UNIVERZITA PARDUBICE
DOPRAVNÍ FAKULTA JANA PERNERA

METODY KOMPRESE OBRAZU

Michal Taraba

Bakalářská práce

2013

Univerzita Pardubice
Dopravní fakulta Jana Pernera
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal Taraba**
Osobní číslo:	**D08399**
Studijní program:	**B3709 Dopravní technologie a spoje**
Studijní obor:	**Dopravní infrastruktura: Elektrotechnická zařízení v dopravě**
Název tématu:	**Metody komprese obrazu**
Zadávající katedra:	**Katedra elektrotechniky, elektroniky a zabezpečovací techniky v dopravě**

Zásady pro vypracování:

1. Algoritmy zpracování obrazového signálu
2. Způsoby ukládání obrázků a metody komprese obrazu
3. Programy na kompresi
4. Příklady komprese využívané v SW pro zpracování obrazu
5. Vlastní program na kompresi obrazu ve vybraných formátech
6. Závěry a doporučení

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

Vlček Karel: Komprese a kódová zabezpečení v multimediálních komunikacích
http://www.cs.vsb.cz/benes/vyuka/pte/texty/komprese/index.html
http://www.netcam.cz/encyklopedie-ip-zabezpeceni/standardy-komprese-videa.php
http://www.imagecompression.info/

Vedoucí bakalářské práce: **Ing. Zdeněk Němec, Ph.D.**
Katedra elektrotechniky

Datum zadání bakalářské práce: **17. prosince 2012**
Termín odevzdání bakalářské práce: **31. května 2013**

prof. Ing. Bohumil Culek, CSc.
děkan

L.S.

doc. Ing. Radovan Doleček, Ph.D.
vedoucí katedry

V Pardubicích dne 25. února 2013

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou v seznamu použité literatury.

Byl jsem seznáme s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 25. 5. 2013

Michal Taraba

Poděkování

Tímto bych chtěl poděkovat panu Ing. Zdeňku Němcovi, Ph.D. za cenné rady, připomínky a čas, který této práci věnoval.

ANOTACE

Tato bakalářská práce se zabývá kompresí obrazu. V teoretické části jsou popsány jednotlivé algoritmy, metody komprese a způsob ukládání obrázků. Dále budou představeny vybrané programy na kompresi, jejich porovnání a ukázka provedených jednotlivých kompresí. Praktická část se zaměřuje na vytvoření jednoduchého programu na kompresi obrazu s možností procentuálního stupňování komprese.

KLÍČOVÁ SLOVA

komprese, obraz, algoritmus, grafika

TITLE

Methods of image compression

ANNOTATION

This thesis deals with the image compression. The theoretical part describes the different algorithms, compression methods and how to save images. In addition, selected programs will be presented to compress, and their comparison sample made of compression. The practical part is focused on creating a simple program to compress the image with a comparison of the percentage of compression.

KEYWORDS

compression, image, algorithm, graphics

Obsah:

Úvod

Komprese se používá pro kódování dat s cílem zmenšit jejich objem tím, že se z nich odstraní nadbytečné informace anebo se data nově uspořádají. Hlavním cílem kompresí je dosáhnout toho, aby překódovaná zpráva byla menší než původní zpráva. Data musí být možno zpětně převést tzv. dekódovacím algoritmem.

Pro kompresi obrazu se používají matematické algoritmy, z kterých po té vzniknou kompresní metody. Algoritmus komprese se skládá z procesu kódování a dekódování. Mezi procesem kódování a dekódování se používá stejný postup, jenom v opačném pořadí.

Hlavním cílem mé bakalářské práce je seznámit čtenáře s problematikou a používáním kompresních programů.

V teoretické části se popisuje celý postup komprese, počínaje algoritmem a metodami komprese, až po ukázku funkčnosti jednotlivých programů. Samozřejmostí bude rozdělení a popis vlastností nejpoužívanějších grafických formátů. Čtenář by si měl odnést informace o celém postupu převodu obrázků, a získat všeobecné informace co se týče této problematiky.

Komprese obrazu je nedílnou součástí úspory místa např. ve Vašem počítači. Když budeme uvažovat, že při dnešních možnostech není problém, aby jedna fotografie z digitálního fotoaparátu měla klidně v průměru i 5MB, tzn., že při uložení např. 10000 fotografií, už ta spotřeba místa může být značná, nemyslíte? Zkuste se zamyslet, kolik máte doma fotografií? Další problém, se kterým se setkávám celkem často, je odesílání fotek prostřednictvím sítě Internet. V lepším případě je lze odeslat s delší časovou prodlevou, a vtom horším to musíme posílat na vícekrát, např. emailem.

Úkolem praktické části je vytvořit jednoduchý program na kompresi obrazu v jazyce C#. Názorně si zde ukážeme, jak je snadné si upravit obrázek při zachování kvality oku nerozeznatelné.

1 Algoritmy zpracování obrazového signálu

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. S pojmem algoritmus se můžeme setkat v jakémkoli vědeckém odvětví. Za jistý druh algoritmu můžeme považovat i např. kuchařský recept. Nejčastěji se s ním setkáme při programování, jako teoretický postup při řešení problému. Algoritmus a samostatné postupy musejí splňovat předem daná kritéria.[3]

1.1 Vlastnosti algoritmů

Na vstupní a výstupní podmínky se kladou specifické požadavky jako je jednoduchost, jasnost, jednoznačnost a další. Algoritmus pracuje s nějakými vstupy (veličinami), které dostane před začátkem činnosti anebo v jeho průběhu. Má alespoň jeden výstup, který je v požadovaném vztahu ke vstupům a tím zjistíme výsledek problému, který řešíme. Algoritmy musejí splňovat předem daná kritéria. Můžeme je definovat podle základních vlastností.[3][14]

1.1.1 Konečnost

Každý algoritmus musí skončit v konečném počtu kroků. Není podstatné jak je krok velký, ale vždy musí mít konečnou hodnotu. Existují i případy kdy toto nefunguje, takové postupy nazýváme výpočetní metody. Jako příklad výpočetní metody si uvedeme tzv. reakční proces. Tento proces mění své vlastnosti během svého života v závislosti na změnách v okolním prostředí. V některých literaturách se setkáme s tím, že i takový postup se zahrnuje mezi algoritmy.[14]

1.1.2 Obecnost (Hromadnost)

Algoritmus nemá za úkol řešit jeden konkrétní problém (např. jak vypočítat 2x3; 3+5 atd.). Musíme se na to dívat jako na celek, algoritmus má za úkol řešit celou řadu takovýchto problémů (např. jak je možné vypočítat součin dvou čísel; jak je možné vypočítat součet dvou čísel atd.). Má na to k dispozici mnoho možných vstupů.[14]

1.1.3 Determinovanost

Každý jednotlivý krok algoritmu musí být jednoznačně a přesně definován. Vždy musí být předem jasně dané, co a jak je třeba udělat a jak má algoritmus vypadat. Nelze se spoléhat na běžný jazyk, protože neposkytuje přesnost a jednoznačnost ve vyjadřování, a z tohoto důvodu byly pro zápis algoritmů vytvořeny tzv. programovací jazyky. V programovacích jazycích má každý krok jednoznačně definovaný význam. Výpočetní metodu, kterou vytvoříme v programovacím jazyce, nazýváme program.[14]

1.1.4 Resultativnost (výstup)

Resultativnost platí, když řešený algoritmus má alespoň jeden výstup. Každý výstup musí být v požadovaném vztahu k některému ze vstupů, aby se dalo jednoznačně říct, ke kterému vstupu se vztahuje výsledek.[14]

1.2 Druhy algoritmů

Algoritmy [5] je možné rozdělovat různými způsoby. Mezi důležité algoritmy patří tzv.: rekurzivní algoritmy, pravděpodobnostní algoritmy, paralelní algoritmy, genetické algoritmy a heuristické algoritmy. Přičemž jeden určitý algoritmus může patřit zároveň do více skupin, např.: je možné, aby zároveň patřil mezi rekurzivní i genetický.[14]

1.2.1 Rekurzivní algoritmy

Rekurzivní algoritmy [14] využívají (volají) sami sebe. Rekurze se např. využívá pro definici přirozených čísel. Při programování se bavíme o tzv. rekurzivní funkci, tj. opakované vnořené volání podprogramu. Rekurzivní funkce obsahuje podmínku, kdy se má vnořování zastavit. Po každém kroku co rekurzivní algoritmy volají sami sebe, se musí problém postupně zjednodušovat, až do doby, kdy nastane koncová situace.

Rekurzivní algoritmy se dělí na dva základní typy. Prvním typem je Přímá rekurze, ta nastává, když program volá sám sebe. Druhým typem je Nepřímá rekurze, o ní se bavíme, když vzájemné volání podprogramů vytvoří

kruh (např. funkce A volá funkci B, funkce B volá funkci C a funkce C volá funkci A).

Hlavní výhodou rekurzivních algoritmů je jejich přehlednost a jednoduchost. Naopak jejich nevýhodou je, že díky velkému počtu vnoření, zabere celkem velké množství paměti.

1.2.2 Pravděpodobnostní algoritmy

Pravděpodobnostní algoritmy [35] patří do skupiny nedeterministických algoritmů. Nedeterministické algoritmy se mohou rozhodovat mezi několika možnostmi dalších kroků, které provedou. Pravděpodobnostní algoritmus se snaží najít řešení rychleji anebo řeší těžko řešitelný problém. Může se rozhodovat mezi různými možnostmi, jak má pokračovat. Z toho plyne, že pro stejný vstup může dávat různé výsledky, ale v jiném čase. Nejčastěji se algoritmus spustí vícekrát, aby se určil správný výsledek. Tyto algoritmy jsou většinou velice jednoduché, ale jejich analýza v závislosti na čase je složitá.

1.2.3 Paralelní algoritmy

Paralelní algoritmy [5] se uplatňují v případě, že máme k dispozici více počítačů a je možné danou úlohu rozdělit mezi ně. Paralelně programovaný software umí rozdělit jeden velký výpočetní problém na několik menších problémů. Dané výpočty jsou řešeny souběžně, např. jde buď o jeden počítač s více procesory, několik počítačů v síti, speciální hardware nebo kombinaci některého z uvedených prvků. Důležitým parametrem u těchto algoritmů je synchronizace jednotlivých výpočtů. Špatná synchronizace může vést k nesprávné funkci programu, např. k deadlocku. Deadlock odpovídá situaci, při které je úspěšné dokončení první akce, podmíněno dokončením druhé akce a dokončení druhé akce je podmíněno ukončením první akce, vzniká nám paradox.

1.2.4 Genetické algoritmy

Genetické algoritmy [34] napodobují biologické evoluční procesy, postupným vytvářením těch nejlepších řešení pomocí mutací a křížení. Je to

tzv. heuristický postup, který se snaží nalézt řešení složitých problémů, pro který není exaktní algoritmus.

Heuristika je řešení problému, pro který neznáme algoritmus nebo přesnější metodu. Toto řešení je založené buď na intuici, odhadu anebo na zkušenostech. Proto je výsledek často jen přibližný. Odhad se může i nemusí zlepšovat, tudíž nám nezaručuje vždy to nejlepší řešení, ale na druhou stranu je jednoduchý, použitelný a rychlý.

Princip genetického algoritmu spočívá v postupné tvorbě generací různých řešení daného problému. Uchovává se tzv. populace, ve které je ke každému jedinci přiřazeno alespoň jedno řešení problému. Řešení se zlepšují, když populace probíhá evolucí. Většinou je řešení tvořeno binárními čísly, ale zřídka se setkáme i se stromem, maticí, atd. První generace populace se skládá z náhodných členů. Při přechodu do další generace je každému stávajícímu jedinci vypočítaná funkce, která vyjadřuje kvalitu řešení tímto jedincem. Podle tohoto se vyberou noví jedinci a díky mutacím a křížení vznikne nová populace. Tento postup se opakuje do té doby, dokud nebude kvalita řešení na dostačující úrovni.

1.2.5 Heuristické algoritmy

Heuristické algoritmy [33] využívají také heuristiku jako předešlý genetický algoritmus. Tento algoritmus často obsahuje možnost volby pokračování výpočtu, tzn. jaká data, v jakém pořadí a jak budou zpracována. Potom se bavíme o konkrétní strategii. V tomto se s genetickým algoritmem liší, ten nám neurčuje konkrétní strategii. Heuristické algoritmy se používají od začátku výpočetní techniky, kde jsou rozvíjeny a používány pro řešení problémů, zejména pro řešení složitých funkcí s mnoha parametry.[33]

1.3 Správnost algoritmu

Správnost algoritmu [5] nastává v případě, že pro veškeré údaje, které splňují vstupní podmínku, se zpracovávaný proces zastaví a všechny výstupní údaje musí splňovat výstupní podmínku. V praxi se často setkáme s tím, že ověření správnosti algoritmu se zkouší reakcí algoritmu na konečný počet vstupních dat. Samozřejmě toto nám mnoho napoví, ale není dokázáno, že se

nám algoritmus nezhroutí při různé kombinaci vstupních dat. Pro určení správnosti algoritmu není univerzální metoda, algoritmus by měl být dokázán skupinou předem známých operací (kroků), které budou pro všechna data vést ke správnému výsledku. Takový algoritmus je korektním řešením úlohy. Korektní je takový algoritmus, u kterého se nezapomene na žádnou možnost zpracování dat při průchodu algoritmem. Algoritmus je správný, když vydá správný výsledek. Důležité je také ověřit, zda algoritmus po konečném počtu kroků pro všechna data skončí, potom se jedná o konečnost algoritmu.[14]

1.4 Rozdělení algoritmů

Algoritmy [5] dělíme podle způsobu, pro který jsou určeny. Celkem jsou čtyři základní skupiny. Rozdílné jsou v tom, jaké data zpracovávají. Mohou zpracovávat data obrazového charakteru, data zvukového charakteru, data textového charakteru a videa. Jsou také tzv. univerzální algoritmy, ty mohou pracovat s jakýmkoli typem vstupních souborů. Ale v praxi nedosahují takových kompresních poměrů jako specializované algoritmy, proto se neprosadily. Většinou používáme více algoritmů za sebou, z důvodu zlepšení kompresního poměru. Algoritmy rozdělujeme např. do těchto skupin [4]:

- Jednoduché algoritmy
- Slovníkové algoritmy
- Statistické algoritmy
- Transformační algoritmy

Všechny čtyři uvedené algoritmy patří do skupiny tzv. bezeztrátových metod.

1.4.1 Jednoduché algoritmy

Jednoduché algoritmy jsou založeny na principu kódování opakujících se posloupností znaků. Mezi nejznámější patří [14]:

- Run lenght encoding (RLE)[2]
- Potlačení nul
- Bitové mapy
- Půlbajtové kódování

1.4.2 Slovníkové algoritmy

Slovníkové algoritmy [4] vytvářejí v průběhu komprimace slovník na základě zkomprimovaných dat. Potom se ve slovníku snaží vyhledat data, která ještě nebyly komprimovány. V případě, že shodná data najde, algoritmus zapíše pozici dat ve slovníku, místo toho aby komprimoval stejná data vícekrát. Kóduje se celý vzorek namísto jednotlivých znaků.

