změny oproti původnímu obrázku. Jeho velikost je díky podpoře 24 bitové palety 668kB (684 644) a to je znatelně nižší než u formátu BMP komprimovaného RLE.

6.1.6 TIFF

Typ souborů TIFF má více možností komprimace, my zde zkusíme využít typy bez komprimace, s komprimací LZW. Ostatní druhy komprimací jako jsou Huffmanovo kódování (bylo ve verzi TIFF 3.0, pouze stupně šedi) nebo komprimační metody pro faxové dokumenty jsou pro nás nepoužitelné. Bez komprimace je obrázek totožný jako obrázek nekomprimovaný BMP. Je to vlastně základní neboli nejkvalitnější obrázek. Komprimace metodou LZW je velice kvalitní, nejsou zde žádné viditelné problémy. Oproti

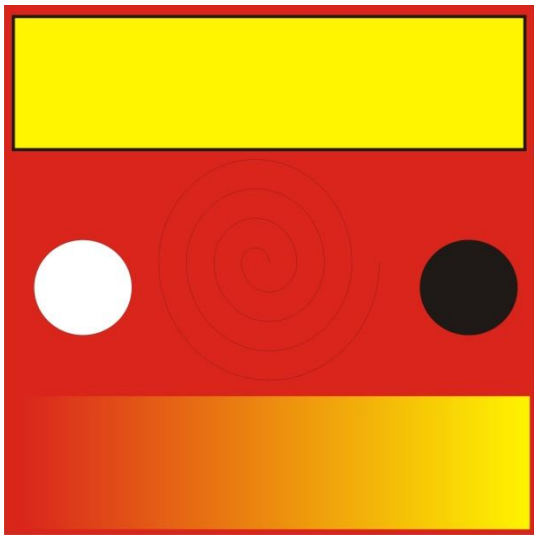
formátu GIF, který využívá stejnou metodu, je obrázek větší, je to dáno bitovou hloubkou, jelikož TIFF využívá až 24 bitů.



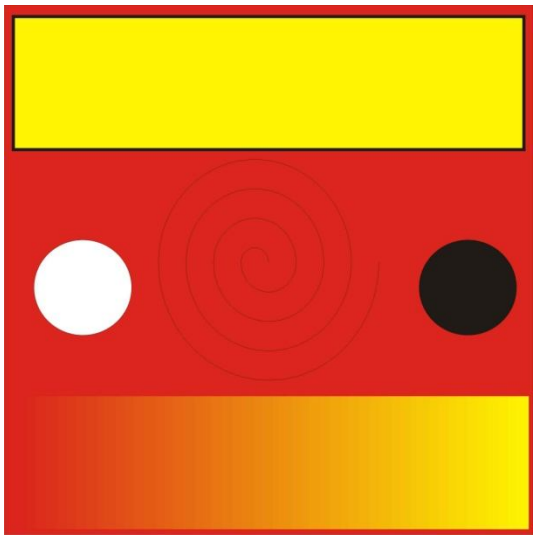
Obrázek 15: TIFF metodou LZW

6.1.7 JPEG

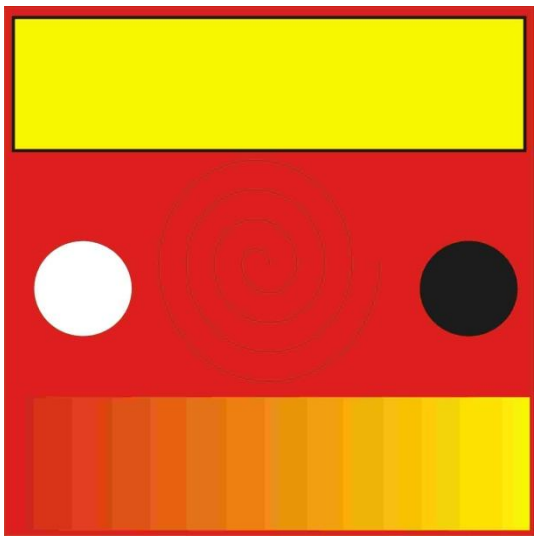
JPEG také podporuje až 32 bitovou barevnou hloubku, takže využijeme obrázek 24 bitový. Obrázek zkusíme komprimovat s různým stupněm komprimace. Začneme stupněm komprimace 0 a zkusíme, jak obrázek bude vypadat při maximu, tj. při stupni komprimace 100. Při stupni nula je obrázek velice kvalitní jen s minimálními, okem neviditelnými změnami oproti originálu. Jenže prakticky nikdy by se stupeň komprimace 0 neměl nastavovat, protože v případě požadavku na vyšší kvalitu bývá výhodnější použít bezeztrátové komprimační metody např. PNG. U stupně komprimace 30, i když začínají být viditelné změny v obraze hlavně u části s přechodovou barevnou výplní, dělají se pruhy místo plynulého přechodu barev, je obraz stále kvalitní. Nastavíme-li stupeň komprimace na 60, pruhy se hodně zvýrazní a změny jsou viditelné i na dalších částech obrázku. Linka, kterou je narýsována spirála začíná být rozostřena a hrany ostatních částí taktéž. Stupeň komprimace 90 je pro obrázek již úplně nepoužitelný. Změny v obrázku jsou již obrovské, vznikají zde čáry a ostatní artefakty, které sem nepatří a v původním obrázku nebyli. U stupně komprimace 100 jsme dostali už jiný obrázek, než byl počáteční, zmizela nám spirála a z barevného přechodu jsou obdélníky.



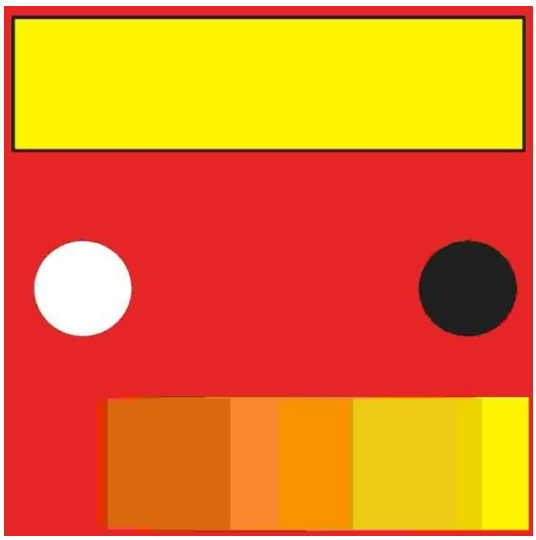
Obrázek 11: JPEG stupněm komprimace 0



Obrázek 12: JPEG stupněm komprimace 60



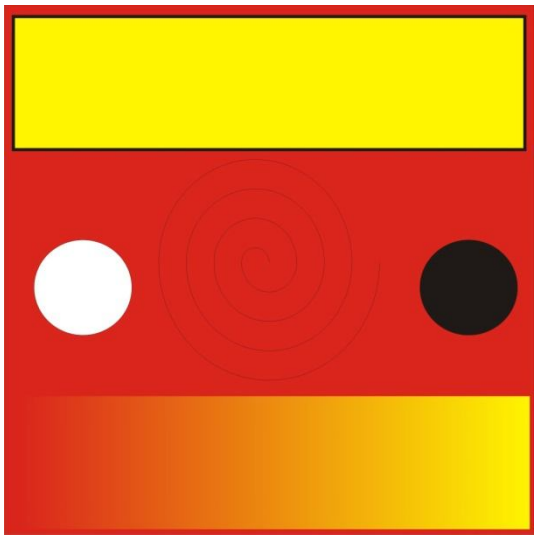
Obrázek 13: JPEG stupněm komprimace 90



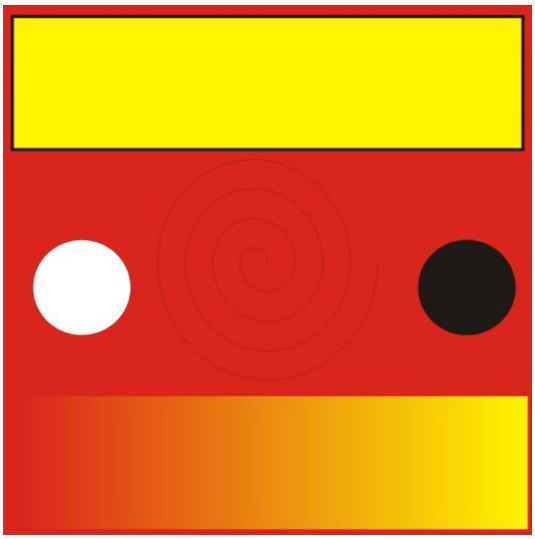
Obrázek 14: JPEG stupněm komprimace 100

6.1.8 JPEG 2000

JPEG 2000 na rozdíl od JPEGu klasického používá poměrně novou metodu vlnkové transformace. U stupňů komprimace 0-60 nebyli téměř viditelné žádné změny oproti původnímu obrázku. Až u stupně komprimace 90 začíná být vidět lehké rozmazání objektů. Ale na rozdíl od JPEGu to nejsou klasické kostičky. Poté když je ještě více přidáván stupeň komprese přes 95, kvalita obrázku rapidně klesá. U stupně 100 téměř není poznat, jak vypadal originál.



Obrázek 16: JPEG 2000, stupeň komp. 0



Obrázek 17: JPEG 2000 stupeň komp. 90

Obrázek 18: JPEG 2000 stupeň komprimace 100

Máme-li jednoduchý obrázek, vytvořený pomocí čárové grafiky je velice výhodně zvolit jednu z bezztrátových komprimačních metod. Výborné výsledky mají všechny typy souborů, které tyto komprimační metody využívají až na soubor typu BMP komprimovaný RLE metodou. U ztrátových komprimačních metod jsou viditelné velké deformace obrazu, při stejném komprimačním poměru jako má například soubor uložený ve formátu PNG.

6.2 Obrázek vyfocený fotoaparátem s velkým rozlišením

Pro další část testování kvality obrázků byl vybrán obrázek vyfocený fotoaparátem Panasonic DMC-LS80. Fotografie byla pořízena v rozlišení 8,1 megapixelu (3264x2448 pixelu). Uložena byla automaticky fotoaparátem do souboru JPEG, který měl velikost 3,76 MB. To znamená, že fotografie je již komprimovaná, komprimace je minimální a proto

byla převedena do formátu BMP s původním rozlišením a barevnou hloubkou 24bitů. V tomto formátu má fotografie velikost 22,8 MB (23 970 870 bytů) a my tento obrázek budeme považovat za základní. Pro komprimace s podporou nižších barevných hloubek ji převedeme na obrázek využívající 8 bitovou paletu barev, ten má velikost 7,62 MB (7 991 270 bytů).

Obrázek 19: Základní obrázek vyfocený fotoaparátem

6.2.1 BMP s komprimací RLE

Nyní zkusíme fotografii komprimovat pomocí metody RLE. Uložíme ji do formátu BMP, musíme použít obrázek, který využívá 8 bitovou paletu barev kvůli již zmíněnému nepodporování truecolor palety barev u komprimace RLE v kombinaci s formátem obrázku BMP. Obrázek se oproti originálu vůbec neliší, komprimační poměr je velice malý což se u tohoto typu obrázku dalo očekávat. Velikost výsledného komprimovaného obrázku je 6,53MB (6 850 348 bytů).

6.2.2 GIF

Pořízenou fotografii zkomprimujeme metodou LZW a uložíme do formátu GIF. Tak jako v předchozím případě užijeme 8 bitový obrázek, jelikož formát GIF nepodporuje větší barevnou paletu. Obrázek komprimovaný touto metodou opět nezaznamenal žádných změn oproti originálu, komprimační poměr je v tomto případě o něco lepší než u předchozí komprimační metody. Velikost souboru je v tomto případě 1,93MB (2 034 352 bytů).

6.2.3 PNG

Jelikož PNG podporuje i 24 bitovou barevnou hloubku, použijeme obrázek základní s 24 bitovou barevnou hloubkou. PNG využívá taktéž bezztrátovou metodu a to LZ-77. Komprimace obrázku touto metodou byla ze všech nejdelší. Kvalita obrázku je opět výborná, opět se obrázek vůbec nezměnil. Komprimační poměr je docela špatný, výsledný obrázek má velikost 11,9MB (12 558 876 bytů), je tedy horší než u předchozího formátu GIF.

6.2.4 PCX

Dalším formátem, do kterého byl obrázek uložen je PCX. Tento formát využívá metodu komprimace RLE s použitím 24 bitové barevné hloubky. Ani u tohoto formátu se na obrázku vůbec nic nestalo, zůstal zachován ve své původní kvalitě, komprimační poměr je však velice špatný, velikost výsledného souboru je 19,2 (20 146 151 bytů). To se od původního nekomprimovaného obrázku příliš neliší.

6.2.5 TIFF

U TIFF byla opět využita komprimace LZW, jelikož ostatní, které lze u tohoto typu souboru použít se nám opět nehodí. V nekomprimované formě souboru se opět nic nemění, jak velikost, tak kvalita obrazu. V komprimované formě je obrázek nezměněn oproti původnímu a jeho komprimační poměr je opět špatný, velikost výsledného souboru je 11,4MB (12 023 011 bytů). Je sice lepší než u typů souboru PCX nebo PNG, ale stále je to nedostatečné.

Obrázek 20: Uložení do souboru TIFF s LZW komprimací

6.2.6 JPEG

Jak již bylo několikrát zmiňováno formát JPEG je ztrátový, a proto se od něj dají očekávat horší výsledky v kvalitě obrazu, ale výrazně lepší komprimační poměry. Obrázek budeme zkoušet komprimovat stupni komprimace od 0-100.

U stupňů komprimace 0-30 není viditelné žádné velké změny v cílovém obrázku. Při vyšším stupni, mezi 40-50 dochází k mírné změně barev oblohy a domu v levé části obrázku. Od stupně komprimace 60 u obrázku začíná docházet ke změnám velkým, barevné přechody přestávají být plynulé. Stupeň 60 a 70 by mohl být při zmenšení použitelný. Stupně komprimace vyšší již jsou zcela nepoužitelné. Dochází u nich k obrovskému znehodnocení obrázku. U stupně 0 komprimační poměr docela špatný, výsledný soubor má velikost 4,43MB (4 646 490 bytů), zatímco stupeň 30, na kterém nejsou viditelné změny má komprimační poměr znatelné lepší. Velikost tohoto výsledného souboru je 783kB (802 103 bytů).