Slovník je možné vytvořit několika způsoby. Buď je tvořen staticky, nebo dynamicky adaptivními metodami. Dále může být založený na přirozeném jazyku anebo na bázi obecných řetězců. Mezi základní typy patří [16][9]:

- Lempel-Ziv 77 (LZ77)
- Lempel-Ziv 78 (LZ78)
- Lempel-Ziv-Welch (LZW) – podobný LZ78 [9]
- LZMA

1.4.3 Statistické algoritmy

Statistické algoritmy [9] se snaží určitým způsobem předvídat, jaké znaky budou v datech následovat. Pro znaky s vyšší pravděpodobností výskytu vyhradí algoritmus kratší informaci pro jejich zapsání. Naopak pro znaky s nižší pravděpodobností výskytu vyhradí delší informaci pro jejich zápis. Tyto algoritmy lze rozdělit do dvou metod. První metodou je tzv. metoda se statickým modelem. Ta vytvoří před komprimací dat model, podle kterého se zkomprimují veškerá data. Model slouží pro vypočítávání pravděpodobnosti výskytu znaků. Druhá metoda je tzv. metoda s adaptivním modelem. Tato metoda průběžně aktualizuje model. Do statistických algoritmů patří [16]:

- Huffmanovo kódování [9]
- Shannon-Fanovo kódování
- Aritmetické kódování
- Range coding (RC)
- ACB
- Prediction by partial match (PPM)

1.4.4 Transformační algoritmy

Transformační algoritmy ve skutečnosti nic nekomprimují, pouze upravují data, aby se dala lépe zkomprimovat. Algoritmy mohou měnit pořadí a hodnoty jednotlivých prvků. Jsou založeny na lineární transformaci, která převede data z časové oblasti vzorků do frekvenční oblasti. Ke každé transformaci musí existovat opačná transformace, která obnoví původní data. Mezi základní typy patří[16]:

- Move-to-front transformace (MTF I a MTF II)
- Burrows-Wheelerova trransformace (BWT)
- Weighted frequency count (WFC)
- Distance coding (DC)
- Inverse frequency coding (IF)

1.5 Vybrané metody kódování

U kódování je důležitým parametrem např. faktor komprese a kompresní poměr. Faktor komprese udává, jakou část původního prostoru zabírají údaje po kompresi. Kompresní poměr je jeho převrácená hodnota. Čím větší je kompresní poměr, tím máme úspěšnější kompresi.

1.5.1 Metoda RLE

Metoda RLE [2] je založená na principu opakujících se symbolů jdoucích po sobě. Opakující symboly se nahradí dvojicí symbolů, a to kódovaným znakem a počtem jeho opakování. Především se využívá pro obrazovou informaci, pro kompresi textu zřídka.

Kódují se pouze znaky, které se opakují 3 a více za sebou. V případě menšího opakování se znaky pouze opíší.[3]

Př.:

Vstupní data kodéru: MMMMAAMAAAAAMMMMMMAMAAA

Výsledek kódování: 4MAAM5A6MAM3A

Faktor komprese: $= \frac{13}{23} * 100 \cong 57\%$

Kompresní poměr: $= \frac{23}{13} \cong 1{,}77$

1.5.2 Metoda potlačení nul

Jedna z nejstarších metod, pracuje na podobném principu jako metoda RLE, jen s tím rozdílem, že se kódují pouze nuly a mezery. Když budeme mít více stejných znaků za sebou, uvedeme spolu s počtem znaků tzv. indikátor komprimace I_k.[3]

Př.

Vstupní signál: hn---dsssooppp------mm

Výstupní signál: hnI_k3dsssoopppI$_k$6mm

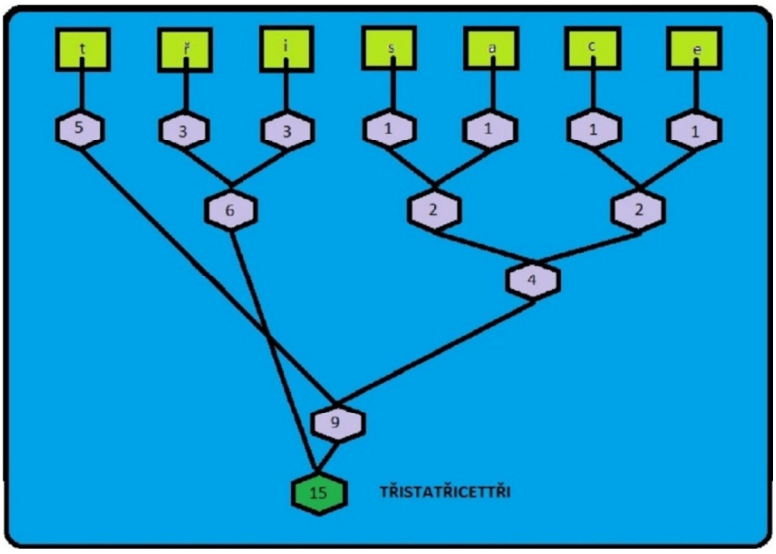
1.5.3 Bitové mapy

Využívají se v datech, kde je mnohokrát uveden jeden znak. Čím více stejných znaků, tím lepší kompresní poměr.[3]

1.5.4 Huffmanovo kódování

Huffmanovo kódování [3] je založeno na principu opakujících se znaků ve vstupním souboru. Je zapotřebí zjistit četnost výskytu jednotlivých znaků. Na základě toho se znakům přiřadí binární hodnota. Znaky s největší četností výskytu, mají krátký binární kód, klidně i jeden znak a naopak znaky s nejmenší četností výskytu mají velký binární kód. Algoritmus využívá binární stromy. Znaky s nejmenší četností se postupně spojují, až vznikne jediný bod. Při komprimaci se postupuje od zdrojových znaků až po ten jediný bod, kód vznikne z 0 a 1 např. tak, že při pohybu po hraně doleva, zapíšeme odzadu 0 a naopak při pohybu doprava zapíšeme 1.[4]

Př. Kódování slova třistatřicettři



Obr. 1 - Huffmanovo kódování

Znak	t	ř	i	s	a	c	e
Kód	01	11	10	0011	0010	0001	0000

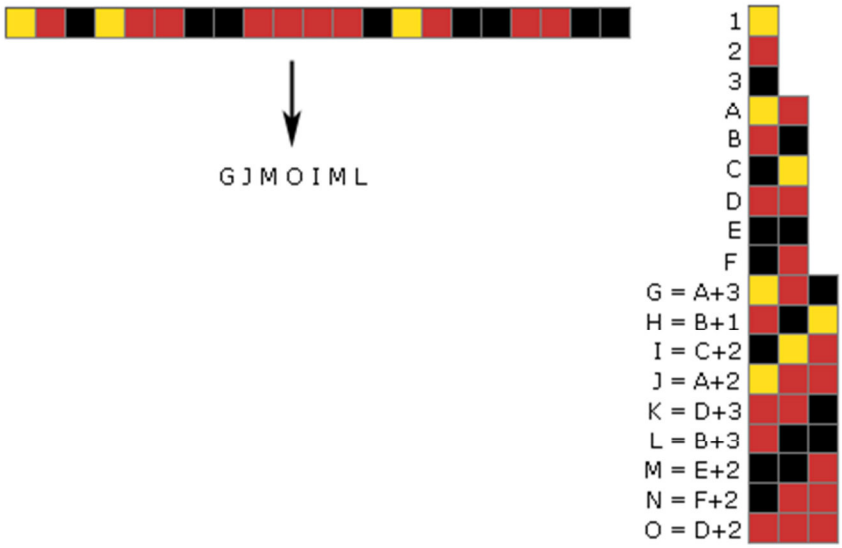
Tab. 1 - Tabulka kódů

Výsledný kód: 011110001101001001111000010000010111110

1.5.5 Lempel-Ziv-Welch kódování

LZW metoda [9] čte informace po řádcích a po jednotlivých pixelech. Má slovník, do kterého ukládá jedinečné symboly, které se opakují. Program projede celý obrázek po jednom pixelu, zaznamená do slovníku jednotlivé symboly a ve výsledku opakující se symboly nahradí binárními kódy. Poté pokračuje se dvěma pixely, opět zaznamená jedinečné dvojice symbolů a dále pracuje s trojicí znaků. Takhle lze pokračovat až do velikosti o 12 členech. Při překročení této hranice se slovník maže a začíná znovu.[7]

Př.



Obr. 2 - LZW metoda [9]

1.5.6 Burrows – Wheelerova transformace

Tato transformace vstupní data nekomprimuje. Pouze změní pořadí jednotlivých symbolů tak, aby to bylo výhodné pro případné další komprimace. Při transformaci určujeme konec symbolu tzv. EOF symbol - @.

Princip metody spočívá ve vytvoření všech možných rotací řetězce. Rotace se seřadí pod sebe a poslední znak z každého řádku tvoří část výstupní informace. Dekódování probíhá tak, že vstup seřadíme do sloupce a ve vedlejším sloupci bude vstup seřazen podle abecedy. Třetí sloupec se vytvoří tak, že před znaky z druhého sloupce se vloží data z prvního sloupce. Čtvrtý sloupec vznikne opět srovnáním dat podle abecedy. Toto vše se opakuje do té doby, dokud nám nevznikne původní slovo.[40]

Př. transformace textového řetězce

Vstupní řetězec	Všechny rotace	Seřazení rotací	Výstupní řetězec
^PALALA@	^PALALA@ @^PALALA A@^PALAL LA@^PALA ALA@^PAL LALA@^PA ALALA@^P PALALA@^	ALALA@^**P** ALA@^PA**L** A@^PALA**L** LALA@^P**A** LA@^PAL**A** PALALA@**^** ^PALALA**@** @^PALAL**A**	PLLAA^@A

Tab. 2 - B-W transformace

Postup zpětné transformace je uveden v příloze A.

2 Metody komprese obrazu a způsoby ukládání obrázků

V současné době dokážeme komprimovat text s kompresním poměrem ½ a obrázky s kompresním poměrem i 1/50. Což nám umožnuje snížit požadavky na počítače. Obrázky a videa dokážeme komprimovat lépe než text. Vycházíme z toho, že lidské oko nebo ucho není dokonalé, tzn., že při komprimaci zvuku si můžeme dovolit vynechat nějaké tóny a při komprimaci obrazu je zase možné vynechat nějakou barvu, aniž bychom si toho všimli. Jedním z nejdůležitějších pojmů této problematiky je pixel. Pixel je nejmenší jednotka digitální rastrové (bitmapové) grafiky. Představuje jeden svítící bod na monitoru, či bod v obrázku.

2.1 Druhy kompresních metod

Hlavním rozdělením komprese je metoda ztrátová a bezeztrátová.

Bezeztrátová metoda spočívá vtom, že data po kompresi se plně shodují s daty před kompresí. Vyvarujeme se úbytku jakýchkoli částí dat. Využívají se např. na text nebo k ukládání dat na některá přenosová média. Zajímavostí je, že na bezeztrátových metodách stojí internet.

Naopak ve ztrátové kompresi se data po kompresi neshodují s daty před kompresí, ale cíl je podobný, aby se data před a po kompresi co nejvíce shodovala. Tato metoda je více efektivní. Při volbě komprese se dá algoritmu určit, jaké informace může zanedbat. Uplatníme jí např. při kompresi zvuku, obrazu a videa. Při zvukové kompresi, se většinou ztratí informace o vyšší frekvenci než 44 KHz. V některých případech tato frekvence může být nižší, např. 22 KHz. U obrázků rozlišujeme, zda se jedná o fotografie (malá ostrost) anebo o geometrické obrázky (velká ostrost). Jestliže se jedná o fotografie, tak pro ně se používá standart JPEG, ten podle nastavení vynechá některé pixely. U geometrických obrázků je komprese náročnější a složitější. Z toho důvodu nedosáhneme tak dobrého kompresního poměru. Při kompresi videí využíváme metody jako při obrazových kompresích. Výsledkem by mělo být video, o co největší kvalitě a při tom chceme, aby soubor byl co nejmenší.

2.2 Bezeztrátové metody

Některé grafické formáty nevyužívají žádnou komprimační metodu. Týká se to např. souborů s koncovkou PBM, PPM, PAM, PGM a WBMP.

Další soubory BMP, PCX a TGA využívají jednoduché kódování typu RLE. A mezi poslední soubory patří GIF a PNG, které využívají kódovací metodu LZ77 nebo LZW.

2.2.1 BMP

BMP [36] nebo také označován jako Windows Bitmap, DIB (device-independent bitmap) nebo Windows DIB je počítačový formát vyvinutý společností Microsoft. Formát vznikl v roce 1988, jeho hlavní výhodou je jednoduchost a volné šíření není omezeno patentovou ochranou, tudíž s ním umí pracovat většina grafických programů. Je určen pro ukládání bitmapové grafiky.

Tyto obrázky se ukládají po jednotlivých pixelech. Abychom rozlišili barevnou hloubku, je zapotřebí zjistit kolik bitů reprezentuje každý pixel, např. když budeme mít 2 barvy tak každý pixel je reprezentován 1 bitem, když bude 256 barev tak bude 8 bitů na pixel a v případě 65536 to bude 16 bitů. Když by to bylo 24 bitů na pixel, tak se bavíme o 16,7 miliónů barev.
Při výpočtu celkové velikosti souboru, musíme vzít v úvahu i hlavičku souboru, která se liší od samotného obrázku.[6]

Vzorec pro výpočet velikosti nekomprimovaného obrázku:

$$(\text{šířka v pixelech}) * (\text{výška v pixelech}) * \left(\frac{\text{bitů na pixel}}{8}\right)$$

např. na obrázek o velikosti 1024x768 a s 16,7 miliony barev potřebujeme přibližně až 2,4 MB místa. Na 1 pixel využíváme 3 byte.

Formáty BMP buď nevyužívají žádnou kompresi anebo používají RLE. Proto je jejich velikost mnohdy daleko větší než souborů, které využívají sofistikovanější kompresní metody. Tento soubor je nevhodný pro použití na síti internet.

2.2.2 PCX

Formát PCX [37] byl vytvořen firmou Zsoft Corporation. Ve své době to byl velice uznávaný zobrazovací DOS standart. Postupem času ho nahradily důmyslnější formáty jako GIF, PNG a JPEG. Používá se buď bez komprese anebo využívá RLE kompresi. Obrázky lze ukládat s barevnou hloubkou 1bit, 4bity, 8bitů nebo 24bitů. Nejprve byl vytvořen pro PC Paintbrush, ale později byl rozšířen i na jiné aplikace.[6]

2.2.3 GIF

GIF [38] (Graphics Interchange Format) využívá kompresi LZW. Je stejně jako BMP určen pro ukládání bitmapové grafiky, ale díky sofistikovanější kompresi LZW je jeho výsledná velikost menší. Největší nevýhodou tohoto formátu je, že lze použít současně pouze 256 barev (8 bitů) barevné palety. Jeho hlavní výhodou je využití na obrazové animace. Je vhodný i např. k čárové grafice (loga), zde není potřeba takové množství barev.[6]

2.2.4 PNG

PNG [39] (Portable Network Graphics) je také určen pro bitmapovou grafiku. Tento formát vznikl jako náhrada formátu GIF. Formát PNG nemá omezení současně použitých barev jako GIF. Jeho největší nevýhodou je, že neumí zobrazovat animace, pro ty se vytvořil formát APNG a MNG. Hlavní důvod přechodu z GIF na PNG byl takový, že formát GIF byl patentově chráněný díky algoritmu LZW. V dnešní době je už patent dávno prošlý.