Obrázek 21: Fotografie ve formátu JPEG se stupněm komprimace 0

Obrázek 22: Fotografie ve formátu JPEG se stupněm komprimace 50

Obrázek 23: Fotografie ve formátu JPEG se stupněm komprimace 90

Obrázek 24: Fotografie ve formátu JPEG se stupněm komprimace 100

6.2.7 JPEG 2000

Poslední typ souboru, který použijeme, je JPEG 2000. Komprimace touto metodou je oproti metodě JPEG výhodná v tom, že i při stupni komprimace 50 nejsou viditelné žádné změny v obraze při zachování výborné velikosti souboru. Stupeň komprimace 90 je ještě stále velice dobře použitelný, oproti formátu JPEG při stejném stupni komprimace je obrázek bez příliš viditelných barevných změn. Až při stupni 95 se obrázek viditelně začíná rozmazávat a při stupni 100 již z obrázku nevidíte, jak vypadal původní, v našem případě dokonce ztratil i barevnost. Z toho vyplývá, že použití komprimace JPEG 2000 je velice výhodná. Obrázky ztrácejí kvalitu až při vysokých stupních komprimace oproti klasickému JPEGu. Dokonce výsledné soubory jsou při stejném stupni komprimace menší.

Obrázek 25: Obrázek ve formátu JPEG 2000 se stupněm komprimace 0

Obrázek 26: Obrázek ve formátu JPEG 2000 se stupněm komprimace 90

Obrázek 26: Obrázek ve formátu JPEG 2000 se stupněm komprimace 99

Obrázek 27: Porovnání stupňů komprimace u JPEG a JPEG 2000

U typu obrázků jako jsou fotografie krajin a podobných složitých obrázků je velice výhodné použít některou ze ztrátových komprimačních metod. Jelikož lidské oko neumí rozeznat některé chyby v obraze, které právě touto komprimací vznikají, je nevhodné používat bezztrátové komprimační metody. U těchto metod je výsledek čili komprimovaný soubor stejně kvalitní jako nekomprimovaný, ale na úkor velikosti souboru. Lepší výsledky

má komprimace JPEG 2000, která používá waveletovou transformaci. Problém je v tom že se tento formát příliš nerozšířil a některé programy ho nepodporují, přitom klasický JPEG je velice oblíbený a kompatibilní s obrovským množstvím programů. Když se u této metody nastaví stupeň komprimace optimálně, výsledek je také postačující.

6.3 Obrázek s malými rozměry a textem

Třetí obrázek, na kterém otestujeme kvalitu komprimace různými metodami, je obrázek s malými rozměry, velkým množstvím detailů a přidaným textem. Obrázek je opět vyfotografován fotoaparátem Panasonic DMC-LS80, dále je do něj přidán text a poté zmenšen na velikost 400x300 pixelů. Provedeme-li uložení do formátu BMP bez jakékoliv komprimace s 24 bitovou hloubkou velikost souboru je 351 kB (360 054 bytů) a s 8 bitovou hloubkou 118kB (120 998 bytů). U 8-bitové barevné hloubky jdou poruchy na obraze zaviněné nedostatečnou paletou barev.

Obrázek 28: Základní obrázek v BMP s 24 a 8 bitovou hloubkou

6.3.1 Bezztrátové metody

U všech bezztrátových komprimačních metod nedošlo ke změnám ve výsledném obrazu, pouze kromě uložení do souboru typu BMP s komprimací RLE, u něj se vytvořil v horní části obrázku černý pruh. U GIFu s komprimací LZW, PNG s komprimací LZ77 a TIFFu s komprimací LZW došlo ke komprimaci souboru, ale u formátů, které používají metodu RLE došlo k expanzi souboru. Jsou to formáty BMP a PCX. U formátu PCX je expanze souboru vysoká, výsledný soubor má velikost 427 kB (438 170 bajtů), oproti 351 kB (360 054 bytů) souboru počátečního nekomprimovaného. U formátu BMP to byly pouze 2 kB rozdílu mezi počátečním a zkomprimovaným souborem. K nejlepšímu komprimačnímu poměru došlo u formátu GIF, i tak je komprimační poměr velice špatný.

Obrázek 29: Uložení do formátu BMP s komprimační metodou RLE

6.3.2 JPEG

Do stupně komprimace 30 je obrázek ještě docela kvalitní, nebude-li se dále zvětšovat. Při zvyšování stupně komprimace již vzniká rozostření hran, začínají se dělat typické kostičky, kolem písma se začínají objevovat drobné tečky, okraje okvětních lístků se začínají rozmazávat, a čím více stoupá stupeň komprese, obrázek je rozmazanější a kolem písma se tečky zvětšují. Obrázek je použitelný do stupně komprimace kolem 60, poté již jsou nepřesnosti oproti základnímu obrázku veliké a stupeň komprimace 90-100 je již úplně nepoužitelný. U posledně jmenovaného již nepoznáme, co je na něm vyobrazeno.

Obrázek 30: Komprimace JPEG se stupněm komprimace 0

Obrázek 31: Komprimace JPEG se stupněm komprimace 60

Obrázek 32: Komprimace JPEG se stupněm komprimace 80

Obrázek 33: Komprimace JPEG se stupněm komprimace 100

6.3.3 JPEG 2000

U tohoto typu komprimace je stejně jako u klasického JPEGu dobrá kvalita do stupně komprimace 30, i když i u něj je viditelné slabé rozmazání, přidáváme-li stupeň komprimace, obrázek se rozmazává více. Kolem stupně 60 jsou již některé části obrázku rozmazané hodně, obrázek je ale stále docela dobře použitelný a stupeň komprimace 90 je pro obrázek kritický, rozmazání už je veliké a u 100 je to vlastně šmouha.

Obrázek 34: JPEG 2000 se stupněm komprimace 0

Obrázek 35: JPEG 2000 se stupněm komprimace 60

Obrázek 36: JPEG 2000 se stupněm komprimace 80

U tohoto obrázku jsme se poprvé setkali s expanzí výsledného souboru, bylo to u metody komprimace RLE. U bezztrátových komprimací došlo jen u formátu BMP s metodou RLE ke změně výsledného obrázku. U ztrátových komprimačních metod, jsou to JPEG a JPEG 2000, jsou výsledky téměř stejné, obrázek má dobrou kvalitu asi do stupně komprimace 30. A velikosti se výrazně liší pouze u stupně komprimace 0, kde je klasický JPEG asi 1a ½ krát větší.

7 POROVNÁNÍ KOMPRIMAČNÍCH METOD Z HLEDISKA VELIKOSTI VÝSLEDNÉHO SOUBORU

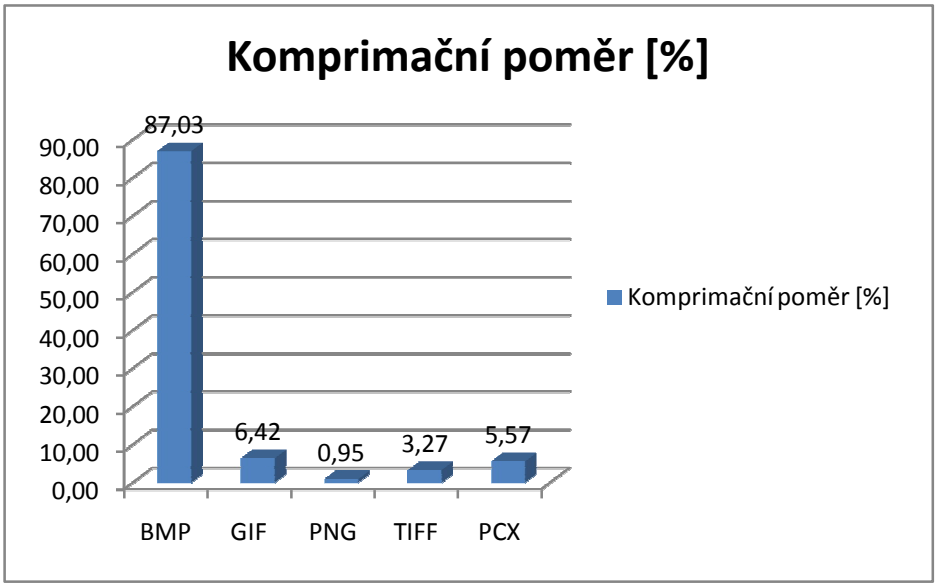
V této části porovnáme výsledné již zkomprimované obrázky podle velikosti. Porovnávat budeme každý ze třech typů souborů a komprimační metody zvlášť.

7.1 Jednoduchý obrázek vytvořený programem Corel Draw

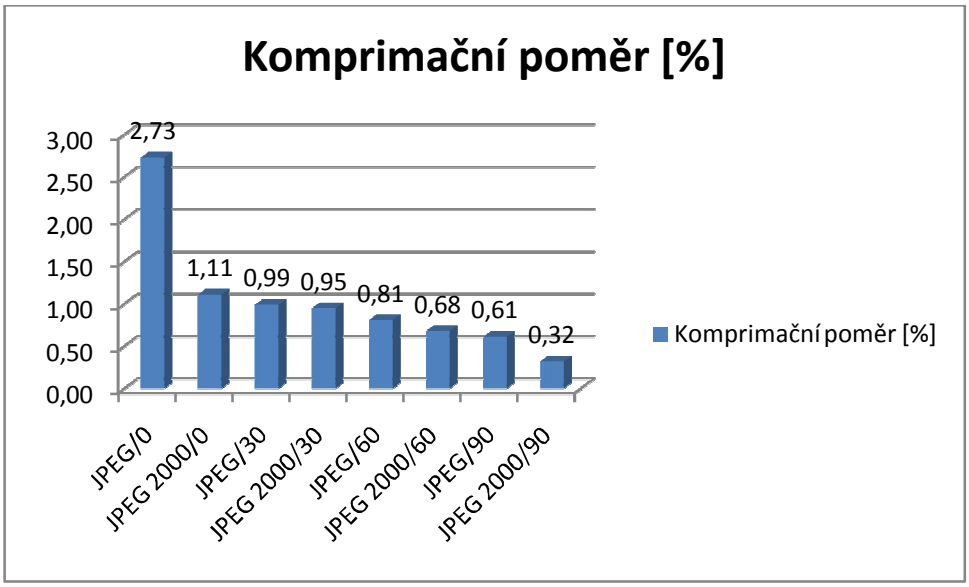
Tabulka 6: Porovnání velikostí výsledných souborů u obrázku vytvořeného v Corel Draw

Formát souboru	Metoda komprimace/stupeň komprimace	Barevná hloubka obrázku	Velikost základního obrázku (kB)	Velikost zkomprimovaného obrázku (kB)	Komprimační poměr (%)
BMP	RLE	8 bitů	4001	3482	87,03
GIF	LZW	8 bitů	4001	257	6,42
PNG	LZ77	24 bitů	12000	114	0,95
TIFF	LZW	24 bitů	12000	392	3,27
PCX	RLE	24 bitů	12000	668	5,57
JPEG	JPEG/0	24 bitů	12000	327	2,73
JPEG	JPEG/30	24 bitů	12000	119	0,99
JPEG	JPEG/60	24 bitů	12000	97,5	0,81
JPEG	JPEG/90	24 bitů	12000	73,6	0,61
JPEG 2000	JPEG 2000/0	24 bitů	12000	133	1,11
JPEG 2000	JPEG 2000/30	24 bitů	12000	114	0,95
JPEG 2000	JPEG 2000/60	24 bitů	12000	81,1	0,68
JPEG 2000	JPEG 2000/90	24 bitů	12000	38,7	0,32

Z bezztrátových komprimačních metod má nejlepší komprimační poměr formát souboru PNG s metodou komprimace LZ77. Oproti základnímu obrázku jeho velikost značně klesla a přitom mu tato metoda neubrala na kvalitě, ostatní metody již byli úspěšné méně, i když velikosti výsledných souborů jsou stále na skvělé úrovni. Ze ztrátových komprimačních metod dopadla lépe metoda JPEG 2000, má lepší komprimační poměr než klasický JPEG. Pro tento typ obrázku je ale lepší použít již zmiňované metody bezztrátové, protože již u JPEG se stupněm komprimace 0 vzniká určitá ztráta v obraze, i když je často pouhým okem neviditelná, při obrovském zvětšení by vidět být mohla. Navíc poměr velikostí souborů mezi ztrátovými a bezztrátovými metodami je téměř stejný, nebo spíše lepší pro ty bezztrátové.



Obrázek 37: Graf komprimačních poměrů používajících bezztrátové komprimační metody u obrázku vytvořeného programem Corel Draw



Obrázek 38: Graf komprimačních poměrů JPEG a JPEG 2000, při různých stupních komprimace u obrázku vytvořeného v programu Corel Draw

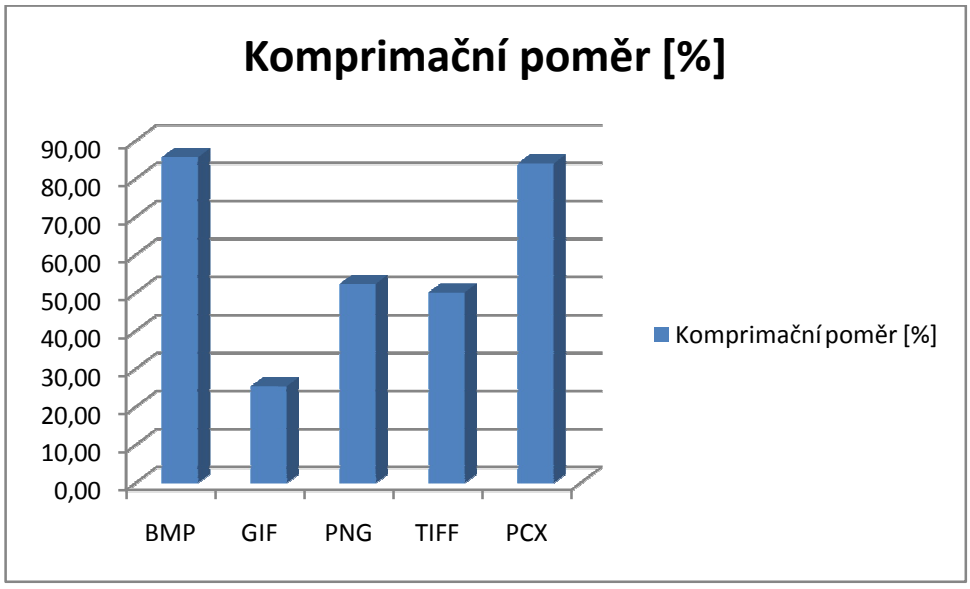
7.2 Obrázek vyfocený fotoaparátem s velkým rozlišením

Tabulka 7: Porovnání velikostí výsledných souborů u obrázku vytvořeného fotoaparátem s vysokým rozlišením