PNG může mít až 32 bit barevnou hloubku, s tím že obsahuje tzv. alfa kanál, tj. osmibitová průhlednost, tzn., že složka pixelu udává hodnotu průhlednosti vybraného pixelu. Jako příklad lze uvést barevný model RGBA, ten kromě přenášených složek RGB má i složku A, která nám přenáší informaci o průhlednosti daného pixelu. Jev průhlednosti je možné popsat jako situaci, kdy obrázek s definovanou průhledností překrývá další obrázek. Zadní obrázek bude zobrazen s intenzitou pixelu danou průhledností prvního obrázku.[6]

Obr. 3 - Průhlednost PNG [21]

2.3 Ztrátové metody

Princip ztrátové metody spočívá v oddělení důležitých informací od těch nedůležitých, tím že přeskupí nebo transformuje data už pří úvodním zpracování. Důležité informace potlačí méně než ty nedůležité a následně data zkomprimuje bezeztrátovým algoritmem. Mezi ztrátové metody, které využívají ztrátovou kompresi, patří např. JPEG a JPEG2000. Ve výjimečných případech se lze setkat i s formátem JPEG, který využívá bezeztrátovou kompresi.

2.3.1 JPEG

Označení JPEG [15] znamená Joint Photographic Experts Group, což je název skupiny, která tuto kompresi navrhla. Využívá se pro bitmapovou grafiku. Mezi přípony tohoto formátu např. patří: JPG, JPEG, JFIF a JPE. Když se budeme bavit o souboru JPEG, většinou tím máme na mysli formát JFIF. Je to nejčastěji využívaný formát pro práci s fotografiemi a obrázky s velkou barevnou hloubkou na internetu. Formát JPEG je povedený, jeho potenciál práce s obrázky je obrovský, má možnost velké pružnosti co se týče změny velikost, tudíž kvality. Pro takovéto soubory je vhodnější např. PNG nebo GIF.

Formát JPEG má některé nedostatky. Pracuje vždy jen s největší barevnou hloubkou 24 bitů. Nepodporuje průhlednost, tzn., nedokáže pracovat s obrázky na průhledném pozadí. Dále se nehodí pro ukládání textů, grafů a kreseb, má špatné zobrazení ostrých hran, např. písmena jakoby rozmaže. Z tohoto důvodu se při velké JPEG kompresi objevují v obraze tzv. JPEG artefakty neboli umělé nechtěné obrazové prvky. Další negativní vlastností je,

že nepodporuje animace a opakovaným ukládáním, je možné zhoršit vlastnosti fotografií. Samotná JPEG komprese bude popsaná v kapitole 4. [15]

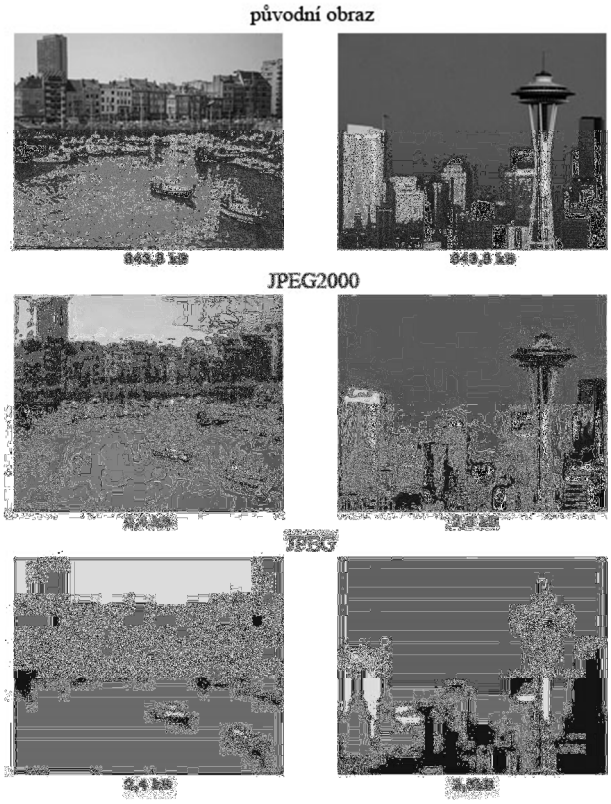
2.3.2 JPEG 2000

U tohoto formátu se užívá přípona např. JP2 a JPX. Při kompresi lze využít ztrátovou i bezeztrátovou metodu. Má kvalitnější obraz než formát JPEG a to při použití stejného kompresního poměru. Cílem tohoto formátu bylo zlepšení vlastností oproti stávajícímu formátu JPEG, jako je např. upravitelnost a efektivnost. Formát JPEG 2000 podporuje velmi vysoké stupně komprese při zachování větší kvality. Při snížení počtu bitů je u JPEGU nutné před zakódováním změnit rozlišení obrázku, u JPEG 2000 to není potřeba, on to dělá automaticky prostřednictvím své vnitřní struktury.

Tento formát má široké využití. Používá se do spotřebitelských aplikací, jako jsou fotoaparáty, mobilní telefony a tiskárny. Dále internet, satelitní snímky, lékařské snímky, velké snímky a další. I když tento formát podporuje bezeztrátové kódování, v dnešní době není schopen plně nahradit formát PNG. PNG je stále prostorově úspornější, hlavně u souborů s mnoha pixely stejné barvy, jako například diagramy. [17]

2.3.3 Srovnání JPEG a JPEG2000

Jak zde již bylo uvedeno, formát JPEG 2000 je lepší jak formát JPEG, hlavně vtom, že při použití stejného kompresního poměru, má výrazně lepší kvalitu obrazu.[8] „ *Pro střední stupeň komprese poskytuje JPEG 2000 asi o 20% lepší kompresi než JPEG, pro nízké a vysoké stupně komprese může být zlepšení ještě větší.*“ [17]. V následujícím obrázku je vidět rozdíl mezi jednotlivými formáty s použitou kompresí 1:350.



Obr. 4 - Srovnání JPEG a JPEG2000 [8]

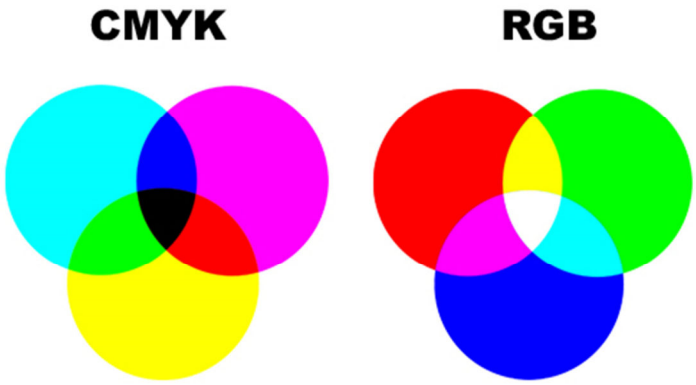
2.4 Způsob ukládání obrazových informací

Mezi dva základní způsoby ukládání obrazových informací patří bitmapová grafika (rastrová grafika) a vektorová grafika. Bitmapovou grafikou se zabýváme tehdy, když pracujeme s nepravidelnými objekty, s velkým množstvím barev (odstínů) a bez zřetelného ohraničení. Vektorovou grafiku využijeme v případě obrázku, ve kterém máme veškeré objekty jasně definované, např. loga, schémata, grafy a podobně. Dalším důležitým rozdělení je, s jakým typem barevného modelu pracujeme. Běžně rozlišujeme dva druhy barevného modelu. Jedná se o model RGB a CMYK. [13]

RGB [31]]je tzv. aditivní způsob míchání barev, ve kterém jsou společně smíchány barvy, jako jsou červená, zelená a modrá, tak aby se zobrazila požadovaná barva. Například bílou barvu získáme smícháním všech tří barev. Názorná ukázka je na obrázku č. 5. S RGB se lez setkat např. v monitorech a projektorech.

Model CMYK pracuje na principu tzv. subtraktivním mícháním barev, tzn., že smíchané barvy od sebe odečítáme. Barevné spektrum je tedy omezováno. Do CMYK patří čtyři základní barvy, azurová (Cyan), purpurová

(Magenta), žlutá (Yellow) a černá (Key), je také znázorněno na obrázku č. 5. Za normálních okolností by stačily jen tři barvy (CMY). Jedním z důvodů proč to tak nemůže být je, že smícháním všech tří barev, nevznikne úplně černá barva, spíše taková tmavě šedivá a další důvod je ekonomický. Cena tří barev je podstatně vyšší než cena samostatné černé barvy. CMYK se používá např. v inkoustových tiskárnách. Konkrétní rozdíly mezi jednotlivými modely jsou vidět na obrázku č. 6. [32][18]



Obr. 5 - Model RGB a CMYK [18]

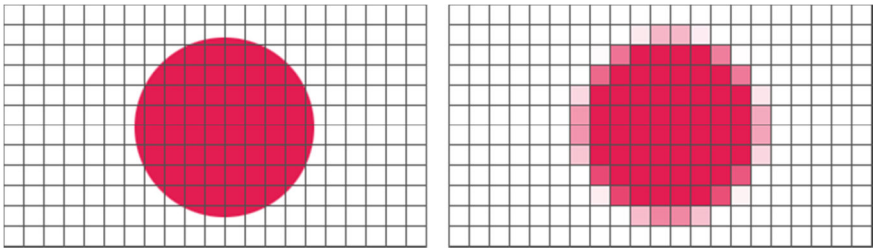
Obr. 6 – Porovnání fotografie mezi RGB a CMYK [18]

2.4.1 Bitmapová grafika

„ *V bitmapové grafice je celý obrázek popsán pomocí jednotlivých barevných bodů (pixelů).* “ [13]. Tyto dané body, mají definovanou svojí polohu a barvu, např. v modelu RGB nebo CMYK. Každý jednotlivý bod je uspořádán do mřížky (rastru). V obrázku používáme pixely různých odstínů a ty nám zjemňují přechody mezi jednotlivými barvami. Důležitými parametry, které určují kvalitu, je barevná hloubka a rozlišení. Čím více bodů, tím máme větší rozlišení (v DPI). DPI označuje kolik je bodů (pixelů) v jednom palci

(2,54cm). Rozlišení pro zobrazení na monitoru se běžně používá a stačí 72 DPI, pro tisk je zapotřebí větší rozlišení, standartní hodnota je 300 DPI. Zařízení pro převod obrazu na bitmapovou grafiku se nazývá např. kamera, digitální fotoaparát a skener.

Výhoda bitmapové grafiky [20] spočívá v jednoduchosti pořízení např. fotografie. Mezi nevýhody patří velké požadavky na paměť, řádově jednotky megabytů, další nevýhodou je změna velikosti, která zapříčiní zhoršení kvality obrázku. Mezi podporované formáty souborů například patří: BMP, GIF, JPEG, JPEG 2000, PCX, PNG a TIFF. Přechod obrazu na rastrovou grafiku je znázorněn v obrázku č. 7, zobrazení výsledné bitmapové grafiky je pak na pravé straně.[11]



Obr. 7 - Přechod na bitmapovou grafiku [19]

2.4.2 Vektorová grafika

Vektorová grafika [11] není tak rozšířená jako bitmapová, protože programy na práci s touto grafikou nejsou mezi uživateli tak běžné. Nicméně jejich použití je velké. Často se s nimi setkáme např. na webu. Mají jednu velkou výhodu oproti bitmapovým a to, že je lze komprimovat na menší velikost při zachování jejich kvality. Jejich princip spočívá v matematických výpočtech, které umožnují daleko vyšší stupeň komprese. Další výhoda této grafiky spočívá v úspoře místa v paměťovém prostoru.

U vektorové grafiky je obrázek složen ze základních geometrických útvarů, jako jsou třeba body, přímky, křivky a kružnice. U každého útvaru se musí definovat souřadnice počátečního bodu a to vektorem, který nám určí směr a tvar až do konečného bodu. K těmto datům se dále přidá informace o barvě a tloušťce čáry. Složitější vektorový obrázek bývá tvořen seskupením jednodušších obrazců a jejích následným propojením. Z toho plyne další výhoda, při práci se složitějším obrázkem si ho lze rozdělit na původní

obrazce a pracovat s nimi jednotlivě. Nevýhoda u této metody je, že vytvoření obrázku je složitější než u metody bitmapové a práce se složitějšími obrázky vyžaduje vetší nároky na procesor a operační paměť.[20]

Mezi formáty souborů vektorové grafiky patří například: pdf (Portable Document Format), cdr (Corel Draw), zmf (Zoner Callisto). Rozdíl v zobrazení mezi bitmapovou a vektorovou grafikou je znázorněn na obrázku 8.

Obr. 8 - Rastrová a vektorová grafika [20]

3 Programy na kompresi obrázků

Programů na kompresi [12] obrázků je celá řada. Od základních až po programy s rozšířenými funkcemi. U některých programů lze provádět i hromadnou kompresi. Podporované formáty jsou u jednotlivých programů různé. Kromě změny komprimační úrovně lze provádět další úpravy, např. změnu rozlišení, vložení textu, prohlížení a tisk. Některé programy umí zjistit maximální dovolenou kompresi, tak aby byla kvalita oku nerozeznatelná. Lze se setkat s programem, co opraví obrázek po přílišné kompresi, tj. vrátí mu zpátky požadovanou kvalitu. Dále si popíšeme vlastnosti a použití u vybraných programů.

3.1 JPEG Resampler 2010 ver. 6.3.1.0

JPEG Resampler [24] je freeware aplikace, tzn., že je distribuován zdarma nebo za symbolickou cenu, autor si ponechává autorská práva, s tím že nedovoluje program upravovat nebo omezuje použití zdarma. Po nainstalování zabírá na disku cca 4MB volného místa. Program slouží ke změně velikosti u obrázků. Ovládání programu je velmi jednoduché, lze nastavit kvalitu výsledných fotografií a tím způsob výsledného výpočtu rozměrů je např. rozlišení, šířka nebo výška, procento z originálu a další. Samotný program je možný využít jako prohlížeč obrázků, ale na toto je velice pomalý. Vzhled programu je na obrázku v příloze B. JPEG Resampler má např. tyto vlastnosti:

- podporuje např. JPEG, BMP, PNG, JP2, GIF, TIFF a TGA
- je velice stabilní
- má režim pro začátečníky a rozšíření i pro pokročilé
- vložený text lze vycentrovat
- podporuje GPS (GPX) – vložení GPS souřadnic kde byla fotografie vytvořena
- lze ukládat nastavení pro převod
- podporuje funkci drag and drop v prohlížeči Explorer, tj. lze přetáhnout myší obrázek do programu
- podporuje vložení souborů z Exploreru

- možnost výběru více souborů či adresářů
- je vhodný pro publikace upravených fotografií na internetu
- obsahuje nejrůznější filtry, jako např. zaostření, zesvětlení a rozmazaní.
- lze vygenerovat HTML dokument (hlavní jazyk pro vytváření stránek WWW) se skupinou obrázků

3.2 Caesium ver. 1.4.1

Caesium [29] je jednoduchý freeware program, použitelný ke kompresi JPG, BMP a PNG. Program lze stáhnout v zabalené verzi a po rozbalení není třeba instalovat, jde spustit. Celkem zabírá cca 27MB místa na pevném disku. Caesium je vzhledově hezký a propracovaný, ale nicméně nemá ani zdaleka takové možnosti úprav obrázků jako předchozí JPEG Resampler. Vzhled programu je zobrazen na obrázku v příloze B. Caesium má např. tyto vlastnosti:

- před kompresí si lze zobrazit preview výsledného obrázku
- u PNG má výsledný soubor barevnou hloubkou jen max. 24 bit
- změna kvality je možná pouze u souboru JPEG
- nastavení kvality lze u každého souborů zvlášť
- zmenšuje pouze velikost souboru, nemění velikost a rozlišení obrázků

3.3 Image Optimizer ver. 5.10

Image Optimizer [25] je propracovanější shareware program (neregistrovaná verze je plnohodnotná pouze po dobu 30 dní, po té je zapotřebí registrace a úhrada poplatku). Zabírá cca 2,5MB místa na pevném disku. Umí zobrazit obrázky s formátem JPEG, JFIF, BMP, GIF a PNG, následně uloží soubory ve formátech JPEG, GIF, PNG a TIFF. Program obsahuje nástrojovou paletu se základními funkcemi, jako jsou např. v malování. Image Optimizer obsahuje obvyklé funkce, jako např. volitelný stupeň komprese, rastrování, nastavení prokládání, změna rozlišení a rozměrů, zaostřování, zesvětlení a natáčení. Vzhled programu je znázorněn v příloze B. Mezi další vlastnosti např. patří:

- automatická optimalizace palety barev a jejich počet
- vkládání obrázků a textů
- převod více souborů najednou
- nastavení individuálních parametrů pro každý soubor zvlášť
- změny v obrázku ihned před sebou vidíme v reálné velikosti
- zobrazuje aktuální údaj o velikosti a úrovni komprese ve zrovna rozpracovaném obrázku
- má propracovanou kompresi tzv. MagiCompression

MagiCompresion je technika komprese založená na potřebě komprimovat jen některé části obrázků, ne všechny části obrázků jsou stejně důležité, proto lze použít na jednotlivé části jinou úroveň komprimace, tak aby to bylo co nejvíce efektivní. Tento algoritmus pracuje automaticky, nastavuje se pouze jeho stupeň. Jednodušší varianta MagiCompression je tzv. Extra Compression, je rychlejší, ale dává většinou horší výsledky.