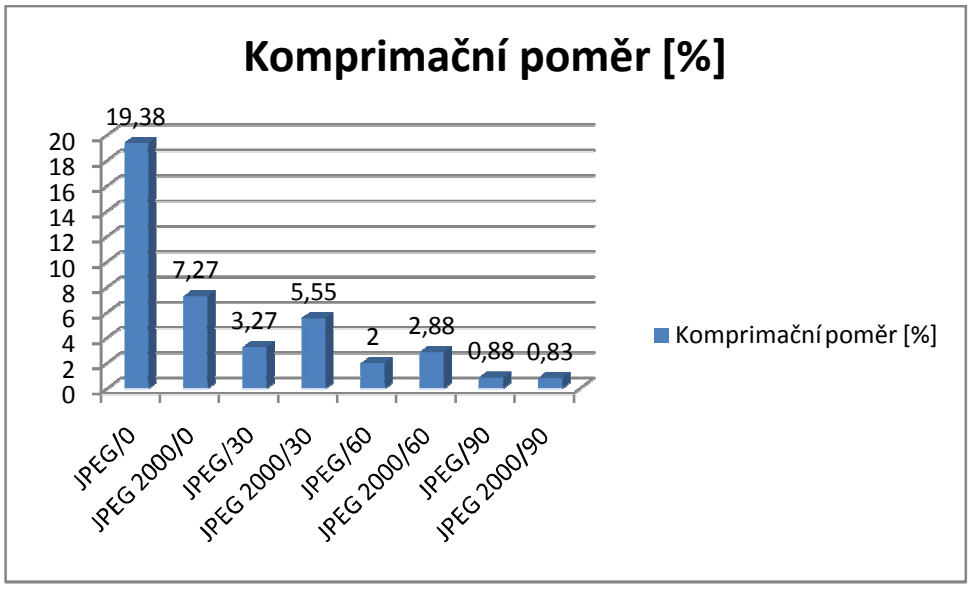
Formát souboru	Metoda komprimace/stupeň komprimace	Barevná hloubka obrázku	Velikost základního obrázku (kB)	Velikost zkomprimovaného obrázku (kB)	Komprimační poměr (%)
BMP	RLE	8 bitů	7991	6850	85,72
GIF	LZW	8 bitů	7991	2034	25,45
PNG	LZ77	24 bitů	23970	12558	52,39
TIFF	LZW	24 bitů	23970	12023	50,16
PCX	RLE	24 bitů	23970	20146	84,05
JPEG	JPEG/0	24 bitů	23970	4646	19,38
JPEG	JPEG/30	24 bitů	23970	783	3,27
JPEG	JPEG/60	24 bitů	23970	480	2,00
JPEG	JPEG/90	24 bitů	23970	212	0,88
JPEG 2000	JPEG 2000/0	24 bitů	23970	1736	7,24
JPEG 2000	JPEG 2000/30	24 bitů	23970	1331	5,55
JPEG 2000	JPEG 2000/60	24 bitů	23970	691	2,88
JPEG 2000	JPEG 2000/90	24 bitů	23970	199	0,83

U obrázku vyfoceného fotoaparátem s velkým rozlišením 8 megapixelů, je použití bezztrátových komprimačních metod zcela nevhodné. Z formátů souborů využívající těchto metod má nejlepší komprimační poměr formát GIF, tento formát využívá pouze 8 bitovou barevnou hloubku, z těch co využívají 24 bitovou barevnou hloubku má nejlepší komprimační poměr formát TIFF, který stejně jako GIF využívá metodu LZW. Vhodnější pro komprimaci tohoto druhu obrázku jsou komprimační metody ztrátové. U nich můžeme využít toho, že lidské oko není schopno rozpoznat malé chyby v obraze jako je rozostření, má jen určitou schopnost rozpoznání barev a jiné. Nejlepší volbou pro komprimaci tohoto typu obrázku je formát JPEG se stupněm komprimace 30 nebo formát JPEG 2000 se stupněm komprimace 60. Vzhledem ke kompatibilitě s ostatními programy je lepší zvolit formát JPEG. Při již zmíněném stupni komprimace 30 má výborný komprimační poměr 3,27 a kvalita je velice uspokojivá. Velikost souboru 783 kB je již bez problému vhodná pro prezentaci na internetu nebo uložení domácí fotogalerie bez obavy z nedostatku místa na disku. Komprimovali-li bychom dále metodami JPEG nebo JPEG 2000 stupni vyššími

než jsou zmiňované, komprimační poměr bude dále klesat, zde již je to na úkor kvality výsledného obrázku.



Obrázek 39: Graf porovnání komprimačního poměru u bezztrátových metod u obrázku pořízeného fotoaparátem s vysokým rozlišením



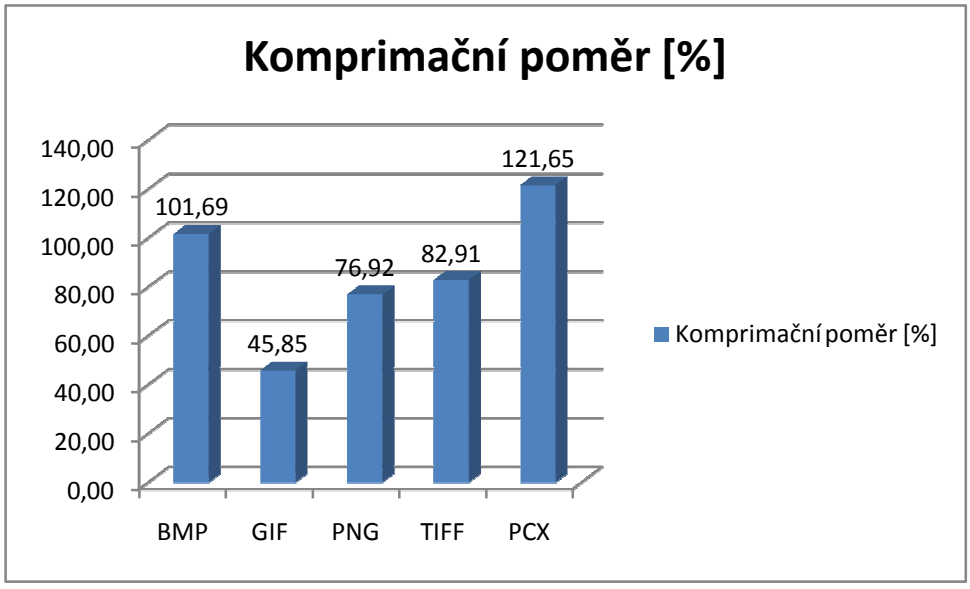
Obrázek 40: Graf porovnání komprimačního poměru u ztrátových metod u obrázku pořízeného fotoaparátem s vysokým rozlišením