3.4 Image Converter Plus ver. 8

Image Converter [26] Plus je vysoce efektivní a velice propracovaný shareware program. Po nainstalování zabírá na pevném disku cca 79MB. Dokáže pracovat s více než 100 formáty. Pracuje s barevnou hloubkou až 32 bitů. Tento program dokáže některé základní i rozšířené funkce co byly u předchozích programů. Od různých úprav jako např. změna velikosti, rozlišení, zaostření, vkládání obrázků a textu, možnost zrcadlení a změnu barevné hloubky, až po hromadný převod souborů. Vzhled program je zobrazen v příloze B. Další zajímavé vlastnosti si uvedeme níže.

- podporuje tisk a skenování obrázků (dokumentů)
- lze nahrát soubory na FTP přímo z programu (FTP je protokol pro přenos souborů mezi počítači)
- převádí velké soubory
- nemá omezení najednou zpracovávaných souborů
- podporuje více jádrové procesory, převody jsou rychlejší
- podporuje 64 bitové systémy
- zobrazuje obsáhlé informace u vybraného obrázku
- obsahuje integrace do průzkumníka Windows

- soubory lze rovnou z programu odeslat na email

3.5 RIOT ver. 0.4.6

Program RIOT [27] je jednoduchý freeware program. Po nainstalování zabírá cca 1,3MB volného místa na pevném disku. Program pracuje s formáty souborů JPEG, GIF a PNG. RIOT není tak propracovaný jako některé předešlé programy. Kromě možnosti změny velikosti obrázků, obsahuje také základní nástroje, jako např. rotace, zoom, převracení, kontrast a průhlednost. Je třeba si dát pozor při ukládání upraveného obrázku, aby se nám nepřepsal původní obrázek, na toto chybí v programu nějaká ochrana. Vzhled programu je znázorněn v příloze B. Mezi další vlastnosti patří např.:

- je rychlý a jednoduchý na používání
- snadná úprava obrázků, co se týče nastavení komprese např. počtu barev
- dokáže přidat nebo odebrat chtěná či nechtěná data v původním obrázku, jedná se o tzv. metadata
- zobrazuje najednou dvě okna, tj. originální obrázek a upravený obrázek
- změny ihned vidíme v reálném čase bez nutnosti uložení
- program obsahuje tlačítko In-place Compare, které slouží k přehození originálního obrázku za upravovaný obrázek, tím lze s naprostou jistotou určit, zda upravovaný obrázek visuálně odpovídá originálu a neliší se od něj.

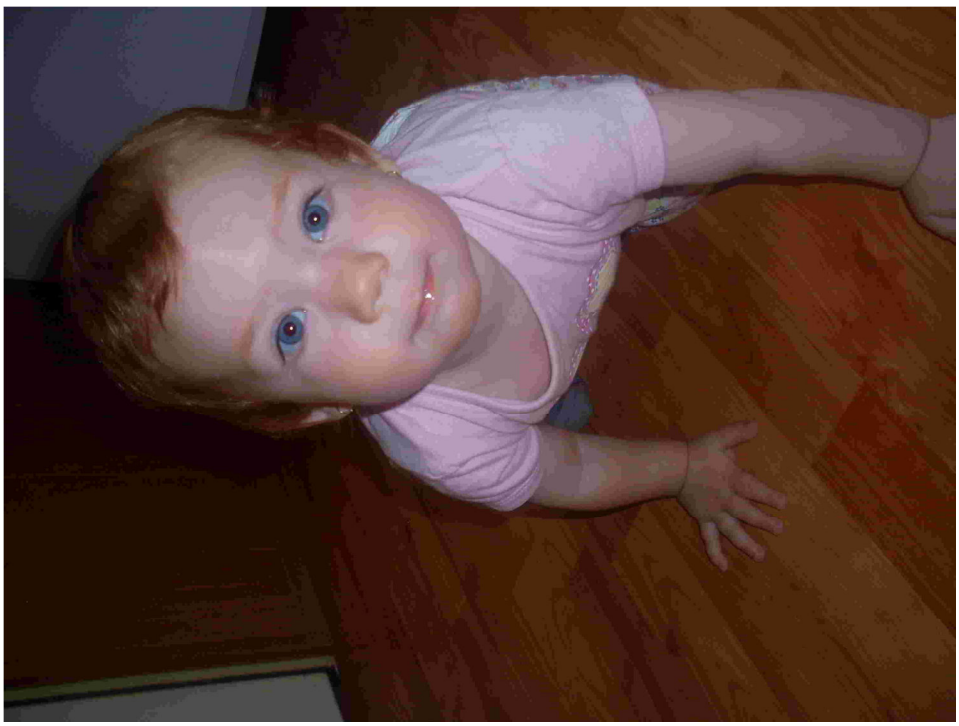
3.6 JPEG Enhancer ver. 1.4

JPEG Enhancer [28] je velmi zajímavý shareware program. Po nainstalování má program cca 2,8 MB. Vůbec neslouží ke kompresi jako předešlé programy, ale slouží k opravě obrázků po přílišné kompresi. Čím větší kompresi použijeme, tím je větší pravděpodobnost vzniků artefaktů. JPEG Enhancer není zázračný, ale celkem dost slušně dokáže vrátit zpátky částečný vzhled vzniklý nepřiměřenou kompresí. Např. při i 90% kompresi, je obnova obrázku znatelná. JPEG Enhancer využívá k obnově obrázků vlastní

technologii. Nejedná se jen pouze obyčejné rozostření, ale o složitější dopočítávání původního stavu obrázku. Mezi další vlastnosti patří např.:

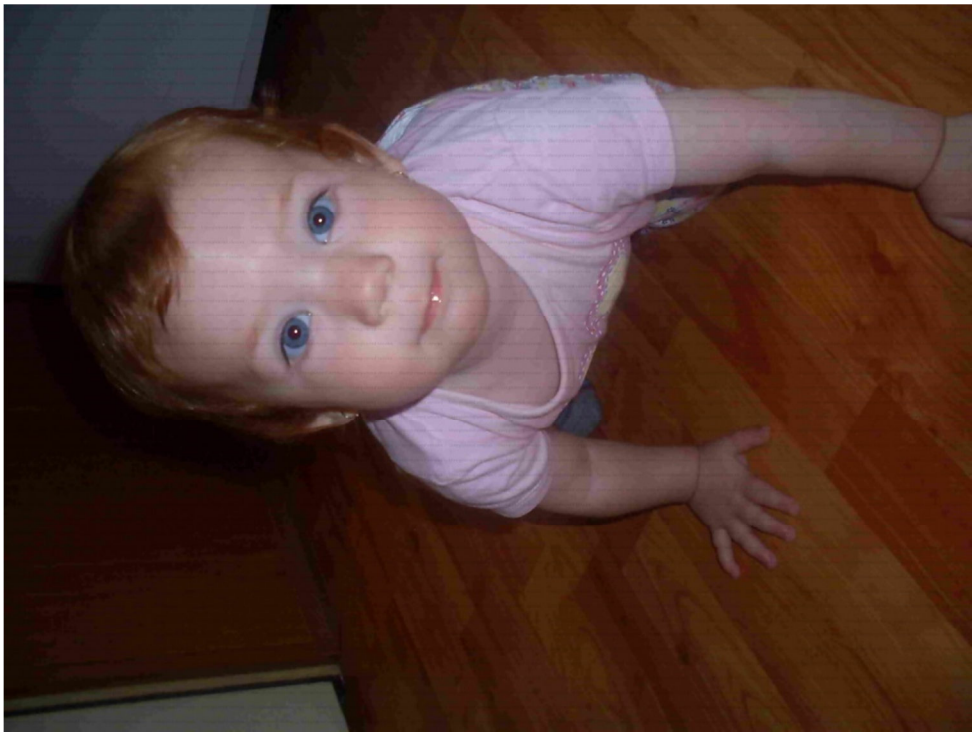
- nepodporuje Windows 7/8, nicméně program i tak běží na těchto systémech
- výběr z několika vzhledů programu
- vysoká míra úspěšnosti
- podporuje čtení pouze formátu JPEG
- ukládat lze ve formátu JPEG, BMP a PNG
- možnost volby velikosti úrovně opravy

Jako příklad si uvedeme obrázek s rozlišením 3072x2304, bitová hloubka 24 bit, pořízen je na fotoaparátu Samsung S760. Velikost souboru je 3131 KB. Úprava je provedena v programu RIOT. Použitá komprese je 90%, náhled je na obrázku č. 9. Velikost nového souboru je pouhých 56 KB.



Obr. 9 – Holčička po přílišné kompresi

Následně ho v JPEG Enhancer obnovíme. Zvolíme největší úroveň obnovy. Výsledný obrázek je dokonce větší než původní obrázek před kompresí, má 3498 KB. Náhled je znázorněn na obrázku č. 10.

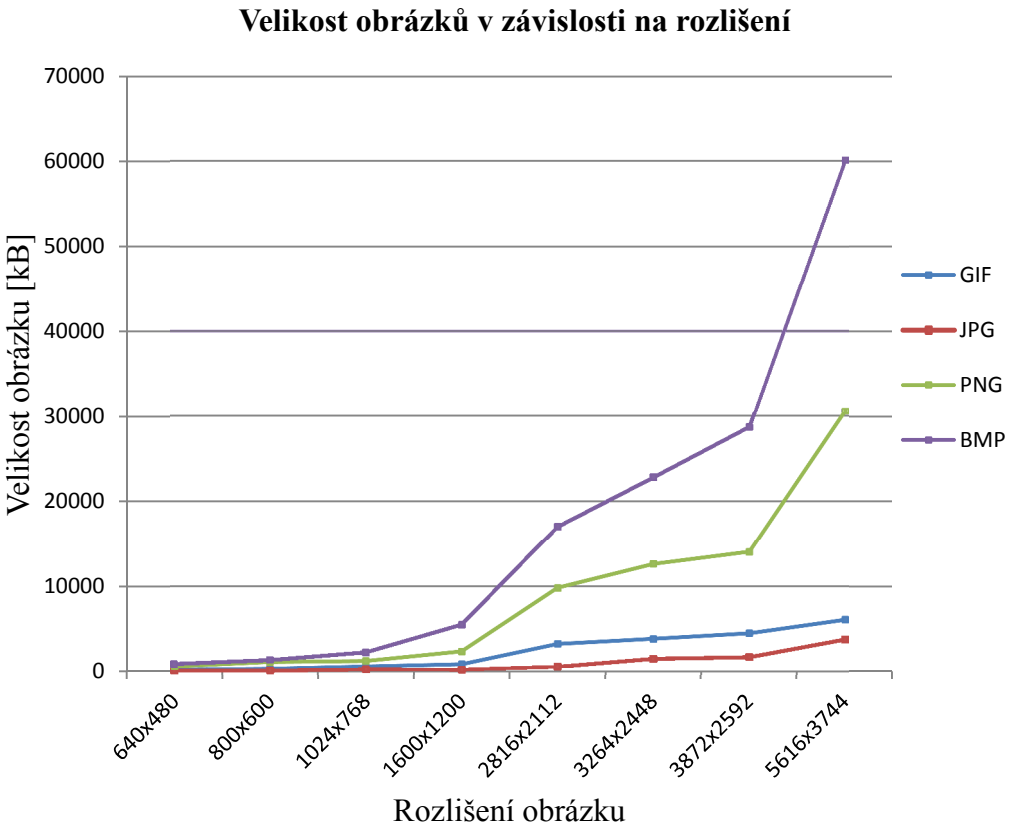


Obr. 10 – Holčička – obnovený obrázek

Rozdíl je znatelný na první pohled. Jen do obnoveného obrázku je vložen vodoznak, protože program je pouze zkušební verze.

3.7 Srovnání velikosti obrázku podle typu souboru

Budeme porovnávat celkem 4 nejpoužívanější typy souborů, JPEG, BMP, PNG a GIF s různou velikostí a rozlišením. Obrázky, jejich vlastnosti a parametry, jsou uvedeny v příloze C. Jednotlivé velikosti souborů, v závislosti na rozlišení jsou zaneseny v grafu na obrázku č. 11.



Obr. 11 – Graf velikosti obrázku v závislosti na rozlišení

Jak je vidět z grafu, velikost souboru s rozlišením úměrně vzrůstá. Samozřejmě musí se vzít v úvahu i bitová hloubka obrázku. V porovnávaných obrázcích je u stejného typu souboru různá bitová hloubka, zvláště u PNG, nicméně je to především rozdíl mezi 24 a 32 bitovou hloubkou, z tohoto důvodu není patrný větší rozdíl v přechodu k vyššímu rozlišení. Kdybychom použili např. bitovou hloubku 8 a 32 bitů, rozdíl ve velikosti obrázku by byl více patrný.

Největší velikost má jednoznačně soubor typu BMP. U porovnávaných obrázků to bylo více než 60 MB. Při dalších úpravách, např. změna rozlišení, při zachování stejné kvality, není problém, aby výsledný obrázek překročil hranici 100 MB, což je ale na běžný obrázek opravdu příliš. Proto je z hlediska úspory místa vhodný soubor typu PNG. Ten i s 32 bitovou hloubkou má menší velikost jak předchozí BMP. Nicméně, bez komprese jsou tyto obrázky ve větším rozlišení také celkem velikostně náročné. Z tohoto důvodu je zde uveden další formát souboru JPEG. Ten má jednoznačně nejlepší vlastnosti co se týče poměru kvality a velikosti. Mezi porovnávanými

obrázky byla velikost, u největšího rozlišení, cca 3,8 MB, což je skoro až neuvěřitelné. JPEG je jednoznačně nejlepší volba pro úsporu místa. Čtvrtým a zároveň posledním porovnávaným formátem souborů je GIF. Jeho poměr mezi rozlišením a velikostí je také velice dobrý. U největšího rozlišení velikost činní cca 6 MB. Nicméně i přesto je zcela nevhodný pro zobrazování např. fotografií, protože jeho bitová hloubka 8 bitů je absolutně nedostačující. Z tohoto důvodu je vhodné používat tento typ souboru pouze na obrázky, co nemají více jak 256 barev. Největší potenciál tohoto formátu je v podpoře animací.