7.3 Obrázek s malými rozměry a textem

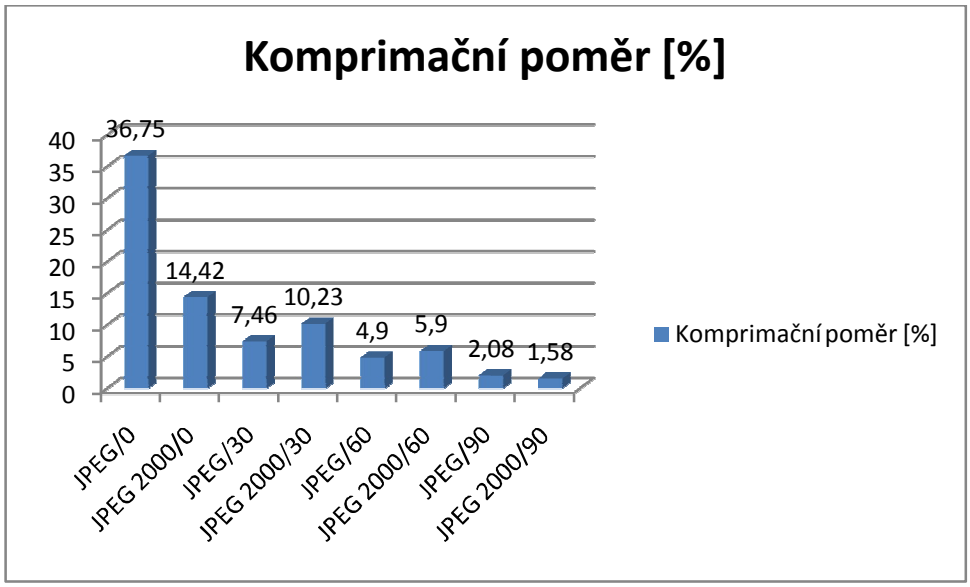
Tabulka 7: Porovnání velikostí výsledných souborů u obrázku s malými rozměry a textem

Formát souboru	Metoda komprimace/stupeň komprimace	Barevná hloubka obrázku	Velikost základního obrázku (kB)	Velikost zkomprimovaného obrázku (kB)	Komprimační poměr (%)
BMP	RLE	8 bitů	118	120	101,69
GIF	LZW	8 bitů	118	54,1	45,85
PNG	LZ77	24 bitů	351	270	76,92
TIFF	LZW	24 bitů	351	291	82,91
PCX	RLE	24 bitů	351	427	121,65
JPEG	JPEG/0	24 bitů	351	129	36,75
JPEG	JPEG/30	24 bitů	351	26,2	7,46
JPEG	JPEG/60	24 bitů	351	17,2	4,90
JPEG	JPEG/90	24 bitů	351	7,31	2,08
JPEG 2000	JPEG 2000/0	24 bitů	351	50,6	14,42
JPEG 2000	JPEG 2000/30	24 bitů	351	35,9	10,23
JPEG 2000	JPEG 2000/60	24 bitů	351	20,7	5,90
JPEG 2000	JPEG 2000/90	24 bitů	351	5,55	1,58

U tohoto typu obrázku se poprvé setkáváme s expanzí. Přihodilo se to u formátů souboru BMP a PCX, obě byli shodně komprimovány metodou RLE. U všech bezztrátových metod byl obraz kvalitní a roven základnímu, nejlepší komprimační poměr s metod bezztrátových má LZW v kombinaci s 8 bitovým GIFem. Komprimační poměr 45,85 % je přesto nedostačující. Bezztrátové metody jsou tedy obecně pro tento typ obrázku nevhodné. Vhodné naopak jsou metody ztrátové, kde je již při stupni komprimace 0 u JPEGu komprimační poměr 36,75 nebo dokonce u JPEGu 2000 14,42 %. Celkově jsou komprimační poměry u těchto metod výborné. Optimální komprimace je JPEGem se stupněm komprimace 30, nebo JPEG 2000 se stupněm komprimace 30. Jak již je psáno výše lidské oko nedokáže rozeznat všechny nedostatky obrázku, a proto komprimace JPEG splňuje požadavky na uspokojivou kvalitu obrazu s výborným komprimačním poměrem. Komprimace s vyššími stupni jako jsou 80-100 mají výborný komprimační poměr, ale vyžití je takřka nulové kvůli nedostačující kvalitě výsledného obrazu.



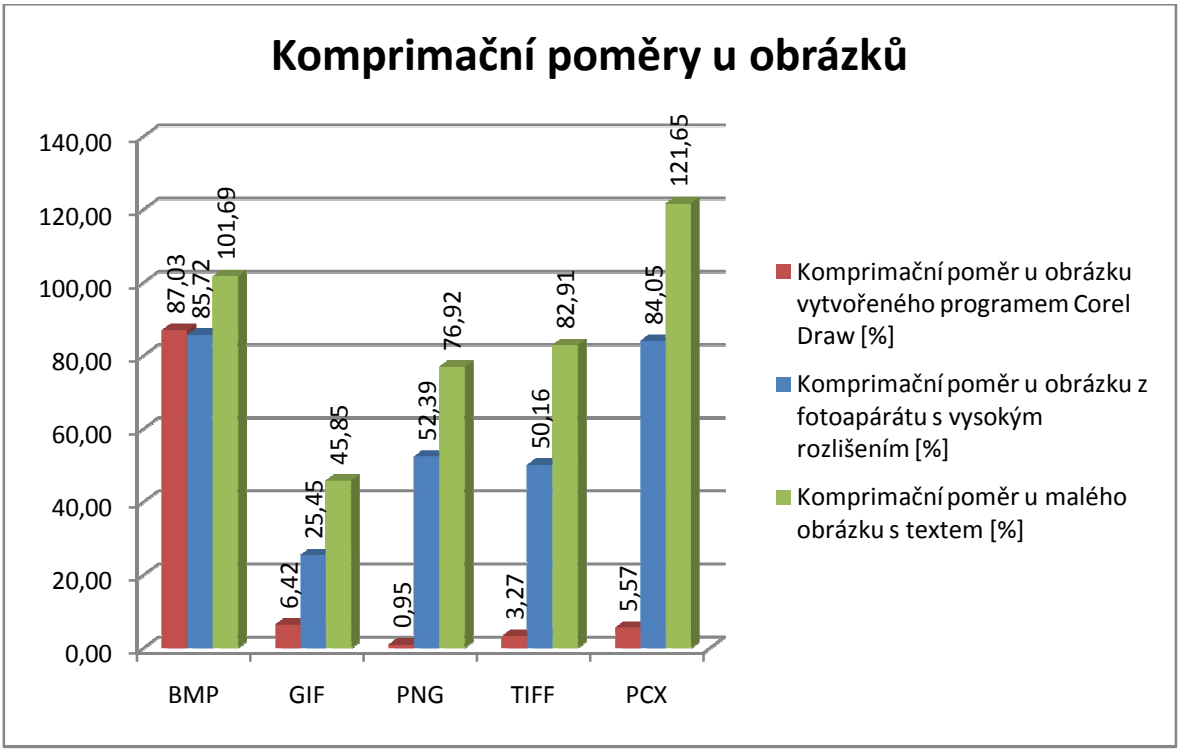
Obrázek 41: Graf porovnání komprimačního poměru u bezztrátových metod s malou velikostí a textem



Obrázek 42: Graf porovnání komprimačního poměru u bezztrátových metod s malou velikostí a textem

Formát obrázku BMP s komprimační metodou RLE není z hlediska velikosti výsledného souboru příliš vhodný, komprimace u žádného s typů obrázků nedosáhla dobrého výsledku, dokonce u malého obrázku s textem byl výsledný soubor po komprimaci větší než před ní. Formát GIF s metodou LZW dosáhl nejlepšího celkového výsledku ze všech bezztrátových komprimačních metod. U žádného s obrázků ale nijak zvlášť nevyčníval, jen snad u obrázku vytvořeného programem Corel Draw měl výsledek vynikající, ale oproti ostatním metodám slabý. Formát obrázku PNG s komprimační metodou LZ77 velice zaujal