3.8 Komprese v programu RIOT

Vybereme si na srovnání tři obrázky s koncovkami JPEG, GIF a PNG. Obrázky zkomprimujeme tak, aby nebyl rozdíl ve vzhledu oproti originálu. Velikosti jednotlivých souborů porovnáme v tabulce č. 3 a zaneseme do grafu.

Název	Typ souboru	Rozlišení	Bitová hloubka [bit]	Velikost [kB]
Babička	GIF	5616x3744	8	6080
Formule	JPEG	5616x3744	24	3760
Letovisko	PNG	5616x3744	24	30600

Tab. 3 - Vybrané obrázky před kompresí

Po úpravě obrázků dostaneme soubory o velikosti viz. tabulka č. 4.

Název	Typ souboru	Rozlišení	Bitová hloubka [bit]	Velikost [kB]
Babička	GIF	5616x3744	8	5180
Formule	JPEG	5616x3744	24	452
Letovisko	PNG	5616x3744	24	8010

Tab. 4 - Vybrané obrázky po kompresi

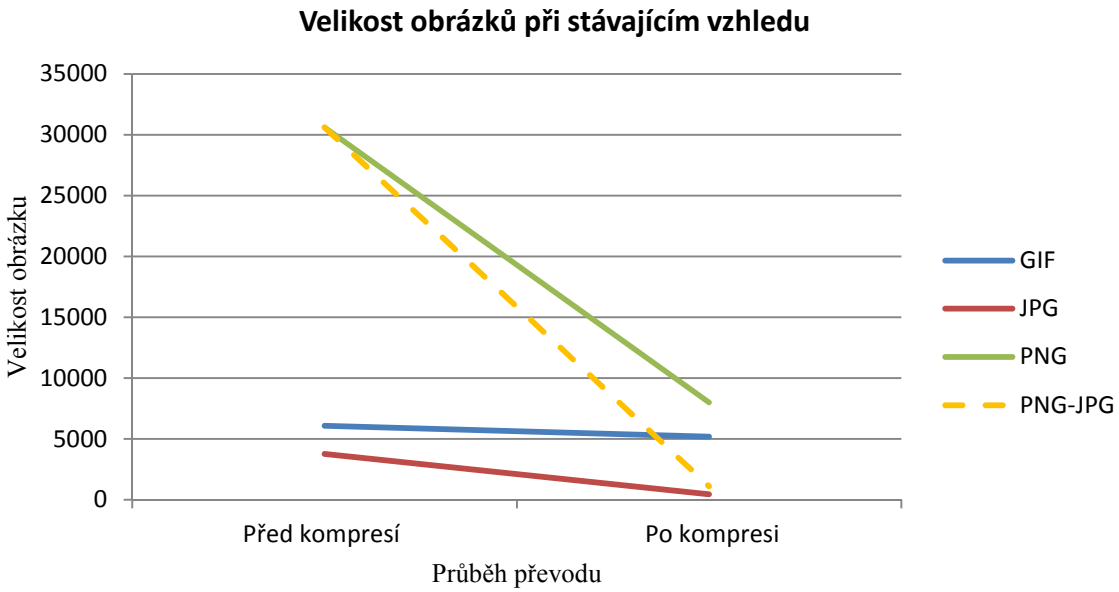
V obrázku Formule, se podle programu RIOT provedla 73% komprese, tak aby nebyl vůbec patrný rozdíl oproti původnímu obrázku. Komprese by mohla být i větší, protože na samotném objektu v obrázku, nebylo viditelné zhoršení kvality, ale na pozadí se už vytvořily nechtěné artefakty. Velikost upraveného obrázku je 452 kB.

Úprava obrázku Babička není tak efektivní jako předchozí. U tohoto souboru se provedla redukce z 256 barev na zobrazení pouhých 83 barev, aby zůstal nerozeznatelný od originálu. Na první pohled se jeví redukce o 173 barev hodně, ale co se týče samotné velikosti obrázku, tak není takový razantní rozdíl oproti originálu. Velikost obrázku se změnila o pouhých 900 kB tzn., že celková velikost upraveného obrázku je 5180 kB.

Dalším obrázkem je Letovisko. Originální soubor je velký, přes 30 MB. Redukce u PNG v programu RIOT spočívá v redukci barev. Tudíž stávajících 24bitů nahradíme 256 barvami. Jinou možnost nám tento program nenabízí. Po převodu máme obrázek, který se mírně liší od originálu, ale jen v případě, že si obrázek přiblížíme a zaměříme se na tyto odlišnosti. V rámci tohoto testu považujeme obrázek za shodný. Výsledný obrázek je značně zmenšen, má něco málo přes 8 MB.

Obrázek formátu GIF se celkově zmenšil o 15%, PNG byl zmenšen o 74% a poslední JPEG se zmenšil o 88%. Z veškerých zjištěných údajů jednoznačně vyplívá, že nejlepší formát na práci s obrázky je JPEG. Grafické znázornění jednotlivých rozdílů celkové komprese je vidět v grafu na obrázku č. 12.

Zajímavostí je, že když obrázek Letovisko převedeme nejprve na formát souboru JPEG a po té tento nový soubor zkomprimujeme 85% kompresí, tak aby nebyl rozdíl oproti originálu, pak výsledný soubor má velikost pouhých 1120 kB. V obrázku č. 12 je tento převod znázorněn žlutou přerušovanou čarou.



Obr. 12 – Graf velikosti obrázků při stávajícím vzhledu

3.9 Srovnání souboru typu BMP s ostatními formáty

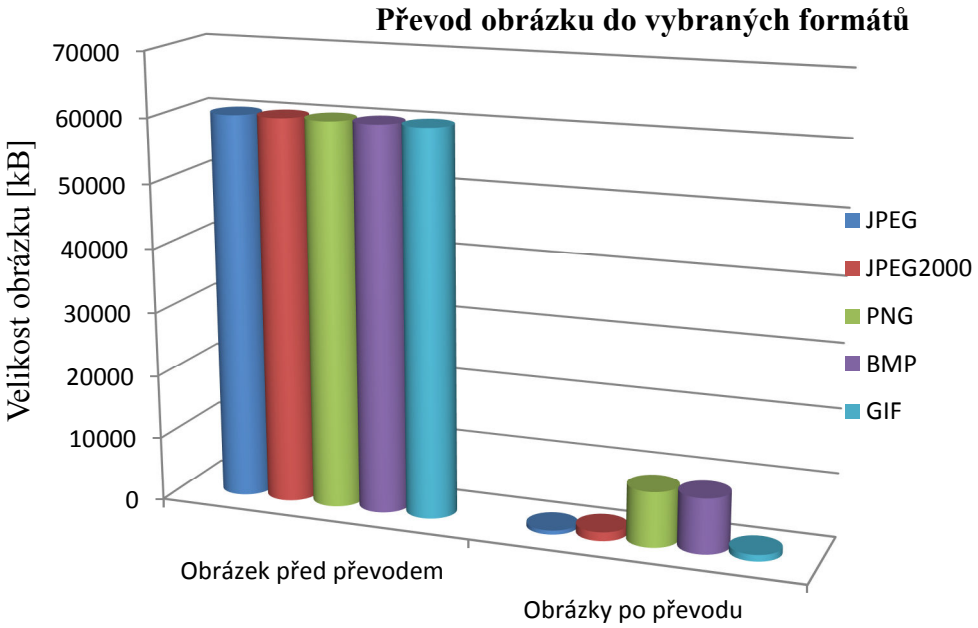
Na toto porovnání je vybrán BMP obrázek Krajina s rozlišením 5616x3744 a velikosti 60,1 MB. V programu JPEG Resampler zmenšíme rozlišení na 3 Mpix tj. rozlišení 2121x1414. Obrázky převedeme do formátu JPEG, JPEG2000, PNG, BMP a GIF. Výsledné velikosti obrázků jsou zapsány v tabulce č. 5.

Typ souboru	Velikost souboru [kB]
JPEG	624
JPEG2000	1370
PNG	8580
BMP	8580
GIF	1010

Tab. 5 – Porovnání komprese z BMP do různých formátů

Jak vidíme v tabulce, JPEG je opět nejlepší, co se týče velikosti výsledného souboru. Všechny vytvořené obrázky mají úplně srovnatelnou kvalitu jako původní obrázek, až na obrázek formátu GIF, 256 barev je opravdu málo. Při stávající kvalitě, je zmenšení rozlišení, co do velikosti souboru, stejné u formátu PNG tak i u BMP. V případě zmenšení kvality

ve formátu PNG je pak rozdíl velikosti oproti BMP znatelný. Grafické znázornění jednotlivých převodů obrázků je vidět na obrázku č. 13.



Obr. 13 – Graf převodu obrázku BMP do vybraných formátů

4 Příklady komprese využívané v SW pro zpracování obrazu

V kapitole 2 jsme se seznámili s obrazovým formátem JPEG. V této kapitole budeme hovořit o jedné z nejpoužívanějších kompresních metod a to JPEG komprese. Ve standardu JPEG může kodér i dekodér být charakterizován jedním ze čtyř režimů činnosti viz. tabulka č. 6. [10]

Režim činnosti	Kódování	Vlastnosti
Progresivní	ztrátové	velmi náročné na paměť, využívá se především pro přenos obrázků po síti
Sekvenční	ztrátové	nejvíce využívaný režim, nejmenší náročnost na paměť
Hierarchický	ztrátové	rychlé náhledy, podpora tisku a zobrazení, mnoho rozlišení v jednom snímku
Bezeztrátový	bezeztrátové	tento způsob není moc známý ani podporovaný, kompresní poměr 1:2

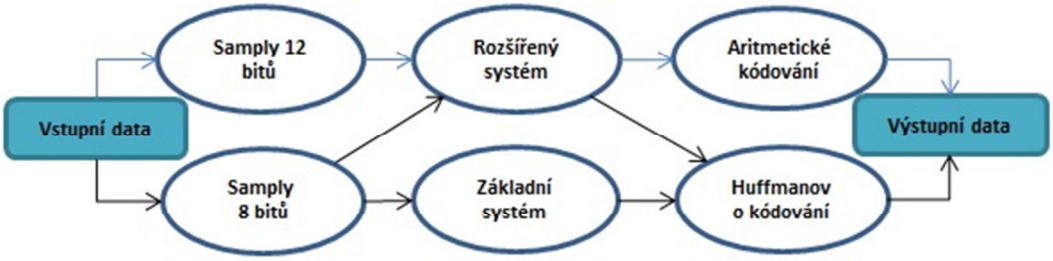
Tab. 6 – Režim činnosti kodéru a dekodéru

Dále bude popsán režim, který zpracovává obrazová data sekvenčně a to po blocích 8x8 a 16x16 pixelů. Mezi největší výhodu tohoto režimu patří minimální využití operační paměti. Díky malé náročnosti na paměť je například umožněno digitálním fotoaparátům vytvářet fotografie s mnoha milióny pixelů i za předpokladu, že fotoaparát má na operačních čipech paměť několikanásobně menší.

Další výhodou sekvenčního režimu je schopnost rychle dekódovat obrázek z formátu JPEG do nekomprimovaného rastru. Zpracovávaný obrázek lze například při otevírání z pomalého přenosového média anebo při přenosu po síti, zobrazovat v prohlížeči postupně. Sekvenční režim je vhodný i pro některé tiskárny. Lze spustit tisk, aniž bychom museli načíst celý soubor dopředu.

U sekvenčního režimu je celkem pět způsobů jakými lze zpracovávat data. Jednotlivé zpracování se mezi sebou liší např. bitovou hloubkou vzorků na vstupu, způsobem výpočtu DCT (diskrétní kosinové transformace) a

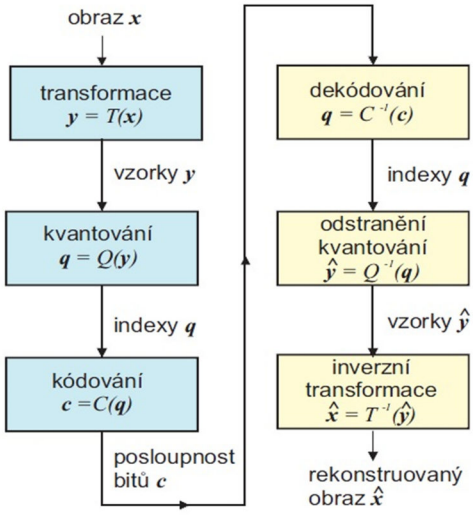
kódováním vzorků. Barevné vzorky na vstupu mohou být osmibitové nebo dvanáctibitové. DCT lze vypočítat standartním nebo rozšířeným algoritmem. Výstupní soubor lze zpracovávat aritmetickým nebo Huffmanovým kódováním. Posloupnost činností v pěti různých způsobech zpracování dat je znázorněn na obrázku č. 14. [10]



Obr. 14 – Pět způsobů zpracování dat [10]

Níže bude popsán nejpoužívanější způsob komprese a to JPEG komprese s osmibitovým vstupním souborem, výpočtem DCT, kvantováním pomocí standardního algoritmu a výstupní soubor bude zakódován Huffmanovým kódem.

4.1 Komprese obrazu a jeho rekonstrukce



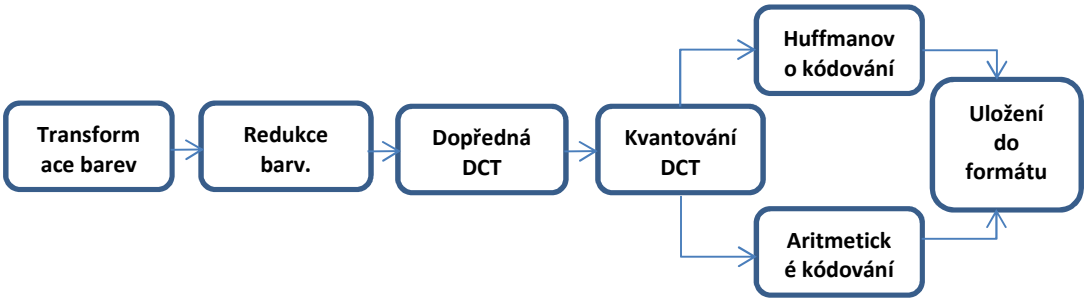
Obr. 15 – Komprese obrazu a jeho rekonstrukce [22]

Transformace T odstraňuje redundanci, jedná se např. o kosinovou transformaci nebo kódování RLE. Kvantování Q odstraňuje irelevanci a není

invertovatelné, jedná se o např. zanedbání koeficientů kosinové transformace odpovídající vysokým frekvencím. Kódování a dekódování je bezeztrátové [22].

4.2 JPEG komprese

Vstupní obrazová data při převodu prochází různými, po sobě jdoucími operacemi, od transformace barev až po uložení obrázku do formátu JFIF (JPEG). Celý postup převodu souboru je znázorněn na obrázku č. 16.



Obr. 16 – Postup převodu souboru [10]

Bloky DCT (8x8) jsou vypočítány pomocí tzv. kvantizačních tabulek. Výsledkem tohoto výpočtu jsou hodnoty, které jsou ve velké míře nulové, tím dochází k největší ztrátě informace a tudíž k největšímu zmenšení objemu dat.

Kvantované DCT koeficienty jsou dále kódovány buď aritmetickým anebo Huffmanovým kódováním. Aritmetické kódování je účinnější, ale také o dost náročnější na výpočetní výkon, a z tohoto důvodu, se spíše využívá Huffmanovo kódování.

Jako poslední operace je uložení zpracovaných dat do souboru typu JFIF (JPEG).

4.2.1 DCT (Diskrétní kosinová transformace)

Princip této transformace spočívá v převedení matice signálových vzorků funkce g(x,y), na matici frekvenčních koeficientů funkce G (fx,fy) [23]. Na první pohled se zdá výhodnější transformovat najednou celý obraz, jenže to by mělo velké nároky na výpočetní techniku, z toho důvodu obraz rozkládá na bloky o 8x8 pixelů. Použitá transformace se volí mezi cenou a užitkem. Z těchto důvodů se nakonec vybrala Diskrétní kosinová

transformace. „*Tato transformace vznikne rozkladem periodické časové funkce v nekonečnou řadu harmonických kmitů vyjádřenou v obecném případě kromě stejnosměrné složky jednotlivými sinusovkami a kosinusovkami. Přímou transformací časové funkce g(t), definujeme Fourierův obraz G(f), který představuje původní funkci ve frekvenční oblasti. G(f) udává frekvenční závislost amplitud harmonických kmitů, ze kterých se funkce g(t) skládá*“.[23]

Diskrétní Fourierova transformace má tvar:

$$F(u,v) = \left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) * e^{j\frac{2\pi}{N}(ux+vy)} \right]$$

$$f(x,y) = \left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} F(u,v) * e^{j\frac{2\pi}{N}(ux+vy)} \right]$$

„Jádro DFT lze rozložit na reálnou a imaginární část – cos + j sin. Pokud je vstupní funkce g(x,y) sudá, pak automaticky přejde DFT na diskrétní kosinovou transformaci DCT (zbavíme se sinových složek). Pokud není, musíme ji na sudou funkci převést.“[23]

Diskrétní kosinová transformace má tvar:

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

Zpětná inverzní transformace má tvar:

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

$$C(u,v) = \frac{1}{\sqrt{2}} \qquad u,v = 0$$

Výpočet je velmi složitý, dá se říct, že to je nejnáročnější krok v celé JPEG kompresi. Diskrétní kosinová transformace je neztrátová vyjma zaokrouhlovacích chyb.

Hlavní výhoda diskrétní kosinové transformace spočívá vtom, že má již zmiňovaný stejnosměrný člen. V tomto členu je obsažena střední energie

signálu. Zbylé střídavé složky odpovídají různým frekvencím v závislosti na pozici vůči stejnosměrnému členu [23], viz. kapitola 4.1.4.

4.2.2 Transformace barev do prostoru YC_BC_R

V první řadě bude provedena transformace barev např. z RGB a CMYK do prostoru YC_BC_R. Při této transformaci nedochází k vůbec žádné ztrátě informace. V převedeném prostoru YC_BC_R nese složka Y informaci o světlosti pixelu a zbylé složky C_BC_R nesou informaci o barvě. Vzorec pro přepočítání barev z prostoru RGB do prostoru YC_BC_R je zobrazen na obrázku č. 17.

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} +0{,}299 & +0{,}587 & +0{,}114 \\ -0{,}16875 & -0{,}33126 & +0{,}5 \\ +0{,}5 & -0{,}41869 & -0{,}08131 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Obr. 17 – Vzorec pro přepočet barvového prostoru [17]

Zpětné přepočítání barvových složek YC_BC_R do prostoru RGB se provádí například v samotném prohlížeči obrázků.

4.2.3 Redukce barvonosných složek

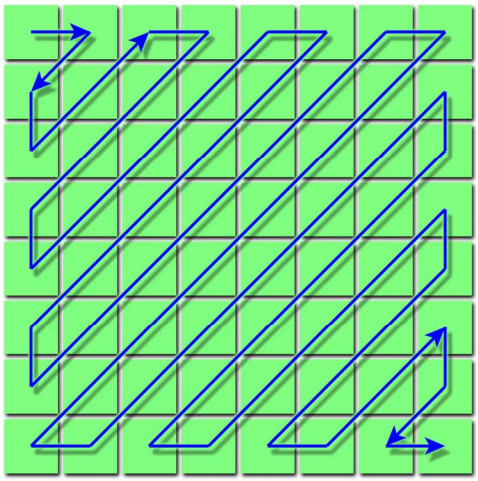
Při druhém kroku může docházet k podvzorkování barvonosných složek a tím lze docílit snížení velikosti zpracovávaných dat. Podvzorkování využívá vlastností lidského oka, které má mnoho senzorů snímajících jak informace o světlosti, tak informace o barvách. Jelikož lidské oko má více senzorů snímajících informace o světlosti, může s větší přesností rozpoznat informace o světlosti než informace o barevných odstínech.

Z předchozího textu tedy vyplívá, že složka Y co nese informaci o světlosti, není podvzorkována a zbylé barvonosné složky C_BC_R mohou být podvzorkovány. Jejich hodnota vznikne buď, z průměrování dvou sousedních pixelů na řádku, anebo čtyř pixelů tvořící čtverec.

V případě zprůměrování dvou sousedních pixelů dochází ke zmenšení dat o 66%, tudíž převod z 6 bytů na 4 byty. Když budeme uvažovat o případu sloučení čtyř pixelů, celková úspora bude větší a to 50%, tzn., že z původních 12 bytů dosáhneme pouze 6 bytů. Tato redukce barvonosných složek je tedy ztrátová, protože už nemáme samostatné informace o jednotlivých pixelech. Toto podvzorkování využívá například digitální televize. [10]

4.2.4 Dopředná DCT

Dopředná DCT nám zpracovává zvlášť složku Y, a zvlášť barvonosné složky C_B a C_R. Jak bylo výše uvedeno, princip dopředné DCT spočívá v rozdělení obrázku na pravidelné bloky 8x8 hodnot a tyto bloky se z časové roviny převedou na jiný blok stejné velikosti do frekvenční roviny. V této rovině definujeme stejnosměrnou (DC) a střídavou (složku). Průměrná hodnota celého bloku 8x8 představuje stejnosměrnou složku, tj. první vypočtený koeficient. Ten má největší vliv na výsledný obrázek. V následujících koeficientech jsou uloženy amplitudy střídavých kosinových složek různých frekvencí. Je žádané, aby dané frekvence, byly seřazeny v rostoucím pořadí, protože největší frekvence mají nejmenší vliv na daný obrázek. Tohoto se docílí tzv. Zig-zag uspořádáním. Při této úpravě se neztrácí žádná informace. Princip uspořádání Zig-zag je znázorněno na obrázku č. 18. [10]



Obr. 18 – Princip uspořádání Zig-zag [30]

V levém horním rohu se nachází DC člen. Ostatní členy jsou AC. Ty s přibývající vzdáleností ztrácejí vliv na výsledný obrázek. Z toho vyplívá, že prvek v dolním pravém rohu má nejmenší vliv na výsledný obraz.[10]

4.2.5 Kvantování DCT koeficientů

V této chvíli máme rozděleny bloky o velikosti 8x8 hodnot. Každý jednotlivý blok je vydělen hodnotami uloženými v tzv. kvantizačních tabulkách. Tyto tabulky jsou vytvořeny, aby obsahovaly nízké hodnoty u DC

členů a také u AC členů nízkých frekvencí. U AC členů vysokých frekvencí se používají vysoké hodnoty. Kvantizační tabulka je stejná pro celý obrázek. V případě barevného obrázku mohou být tyto tabulky dvě. Výsledkem dělení je nový blok téže velikosti (8x8), který má u zmiňovaných AC členů s vysokými frekvenci nulové hodnoty. Výsledky se zaokrouhlují na nejbližší celé číslo, proto zde dochází k největší ztrátě informací. Příklad kvantizační tabulky je znázorněn v příloze D.[8]

V tabulce se vybírají hodnoty pomocí Zig-zag. Čím větší koeficient v tabulce, tím se ztratí větší množství informace. Ve vytváření nulových hodnot spočívá celý význam DCT a kvantování. Nulové hodnoty se dobře kódují. Nejčastěji se k tomu využívá aritmetické a Huffmanovo kódování. [10]

4.2.6 Aritmetické a Huffmanovo kódování DCT koeficientů

Huffmanovo kódování bylo popsáno v kapitole 1.5.4. Aritmetické kódování pracuje s proměnlivou délkou kódového slova. Je založeno na principu přiřazení krátkého binárního kódu znakům, které se vyskytují s větší pravděpodobností, a naopak dlouhého kódu, znakům vyskytujících se s menší pravděpodobností. Aritmetické kódování je oproti Huffmanovu výhodnější ve velikosti výsledného souboru, zhruba o 10%, ale celkové kódování je složitější a tudíž náročnější na hardware a čas. Z tohoto důvodu je více používáno Huffmanovo kódování.[10]

Jako posledním krokem je uložení zpracovaných dat do souboru JFIF (JPEG, JPG, atd.).

5 Vlastní program na kompresi obrazu ve vybraném formátu

Před vytvořením programu je zapotřebí si stanovit základní požadavky jak má tento program vypadat a co má umět. V prvním kroku si stanovíme formát vstupního souboru. Vstupní soubor bude ve formátu JPEG, z důvodu velké rozšířenosti a použitelnosti. Výstupní soubor bude taktéž ve formátu JPEG a to s různými stupni komprese. V dalším kroku se musí vybrat programovací prostředí, ve kterém bude kompresní program vytvořen. V našem případě se jedná o programovací prostředí Microsoft Visual Studio 2010 jazyk C#.

5.1 Provedení programu

Náš program má celkem 9 tlačítek a 2 textová pole. Vzhled programu je zobrazen na obrázku č. 19. První textové pole spolu s tlačítkem procházet slouží k vyhledání a načtení vstupního souboru. Dále máme celkem 3 možnosti, jak budeme v kompresi pokračovat.

Jako první možnost si lze zvolit přednastavenou úroveň komprese, jedním z 6 tlačítek Komprese 10 – 99%. Následně po té, se do složky programu uloží nové soubory s požadovanou kompresí a to ve tvaru např. Komprese30%.jpg.

Jednou z dalších možností komprese je tlačítkem Spusť kompletní kompresi. Toto tlačítko spustí kompresi ve veškerých přednastavených úrovních, a to konkrétně o velikosti 10%, 30%, 50%, 70%, 90% a 99%. Uloží se taktéž do složky programu s koncovkou JPG.

Jako poslední možnost jak pokračovat v kompresi je zvolení libovolné úrovně komprese a to nastavením v druhém textovém poli s názvem Nastav úroveň manuální komprese [%]. Po definování kompresní úrovně, následuje stisknutí tlačítka, Spusť manuální kompresi. Toto tlačítko opět jako dvě předešlé možnosti uloží nový soubor do složky programu a to taktéž ve tvaru KompreseXX.jpg.

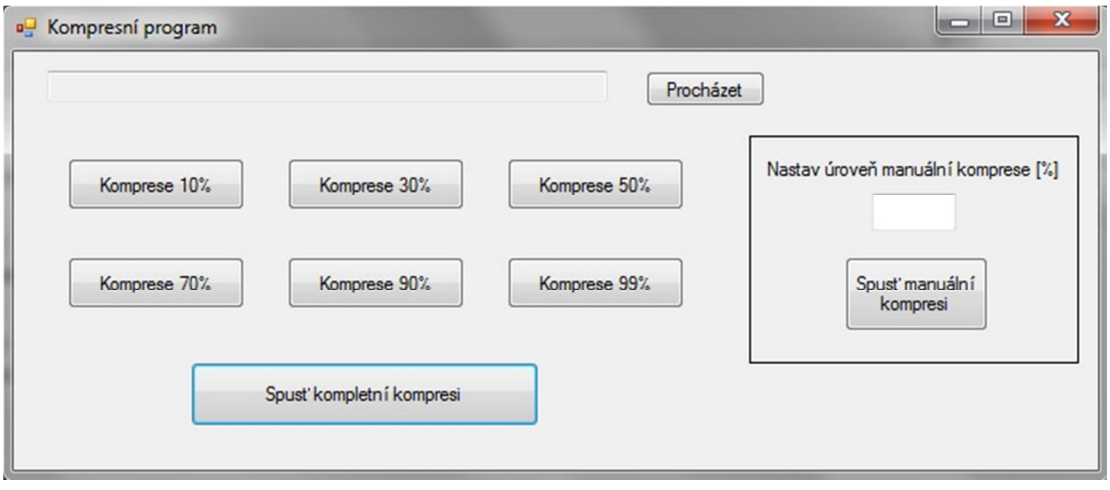
5.2 Popis programu

Po spuštění programu jsou všechna tlačítka až na jedno neaktivní. Jediná možná volba je tlačítko Procházet. To je z důvodu, aby nebylo možné spustit kompresi v případě, když by nebyl načten žádný obrázek. Po stisknutí tlačítka Procházet se objeví okno průzkumníka sloužící k výběru souboru, který chceme upravit. Jako typ souboru lze vybrat pouze JPEG/JPG. Po vybrání souboru, se v okně zobrazí název souboru, spolu i s místem kde je uložen. Dále se nám nabízí buď možnost přednastavitelné komprese anebo volba manuální komprese.

V případě volby přednastavitelné komprese, není nutné zadávat další parametry, komprese se provede dle velikosti a výsledné soubory se uloží do složky s programem.

V případě volby manuální komprese je zapotřebí zadat do příslušného okna úroveň komprese, tj. číslo 1 až 99, tzn., když zadáme číslo např. 94, program udělá kompresi 94%. Z toho tedy vyplívá, že lze zadat pouze dvouciferné číslo. V případě že okno s úrovní komprese nevyplníme a jen spustíme tlačítkem manuální kompresi, program nám vytvoří soubor s 50% kompresí. Opět tyto výsledné soubory se uloží do složky s programem.

Program se standardně vypíná křížkem vpravo nahoře. Další popis programu je znázorněn ve vývojovém diagramu v příloze F.



Obr. 19 – Kompresní program

5.3 Srovnání a využití programu

Tento program je primárně určen pouze ke zmenšení velikosti obrázku a fotografií. Zmenšení velikosti se rozumí změna velikosti dat. Jak vyplívá z předešlého textu, program dokáže provést kompresi 1-99% a to u souborů typu JPEG. Na začátku, před vytvořením tohoto programu, byl hlavní cíl takový, že čtenáři předvedeme jak je snadné a časově nenáročné si obrázky nebo fotografie zmenšit. Myslím si, že to bylo víc než dobře splněno. Důvod, proč se program zaměřuje pouze na kompresi je čistě praktický, a to takový, jako je např. úspora místa v počítačích nebo posílání obrázků či fotografií přes email. Z toho důvodu se tento program liší od jiných volně dostupných programů, které jsou ve velké míře zaměřeny na hlubší úpravu obrázků, tj. např. zesvětlení, zaostření atd. Pro uvedený hlavní cíl je náš program naprosto dostačující.