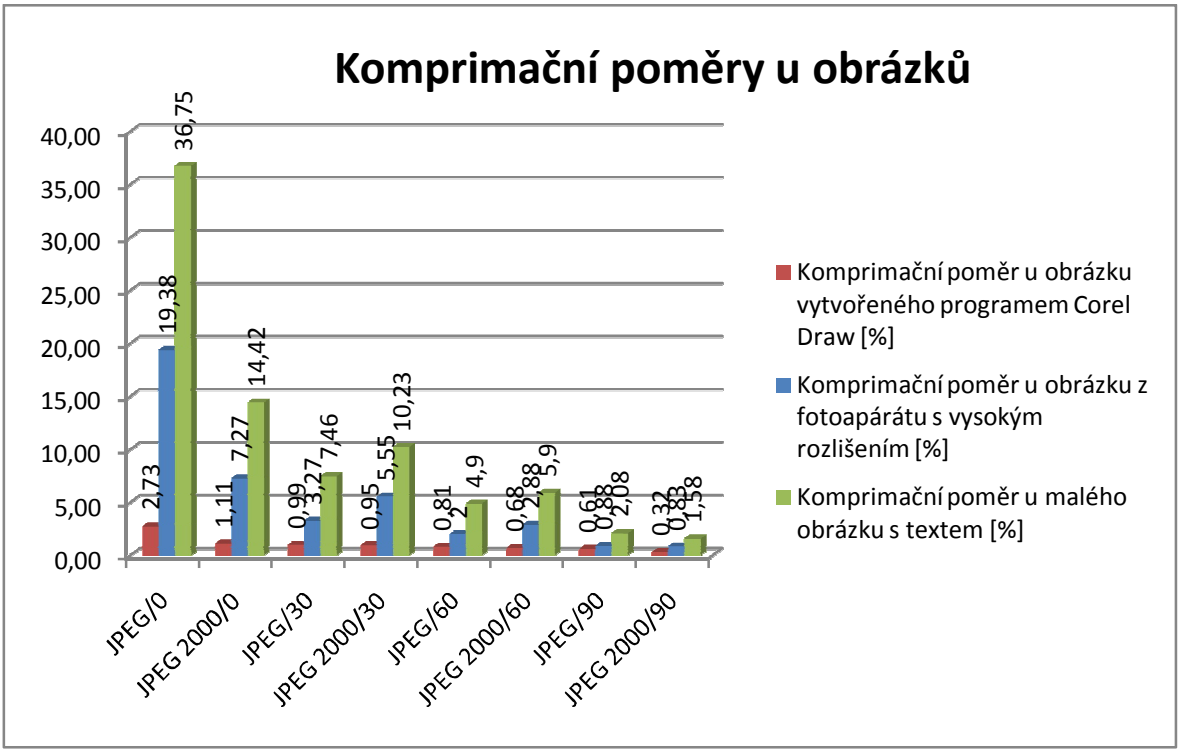
při komprimaci obrázku vytvořeného programem Corel Draw, jeho komprimační poměr byl ze všech metod nejlepší. TIFF s komprimační metodou LZW také nijak zvlášť nevyčníval z řady bezztrátových komprimací, tak jako ostatní bezztrátové metody má výborné výsledky při komprimaci jednoduchého obrázku z Corelu. Má druhý nejlepší výsledek mezi bezztrátovými metodami v komprimaci fotografie a vůbec nejlepší mezi metodami podporujícími 24 bitovou hloubku. Poslední bezztrátovou komprimací je RLE uložena do formátu PCX, tato metoda je nevhodná pro malé fotografie s velkými detaily a byla nejhorší i při komprimaci fotografie s velkým rozlišením.



Obrázek 43: Souhrn komprimačních poměrů u obrázků různých formátů používajících bezztrátovou komprimaci

Komprimační metoda JPEG při stupni komprimace 0 má sice vysokou kvalitu obrazu, ale komprimační poměr zvláště u malého obrázku s textem je špatný. JPEG 2000 má při stejném komprimačním stupni znatelně lepší výsledky, má o více než polovinu lepší tento poměr. Situace se otáčí při stupni komprimace 30. Zde dosahuje o něco lepších výsledku v komprimačním poměru klasický JPEG před JPEGem 2000. Proto je asi celkově nejlepší volbou vezmeme-li v potaz aspekty jako jsou třeba kompatibilita s ostatními programy, již zmíněný komprimační poměr a dobrá kvalita. JPEG a JPEG 2000 s ostatními stupni

komprimace se již u komprimačního poměru liší jen málo. Těchto stupňů komprimace se v praxi využívá jen velmi málo, kvůli špatné kvalitě.



Obrázek 44: Souhrn komprimačních poměrů u obrázků různých formátů používajících ztrátovou komprimaci

ZÁVĚR

Komprimace obrázků je v dnešní době velice důležitá, pro uchovávání a prezentaci obrázků jak na internetu, tak na různých nosičích nebo elektronických albech. Bez komprimovaných obrázků by nemohlo na internetu fungovat téměř nic, od elektronické pošty až reklamu.

Hlavním úkolem komprimace je zachování co nejlepší kvality obrázku, který bude zabírat na nosiči co nejméně místa. To v dnešní době jde mnoha různými algoritmy a způsoby, které odstraňují nadbytečné informace v obrázku. Tím obrázky zjednodušují a díky tomu je pak možné s nimi i jednodušeji pracovat, než by tomu bylo u obrázků nekomprimovaných.

Tato práce se zabývá především porovnávání metod komprimace obrázků. Je zde velice dobře vidět že každá metoda má své pro a proti. Bezztrátové metody komprimace jsou využitelné především u obrázků, které se skládají z jednoduchých čar, takzvané čárové grafiky. Zde dosahují vynikajících výsledků, snad kromě ukládání do formátu BMP v využitím komprimace RLE, která měla komprimační poměry špatné u všech typů obrázku a dokonce jako v jediné z bezztrátových metod docházelo k deformaci obrazu. V praxi se ale více setkáváme s obrázky složitějšími, jako jsou fotografie nebo různé složité kresby. V těchto případech je vhodné využít některou ze ztrátových komprimačních metod. Které dosahují u těchto typů obrázků znatelně lepších komprimačních poměrů než bezztrátové. Jelikož lidské oko neumí rozeznat takovou paletu barev a zachytit takové množství bodů, jako umí počítač, některé obrázky mohou být komprimovány ve výborné kvalitě s výborným komprimačním poměrem.

Bakalářská práce je členěna do několika základních okruhů. Po všeobecném úvodu do komprimace pokračuje vysvětlením základních pojmů, které je u komprimace potřeba znát. Dále pak rozděluje jednotlivé komprimační algoritmy do skupin a představuje přímo jednotlivé metody, algoritmy a formáty ukládání obrázků. Nakonec prakticky ukazuje jednotlivé metody a porovnává je z hlediska kvality a velikosti výsledného obrázku. Porovnávání je většinou vyjádřeno slovně, obrázkem nebo tabulkou.

Hlavní přínos své bakalářské práce spatřuji hlavně v tom, že může pomoci ve studiu počítačové grafiky a podobným oborům. Studenti i ostatní čtenáři v ní najdou poměrně rozsáhlý teoretický základ, co se týče otázek komprimace u obrázků a nakonec i praktickou ukázku využití, těchto komprimací.