5.4 Ověření funkce programu

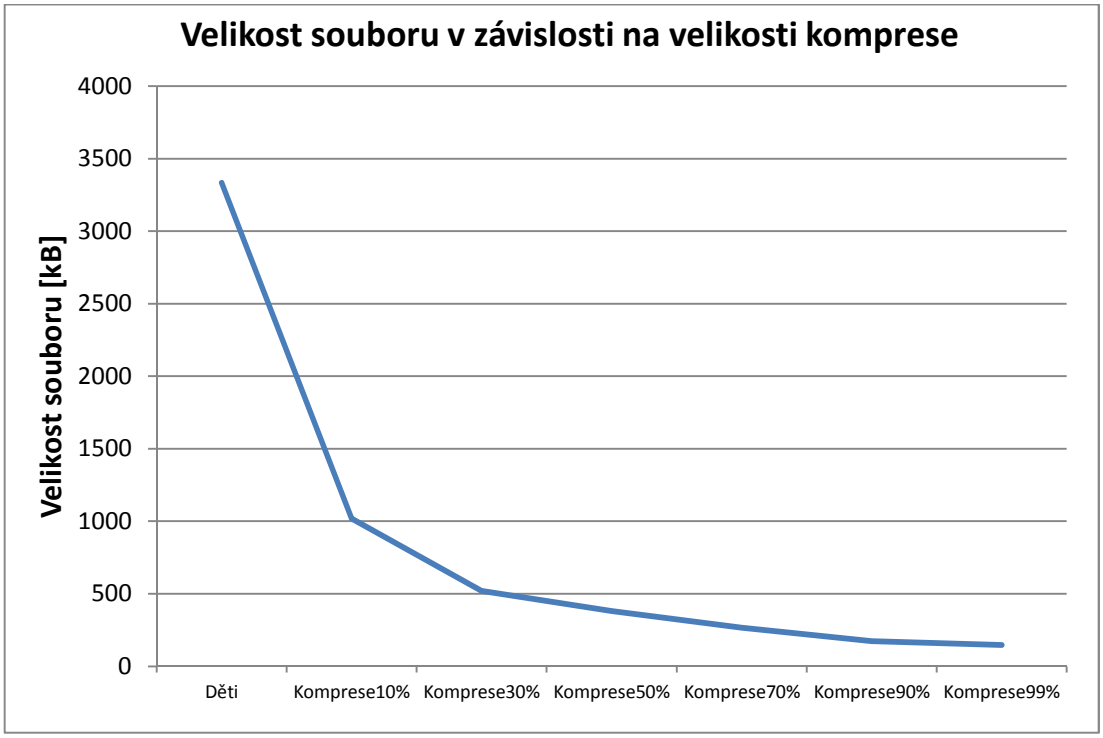
Pro základní test a ukázku funkčnosti jsem zvolil fotografii dětí. Jak vyplívá z předešlého textu, soubor je typu JPG. Originální fotografie je zobrazena na obrázku č. 20. Fotografie byla tedy pořízená na fotoaparátu Samsung S760 s rozlišením 3072 x 2304 a její velikost je 3498 kB.

Obr. 20 - Děti

Po načtení této fotografie do programu si zvolíme možnost, Spusť kompletní kompresi. Jednotlivé převedené obrázky porovnáme a následně zaneseme do tabulky č. 7. Pro lepší znázornění jsou data ještě zanesena do grafu v obrázku č. 21.

Název souboru	Kompresní poměr	Velikost
Děti	1	3498 kB
Komprese10%	3,27	1019 kB
Komprese30%	6,41	520 kB
Komprese50%	8,78	380 kB
Komprese70%	12,54	266 kB
Komprese90%	19,28	173 kB
Komprese99%	22,69	147 kB

Tab. 7 – Komprese ve vlastním programu



Obr. 21 – Graf velikosti souboru v závislosti na velikosti komprese

Podle očekávání, při větším stupni komprese se nám zmenšuje velikost souboru. U 70% komprese ještě není lidským okem možné rozeznat zhoršení kvality obrázku. Přičemž výsledná velikost převedeného obrázku je 266kb, tudíž velikost souboru byla zmenšena přibližně 12,5x oproti originálu.

V příloze E je pro názornou ukázku umístěn náhled testovacího obrázku s kompresí 50%, 70%, 90% a 99%.

K této bakalářské práci je přiloženo CD se samotným programem včetně zdrojových kódů. Dále je přiložena použitá fotografie, která sloužila k ukázce funkčnosti programu jak v originálním tvaru, tak i po provedených kompresích.

Závěr

Hlavním cílem této bakalářské práce bylo seznámit čtenáře s problematikou a používáním kompresních programů. V první řadě bylo však zapotřebí rozebrat celý postup komprese, aby bylo zřejmé, co a jak je důležité.

Na začátku práce, byly popsány základní a zároveň nejdůležitější druhy algoritmů. V další části se práce zabývá kódováním, od vlastností, principu až po reálné využití.

Jako za jednu z nejdůležitějších částí práce považuji metody komprese, a to konkrétně popsání jednotlivých typů formátu souborů. Některé formáty jsou samozřejmě pro běžné uživatele známé, ale věřím, že i více znalý uživatel se po přečtení práce seznámí s vlastnostmi a využitím dosud ne moc známých typů souborů.

Dále jsou v práci ukázány již výše zmiňované kompresní programy. Zaměřili jsme se pouze na ty, co jsou nejpoužívanější a volně dostupné. Možnosti úpravy obrázků se v každém programu liší a podpora typu formátu souborů je také různá. V práci je zajímavé srovnání obrázků po kompresi, tj. výsledná velikost souboru vs. jeho kvalita. Je patrné, že i při celkem velké kompresi, je velikost souboru značně zmenšena a rozdíl výsledného obrázku od originálního, není mnohdy běžným okem rozeznatelný.

Bylo poukázáno, že jeden z často nejvyužívanějších formátů je formát JPEG. Na tento formát je také zaměřena praktická část, tj. vytvoření jednoduchého programu na zmenšení obrázků. Tento program je jednoduchý a pracuje velmi dobře. Na konci práce je srovnání stupňovité komprese s velikostí výstupního souboru a to i v grafické podobě. U použitého obrázku není okem rozeznatelný žádný rozdíl a to až při 70% kompresi. Jeho výsledná velikost je o 12,5x menší než původní, je to až neuvěřitelné. Zajímavostí je, že i při pouze 10% kompresi, je velikost obrázku značně zmenšena a rozdíl není znatelný ani při velkém přiblížení.

Použitá literatura

[1] Vlček, Karel. *Komprese a kódová zabezpečení v multimediálních komunikacích*. 1 vydání: BEN – technická literatura, 2004. 260 s. ISBN 80-7300-134-9

[2] Wikipedia, otevřená encyklopedie. *RLE* [online]. Stránka byla naposledy editována 10. 3. 2013 [cit. 2013-05-03]. Dostupné z WWW: http://cs.wikipedia.org/wiki/RLE

[3] Gimli web. *Algoritmy* [online] [cit. 2013-05-03]. Dostupné z WWW: http://gimli.mysteria.cz/komprese/algoritmy.html

[4] Miroslav Beneš, Katedra informatiky FEI VŠB-TU Ostrava *Komprese dat* [online] [cit. 2013-05-03]. Dostupné z WWW: http://www.cs.vsb.cz/benes/vyuka/pte/texty/komprese/index.html

[5] Primat. *Algoritmus* [online] 7. 12. 2011 [cit. 2013-05-02]. Dostupné z WWW: http://www.primat.cz/stredniskoly/predmety/programovani-a-vyvoj-aplikaci-q36955/algoritmus-m104867#

[6] Pavel Tišnovský, Root. *Typy souborů* [online] [cit. 2013-05-03]. Dostupné z WWW: http://www.root.cz/clanky/pcx-prakticky-implementace-komprimace-rle/

[7] Ing. Simona Martínková. *Komprimace* [online], 2009 [cit. 2013-05-04]. Dostupné z WWW: http://www.mgplzen.cz/download/ivt/ivt_komprimace.pdf

[8] Ing. Jan Malý, VUT Brno Fakulta elektrotechniky a komunikačních technologií. *Srovnání metod pro ztrátovou kompresi obrazu* [online] [cit. 2013-05-03]. Dostupné z WWW: http://www.elektrorevue.cz/clanky/06042/index.html#lit13

[9] Google Web. *Komprese rastrového obrazu* [online] [cit. 2013-05-06]. Dostupné z WWW: https://sites.google.com/site/xgrafika/komprese-rastroveho-obrazu

[10] Pavel Tišnovský, Root. *Ztrátová komprese obrazových dat pomocí JPEG* [online] [cit. 2013-05-03]. Dostupné z WWW: http://www.root.cz/clanky/ztratova-komprese-obrazovych-dat-pomoci-jpeg/

[11] Bc. Zlata Štědrová, *Grafika vektorová vs. bitmapová* [online] 2009 [cit. 2013-05-03]. Dostupné z WWW: http://lorenc.info/soubory/3MA481_pocitacova-grafika-teorie_xstez14.pdf

[12] Mike Williams, Netmagazin. *The programs* [online] 2013 [cit. 2013-05-11]. Dostupné z WWW: http://www.netmagazine.com/features/best-image-compression-tools

[13] Wikipedia, otevřená encyklopedie. *Bitmapová grafika* [online]. Stránka byla naposledy editována 13. 5. 2013 [cit. 2013-05-15] Dostupné z WWW: http://cs.wikipedia.org/wiki/Bitmapa

[14] Wikipedia, otevřená encyklopedie. *Algoritmus* [online]. Stránka byla naposledy editována 18. 3. 2013 [cit. 2013-04-19]. Dostupné z WWW: http://cs.wikipedia.org/wiki/Algoritmus

[15] Wikipedia, otevřená encyklopedie. *Jpeg* [online] [cit. 2013-05-01]. Stránka byla naposledy editována 30. 4. 2013. Dostupné z WWW: http://cs.wikipedia.org/wiki/JPEG

[16] Wikipedia, otevřená encyklopedie. *Bezeztrátová komprese* [online]. Stránka byla naposledy editována 13. 5. 2013 [cit. 2013-05-17]. Dostupné z WWW: http://cs.wikipedia.org/wiki/Bezeztr%C3%A1tov%C3%A1_komprese

[17] Wikipedia, otevřená encyklopedie. *JPEG 2000* [online]. Stránka byla naposledy editována 17. 3. 2013 [cit. 2013-03-28]. Dostupné z WWW: http://cs.wikipedia.org/wiki/JPEG_2000

[18] Jason Comentale, Duggal. *CMYK vs. RGB* [online] 2011. Dostupné z WWW: http://duggal.com/connect/getting-started/cmyk-vs-rgb

[19] Ryšavý, *Počítačová grafika* [online]. Dostupné z WWW: https://sites.google.com/site/ppaffuhk/statnicove-otazky/15-pocitacova-grafika-rysavy

[20] Grafické studio Marionetti Ostrava, *Pojmy grafického designu* [online] [cit. 2013-04-07]. Dostupné z WWW: http://www.marionetti.eu/pojmy_grafickeho_designu.php

[21] PC Blog, *Průhledné PNG v IE6* [online]. Dostupné z WWW: http://www.pcblog.cz/clanek/2481/pruhledne-png-v-ie6-jednoduchy-zpusob/

[22] Václav Hlaváč, Fakulta elektrotechnická ČVUT Praha, Katedra kybernetiky, *Komprese obrazů* [online] [cit. 2013-05-01]. Dostupné z WWW: http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/22DigFoto/81KompreseObrazu.pdf

[23] Vladimír Josefy, *Detailní popis kompresní metody formátu JPEG.* [online] 2002 [cit. 2013-05-03]. Dostupné z WWW: http://josefy.sweb.cz/formaty/JPEG_ADV.htm

[24] Slunečnice, Programy rychle a zadarmo, *JPEG Resampler* [online]. Poslední aktualizace 9. 2. 2012 [cit. 2013-05-09]. Dostupné z WWW: http://www.slunecnice.cz/sw/jpeg-resampler/

[25] Slunečnice, Programy rychle a zadarmo, *Image Optimizer* [online]. Poslední aktualizace 5. 3. 2006 [cit. 2013-05-09]. Dostupné z WWW: http://www.slunecnice.cz/sw/image-optimizer/

[26] Slunečnice, Programy rychle a zadarmo, *Image Converter Plus* [online]. Poslední aktualizace 9. 12. 2006 [cit. 2013-05-10]. Dostupné z WWW: http://www.slunecnice.cz/sw/image-converter-plus/

[27] Slunečnice, Programy rychle a zadarmo, *RIOT* [online]. Poslední aktualizace 22. 4. 2011 [cit. 2013-05-10]. Dostupné z WWW: http://www.slunecnice.cz/sw/image-converter-plus/

[28] Stahuj, Svět software, *JPEG Enhancer* [online]. Poslední aktualizace 26. 5. 2005 [cit. 2013-05-09]. Dostupné z WWW: http://www.stahuj.centrum.cz/grafika_a_design/prevody_a_optimalizace/jpegfixer/

[29] Stahuj, Svět software, *Caesium* [online]. Poslední aktualizace 22. 7. 2012 [cit. 2013-05-11]. Dostupné z WWW: http://www.stahuj.centrum.cz/utility_a_ostatni/komprese/caesium/

[30] Wikipedia, The Free Encyklopedia. *Bezeztrátová komprese* [online]. Stránka byla naposledy editována 21. 3. 2012. Dostupné z WWW:

http://en.wikipedia.org/wiki/File:JPEG_ZigZag.jpg

[31] Wikipedia, otevřená encyklopedie. *RGB* [online]. Stránka byla naposledy editována 5. 5. 2013 [cit. 2013-05-08] Dostupné z WWW: http://cs.wikipedia.org/wiki/RGB

[32] Wikipedia, otevřená encyklopedie. *CMYK* [online]. Stránka byla naposledy editována 4. 5. 2013 [cit. 2013-05-11] Dostupné z WWW: http://cs.wikipedia.org/wiki/CMYK

[33] Wikipedia, otevřená encyklopedie. *Heuristické algoritmy* [online]. Stránka byla naposledy editována 14. 3. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/Heuristick%C3%A9_algoritmy

[34] Wikipedia, otevřená encyklopedie. *Genetické algoritmy* [online]. Stránka byla naposledy editována 4. 5. 2013 [cit. 2013-05-10] Dostupné z WWW: http://cs.wikipedia.org/wiki/Genetick%C3%A9_algoritmy

[35] Wikipedia, otevřená encyklopedie. *Pravděpodobnostní algoritmus* [online]. Stránka byla naposledy editována 8. 3. 2013 [cit. 2013-05-11] Dostupné z WWW: http://cs.wikipedia.org/wiki/Pravd%C4%9Bpodobnostn%C3%AD_algoritmus

[36] Wikipedia, otevřená encyklopedie. *BMP* [online]. Stránka byla naposledy editována 9. 3. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/BMP

[37] Wikipedia, otevřená encyklopedie. *PCX* [online]. Stránka byla naposledy editována 4. 5. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/PCX

[38] Wikipedia, otevřená encyklopedie. *GIF* [online]. Stránka byla naposledy editována 13. 3. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/GIF

[39] Wikipedia, otevřená encyklopedie. *PNG* [online]. Stránka byla naposledy editována 19. 4. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/PNG

[40] Wikipedia, otevřená encyklopedie. *B-W transformace* [online]. Stránka byla naposledy editována 4. 5. 2013 [cit. 2013-05-12] Dostupné z WWW: http://cs.wikipedia.org/wiki/Burrowsova-Wheelerova_transformace

Seznam obrázků

Seznam tabulek

Seznam příloh

Příloha A – Postup zpětné Burrows – Wheelerovy transformace

Vstup

PLLAA^@A

1 známý sloupec	seřazené	2 známé sloupce	seřazené
P	A	PA	AL
L	A	LA	AL
L	A	LA	A@
A	L	AL	LA
A	L	AL	LA
^	P	^P	PA
@	^	@^	^P
A	@	A@	@^

3 známý sloupec	seřazené	4 známé sloupce	seřazené
PAL	ALA	PALA	ALAL
LAL	ALA	LALA	ALA@
LA@	A@^	LA@^	A@^P
ALA	LAL	ALAL	LALA
ALA	LA@	ALA@	LA@^
^PA	PAL	^PAL	PALA
@^P	^PA	@^PA	^PAL
A@^	@^P	A@^P	@^PA

5 známých sloupů	seřazené	6 známých sloupců	seřazené
PALAL	ALALA	PALALA	ALALA@
LALA@	ALA@^	LALA@^	ALA@^P
LA@^P	A@^PA	LA@^PA	A@^PAL
ALALA	LALA@	ALALA@	LALA@^
ALA@^	LA@^P	ALA@^P	LA@^PA
^PALA	PALAL	^PALAL	PALALA
@^PAL	^PALA	@^PALA	^PALAL
A@^PA	@^PAL	A@^PAL	@^PALA

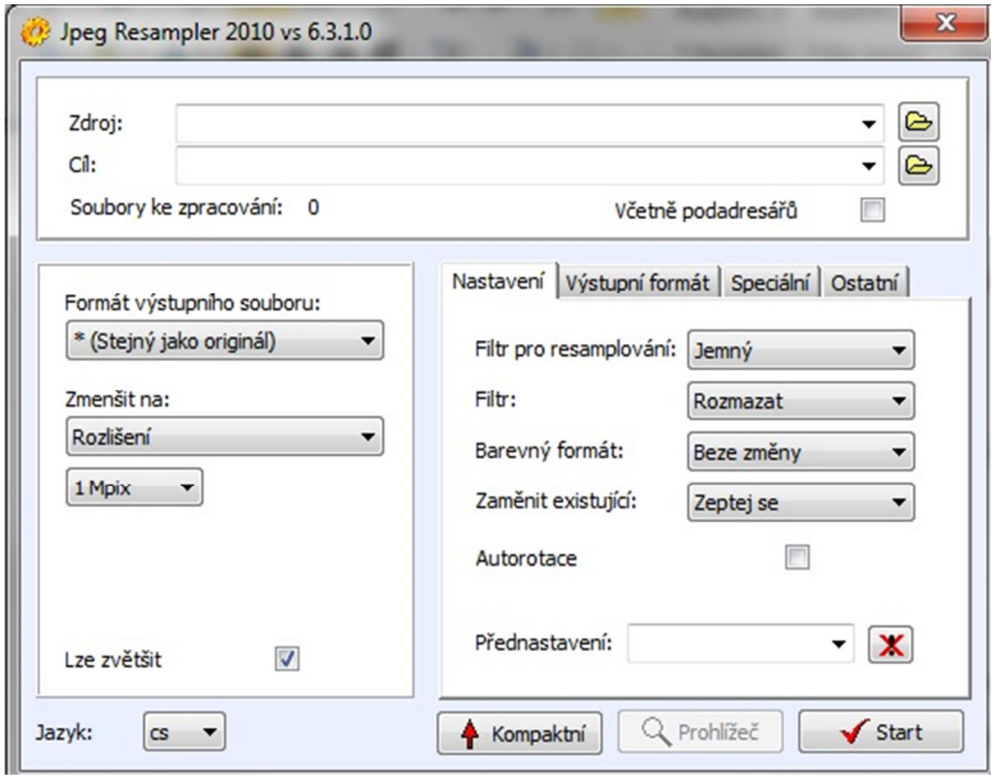
7 známých sloupů	seřazené	8 známých sloupců	seřazené
PALALA@	ALALA@^	PALALA@^	ALALA@^P
LALA@^P	ALA@^PA	LALA@^PA	ALA@^PAL
LA@^PAL	A@^PALA	LA@^PALA	A@^PALAL
ALALA@^	LALA@^P	ALALA@^P	LALA@^PA
ALA@^PA	LA@^PAL	ALA@^PAL	LA@^PALA
^PALALA	PALALA@	^PALALA@	PALALA@^
@^PALAL	^PALALA	@^PALALA	^PALALA@
A@^PALA	@^PALAL	A@^PALAL	@^PALALA

Výstup

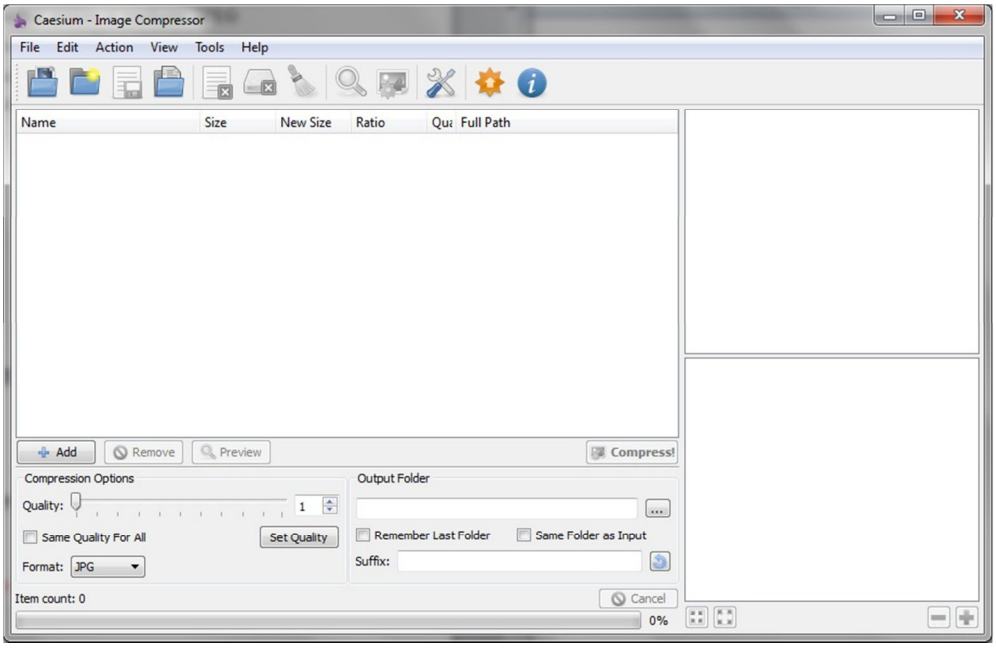
^PALALA@

Příloha B – Programy na úpravu obrázků

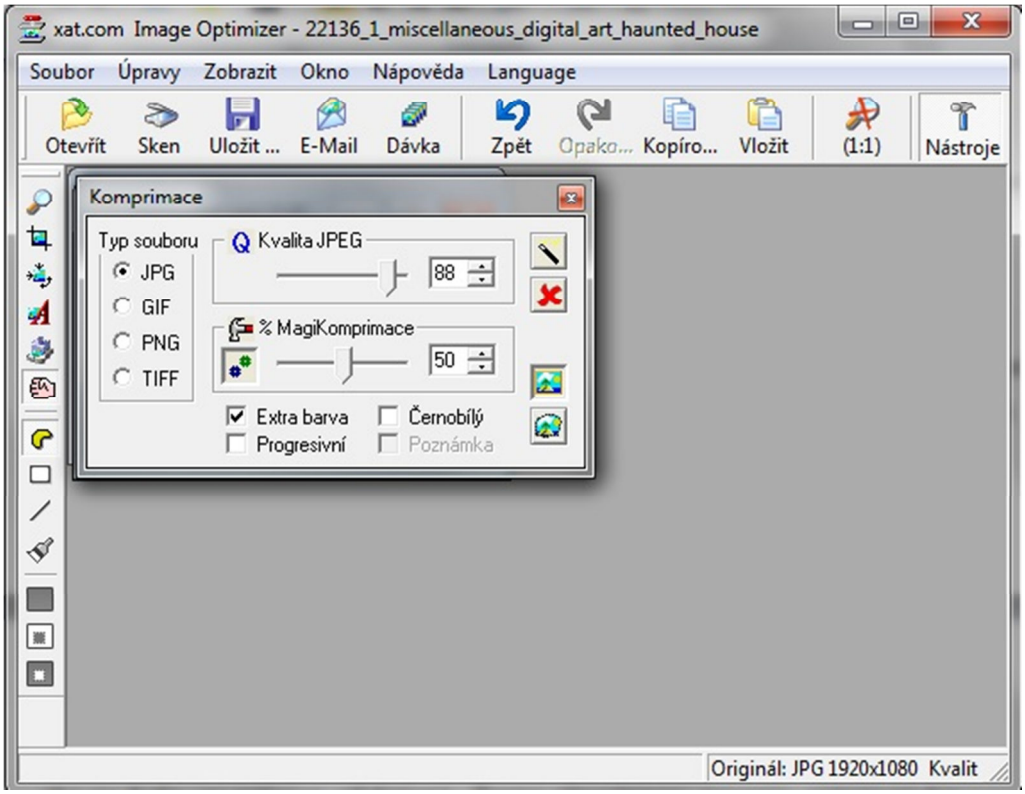
Vzhled programu Jpeg Resampler



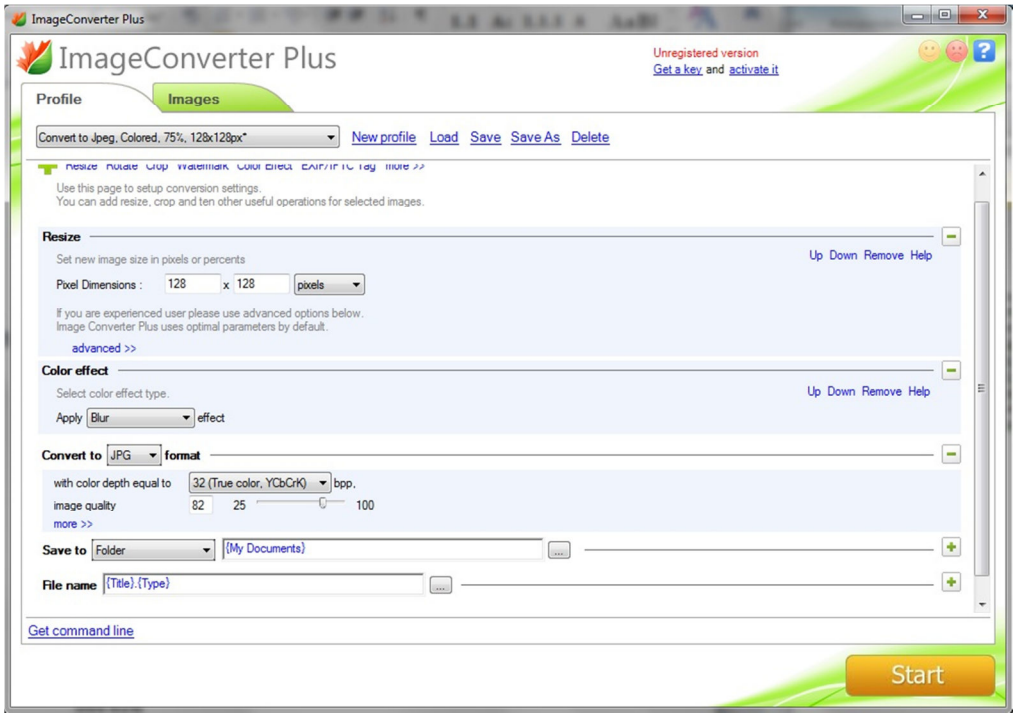
Vzhled programu Caesium



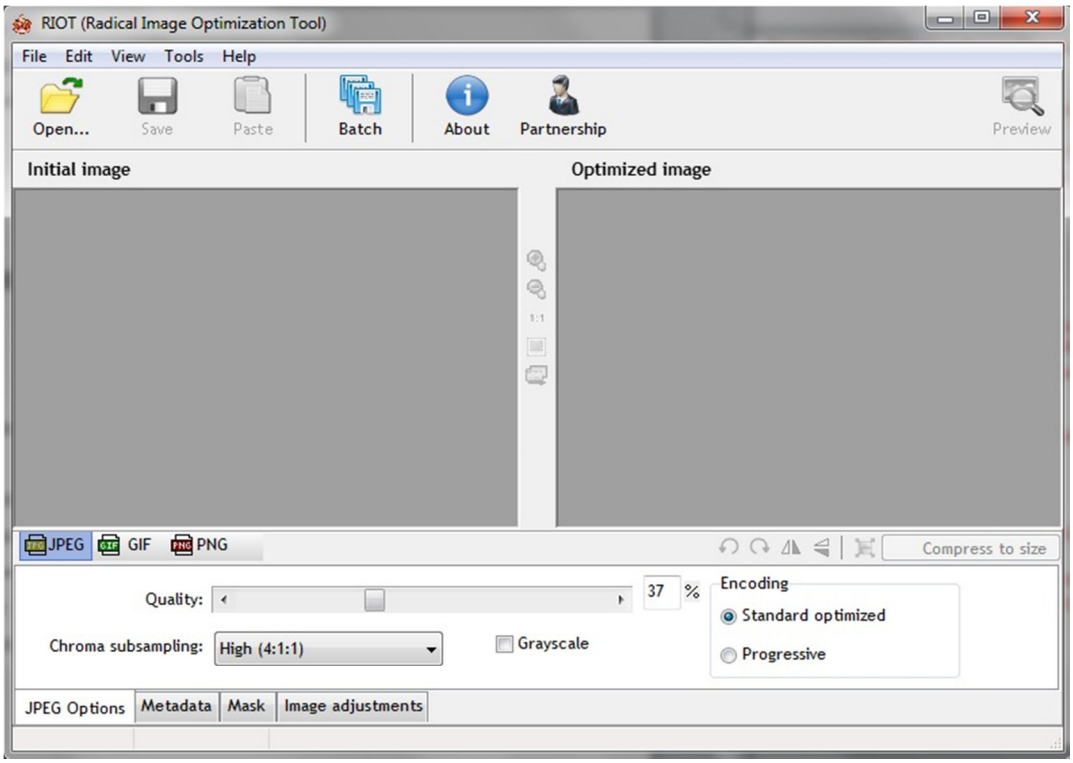
Vzhled programu Image Optimizer



Vzhled programu Image Converter



Vzhled programu RIOT



Příloha C – Srovnávané obrázky podle typu souboru

Název	Typ souboru	Rozlišení	Bitová hloubka [bit]	Velikost [kB]
Muž [zdroj]	GIF	640x480	8	218
Gepard	JPEG	640x480	24	86
Poháry	PNG	640x480	32	602
Psy	BMP	640x480	24	900
Veverka	GIF	800x600	8	327
Pták	JPEG	800x600	24	136
Zebra	PNG	800x600	32	1120
Bagr	BMP	800x600	24	1370
Tukan	GIF	1024x768	8	617
Pes	JPEG	1024x768	24	257
Dívka	PNG	1024x768	32	1250
Balón	BMP	1024x768	24	2250
Autobus	GIF	1600x1200	8	869
Tulipány	JPEG	1600x1200	24	215
Kůň	PNG	1600x1200	24	2370
Město	BMP	1600x1200	24	5490
Prasátka	GIF	2816x2112	8	3260
Jezero	JPEG	2816x2112	24	571
Srnka	PNG	2816x2112	32	9780
Loď	BMP	2816x2112	24	17000
Passat	GIF	3264x2448	8	3840
Pták	JPEG	3264x2448	24	1500
Útes	PNG	3264x2448	24	12600
Velbloud	BMP	3264x2448	24	22800
Kůň	GIF	3872x2592	8	4500
Sup	JPEG	3872x2592	24	1710
Kajak	PNG	3872x2592	24	14000
Pistole	BMP	3872x2592	24	28700
Babička	GIF	5616x3744	8	6080
Formule	JPEG	5616x3744	24	3760
Letovisko	PNG	5616x3744	24	30600
Krajina	BMP	5616x3744	24	60100

Příloha D – Kvantizační tabulka

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	61
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	89	98	103	121	120	101
72	92	95	98	112	100	103	99

Příloha E – Náhled testovaného obrázku s konkrétní kompresí

Komprese 50%

Komprese 70%

Komprese 90%

Komprese 99%

Příloha F – Vývojový diagram programu