ZÁVĚR V ANGLIČTINĚ

Compression of pictures is very important in these days, for instance for storing and presenting pictures on the Internet, media, or electronic galleries. Almost nothing on the Internet, from the e-mail to advertising, could work without the compression of pictures.

The main aim of compression is to maintain the best possible quality of the picture, which would take up as little space as possible on the medium. This can be done, using many algorithms and ways that replace the superfluous information from the picture. The pictures are simplified by this, and thanks to it, it is possible to work more easily with them, than it would be with uncompressed pictures.

This thesis is primarily concerned with the comparison of the compression methods. Here, we can see that each of the methods has some advantages and disadvantages. The compression methods with no loss of information are useful for pictures that are composed of simple lines, the so called line-graphics. The methods reach outstanding results here, except for storing in the BMP format with the use of RLE compression, which had bad compression ratios with all the picture types. Moreover, it was the only method with no loss of information by which the pictures were deformed. In everyday life, we often encounter more complicated pictures like photographs or various complicated drawings and designs. In these cases, it is convenient to use one of the compression methods with loss of information, that reach visibly better compression ratios with these kinds of pictures than the methods with no loss of information. Because the human eye cannot distinguish such a broad colour scale and capture such a high number of dots (unlike the computer), some pictures can be compressed in a great quality with a great compression ratio.

The bachelor thesis is divided into several basic topics. The general introduction to compression is followed by the explanation of basic terms that are necessary to understand compression. Further, the thesis divides compression algorithms into groups and presents the methods, algorithms and formats of pictures. Lastly, it demonstrates the methods themselves and compares them with respect to the quality and size of the final picture. The comparison is mostly expressed verbally, with a picture, or with a chart.

I observe the main contribution of my bachelor thesis in the fact that it can help during the studies of computer graphics, as well as in other fields. Students and other readers will find a relatively extensive theoretical basis about the compression of pictures and in the end, a practical demonstration of the use of compression.

SEZNAM POUŽITÉ LITERATURY

[1] MORKES D.: *Komprimační a archivační programy*. Computer Press, Brno, 1998.

[2] WIKIPEDIA.: *Ztrátová komprese [online]*. 2010 [cit. 2010-04-19]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Ztratova_komprese

[3] MURRAY J., VAN RYPER W.: *Encyklopedie grafických formátů.* Computer Press, Praha 2000.

[4] ČAPEK J., FABIAN P.: *Komprimace dat - principy a praxe.* Computer Press, Praha, 2000.

[5] SALOMON D.: *Data compression. 3rd edition.* Springer, New York 2004.

[6] TIŠNOVSKÝ P.: *PCX prakticky - implementace komprimace RLE [online]*. 2006 [cit. 2010-04-22]. Dostupný z WWW: http://www.root.cz/clanky/pcxprakticky-implementace-komprimace-rle/

[7] CODEPEDIA: *Huffman Trees for Data Compression [online]*. 2006 [cit. 2010-04-25]. Dostupný z WWW: http://www.codepedia.com/1/Art_Huffman_p1

[8] NOVAK T.: *Aritmetické kódování [online]. 1997* [cit. 2010-04-24]. Dostupný z WWW: http://atrey.karlin.mff.cuni.cz/~tnovak/compress/arit/

[9] NOSEK A.: *Implementace kompresní metody PPM.* Praha, ČVUT FEL. Diplomová práce. 2006, Dostupný z WWW: https://dip.felk.cvut.cz/browse/pdfcache/noseka1_2006dipl.pdf

[10] TEZAUR R.: *Jak se obrázky zapisují [online]*. 2001 [cit. 2010-04-29]. Dostupný z WWW: http://www.paladix.cz/clanky/jak-se-obrazky-zapisuji.html

[11] TIŠNOVSKÝ P.: *Grafický formát SVG a animace [online]*. 2007 [cit. 2010-04-30]. Dostupný z WWW: http://www.root.cz/clanky/graficky-format-svg-a-animace/

[12] KALUŽA R.: *JPEG 2000 [online]*. 2006 [cit. 2010-05-01]. Dostupný v www: http://radovan.bloger.cz/IT-internet/JPEG-2000

[13] TIŠNOVSKÝ P.*: JPEG, král rastrových grafických formátů [online]*. 2007 [cit. 2010-04-30]. Dostupný z WWW: http://www.root.cz/clanky/jpeg-kral-rastrovych-grafickych-formatu/

[14] Vlček K.: *Komprese a kódování zabezpečení*, edice BEN, Praha, 2004.

[15] Prchal J, Šimák B.: *Digitální zpracování obrazu v telekomunikacích.* Vydavatelství ČVUT, Praha, 2001.

[16] Levoj M.: *Polygon-Assisted JPEG and MPEG Compression of Synthetic Images,* Center of integrated systems, Stanford University, 1995.

[17] Savakis A. E.: Evaluation of lossless compression methods for grey scale document images, Department of Computer Engineering, Rochester Institute of Technology, dostupný z WWW: http://www.ce.rit.edu/asavakis/papers/ICIP00_savakis.pdf

[18] Grigorov L., Abolmaesumi P.: *Region-based method for visually lossless image compression,* Queen's University, Kingston, Ontario, Kanada, dostupný z WWW: http://research.cs.queensu.ca/home/grigorov/ICIP04-first.pdf

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

bit	z anglického binary digit, základní a současně nejmenší jednotka informace.
byte	jednotka množství dat v informatice, zpravidla označuje osm bitů. V češtině se může používat fonetický zápis bajt.
XML	obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C.
kB	jednotka množství dat, jeden kilobajt označuje 1024 bajtů
MB	jednotka množství dat, jeden megabajt označuje 1024 kilobajtů
RGB	barevný model, který se skládá ze tří základních barev: červená, zelená a modrá.
CMYK	barevný model skládající se z azurové, purpurové, žluté a černé barvy
YUV	barevný model, který k popisu používá tři složky, jednu jasovou a dvě barevné
YCbCr	shodné s YUV
CSS	znamená kaskádové styly, je to jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: OBRÁZKY VYTVOŘENÉ V PROGRAMU COREL DRAW, S RŮZNÝMI METODAMI KOMPRIMACE

Příloha je elektronická. Nachází se v kořenovém adresáři CD-Romu ve složce přílohy a dále je to složka PI-Obrázek vytvořený v Corel Draw. Jednotlivé obrázky jsou pojmenovány názvem typu souboru, popřípadě u typu JPEG a JPEG 2000 ještě číslem, které označuje stupeň komprimace.

PŘÍLOHA P II: OBRÁZKY S VELKÝMI ROZMĚRY S RŮZNÝMI METODAMI KOMPRIMACE

Příloha je elektronická. Nachází se v kořenovém adresáři CD-Romu ve složce přílohy a dále je to složka PII-Obrázek velkých rozměrů. Jednotlivé obrázky jsou pojmenovány názvem typu souboru, popřípadě u typu JPEG a JPEG 2000 ještě číslem, které označuje stupeň komprimace.

PŘÍLOHA P III:OBRÁZKY S MALÝMI ROZMĚRY A TEXTEM, S RŮZNÝMI DRUHY KOMPRIMACE

Příloha je elektronická. Nachází se v kořenovém adresáři CD-Romu ve složce přílohy a dále je to složka PIII-Obrázek s malým rozlišením. Jednotlivé obrázky jsou pojmenovány názvem typu souboru, popřípadě u typu JPEG a JPEG 2000 ještě číslem, které označuje stupeň komprimace.