MULTIMEDIÁLNÍ A HYPERMEDIÁLNÍ SYSTÉMY

9)
Uložení a komprese
statického bitmapového obrazu

Petr Lobaz, 28.4.2015

BITMAPOVÝ OBRAZ

PŮVOD OBRAZOVÝCH DAT

- (kreslicí) software
  - data typicky připravena k přímé manipulaci a zobrazení
- obrazový snímač
  - data představují výstup základní OETF (opto-electronic transfer function), k manipulaci a zobrazení je třeba data upravit – tzv. „*raw data*"
  - při úpravě dochází ke ztrátě informace a (úmyslnému) zkreslení ⇒ uložení raw dat odkládá důležitá rozhodnutí do postprodukce
- digitální fotoaparáty a kamery obvykle nabízí data zpracovaná k zobrazení a manipulaci, „lepší" zařízení pak i přímé uložení raw dat

BITMAPOVÝ OBRAZ

ULOŽENÍ BITMAPOVÝCH DAT

- nekomprimovaná – typicky v paměti počítače
- bezztrátově komprimovaná
  - maximální kvalita obrazu
  - není zaručen kompresní poměr ⇒ hardware musí být stejně výkonný jako pro nekomprimovaná data
  - typicky pro počítačovou grafiku, raw data, uložení pracovních meziverzí obrazu
- ztrátově komprimovaná
  - snaha o maximální vizuální věrnost při minimálním datovém toku (maximálním kompresním poměru)
  - ve statickém obrazu se nejčastěji nastavuje podle požadované kvality, ve videu často podle datového toku

BITMAPOVÝ OBRAZ

STRUKTURA OBRAZOVÉHO SOUBORU

- informace o počtu vzorků na šířku/výšku, typ barevného prostoru (RGB, stupně šedi, …), počet bitů na pixel
- detailní informace o barevném prostoru (ICC barevný profil)
  - návod, jak transformovat obrazová data do platformně nezávislého barevného prostoru (CIELAB nebo CIEXYZ)
- informace o obrazu samotném
  - typ fotoaparátu, expoziční parametry, datum pořízení, apod. – typicky ve formátu EXIF (Exchangeable Image File Format)
  - slovní popis, klíčová slova apod. – typicky ve formátu IPTC (International Press Telecommunications Council)
- samotná obrazová data

BITMAPOVÝ OBRAZ

- ne všechny položky jsou nutné
- tzv. *obrazový formát* definuje způsob uložení informace
  - některé formáty těsně spjaté s konkrétním způsobem uložení obrazových dat (např. typem komprese), např. JPEG, PNG, GIF
  - některé formáty obecnější, podporují mnoho způsobů uložení obrazových dat, např. TIFF
- ve speciálním případě obsahuje soubor pouze obrazová data, pokud jsou parametry pro dekódování zřejmé z kontextu (např. soubory YUV ve zpracování videa)
  - někdy se takovým souborům říká „raw" – nesouvisí s raw daty z obrazového snímače, v dalším nebudeme tento význam používat!

ZPRACOVÁNÍ RAW

ZPRACOVÁNÍ DAT Z OBRAZOVÉHO SNÍMAČE

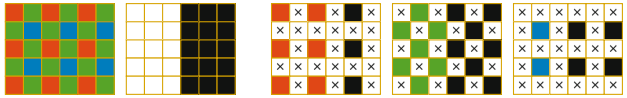
- obrazové snímače typicky obsahují elementy citlivé na energii dopadajícího světla (nezávisí na vlnové délce světla)
  - spektrální citlivost nejčastěji zařízena Bayerovou maskou
  - některé elementy mají před sebou červený, jiné zelený, jiné modrý filtr
  - ⇒ snímají jen R, G, B část spektra
  - pro zobrazení je třeba v každém elementu (pixelu) dopočítat zbývající složky (demosaicking)

Bayerova maska

ZPRACOVÁNÍ RAW

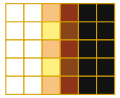
DEMOSAICKING

- odhaduje chybějící složky pixelu ⇒ vždy je zatížen chybou
- příklad: snímání rozhraní bílá (složky RGB = 100) a černá (složky RGB = 0)



Bayerova maska snímaný obraz

zaznamenané energie R, G, B (maximum nebo nula)



demosaicking – odhad RGB energií

interpolované energie R, G, B

ZPRACOVÁNÍ RAW

100 × ×	× 100 ×	100 × ×	× 0 ×	0 × ×	× 0 ×
× 100 ×	× × 100	× 100 ×	× × 0	× 0 ×	× × 0
100 × ×	× 100 ×	100 × ×	× 0 ×	0 × ×	× 0 ×
× 100 ×	× × 100	× 100 ×	× × 0	× 0 ×	× × 0
100 × ×	× 100 ×	100 × ×	× 0 ×	0 × ×	× 0 ×

zaznamenané RGB složky spektra
(× značí „kvůli Bayerově masce nezjištěno“)

100 100 100	100 100 100	100 75 50	50 0 0	0 0 0	0 0 0
100 100 100	100 100 100	100 100 50	50 25 0	0 0 0	0 0 0
100 100 100	100 100 100	100 75 50	50 0 0	0 0 0	0 0 0
100 100 100	100 100 100	100 100 50	50 25 0	0 0 0	0 0 0
100 100 100	100 100 100	100 75 50	50 0 0	0 0 0	0 0 0

interpolované RGB složky spektra
jednoduchou (bi)lineární interpolací z okolí

ZPRACOVÁNÍ RAW

- při demosaickingu typicky poškozena informace o barevném odstínu
- náprava – odstranění vysokofrekvenční složky signálu, která k podobným vadám vede
  ⇒ rozostření obrazu
- algoritmus demosaickingu je kompromisem mezi ostrostí a artefakty

ZPRACOVÁNÍ RAW

LINEARIZACE, REDUKCE ŠUMU

- odečtení složky odpovídající tepelnému šumu
- kvůli chybám ve výrobě snímače nereagují všechny pixely stejně – jejich reakce se ale dá změřit a nedokonalosti kompenzovat

MATRIXING

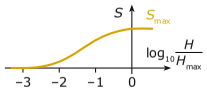
- tvorba lineární kombinace R_M, G_M, B_M z interpolovaných složek R, G, B
- aproximace color matching functions
- úpravou lineární kombinace se dosáhne stavu, kdy je pro jisté spektrum světla výstup R_M, G_M, B_M stejný
  ⇒ vyvážení bílé (white balance)

ZPRACOVÁNÍ RAW

DALŠÍ ÚPRAVY SIGNÁLU

- nelineární úprava barvy – typicky založeno na vyhledávací tabulce (LUT – look-up table)
- simulace senzitometrické charakteristiky
  - složky $R_M G_M B_M$ (v grafu svislá osa – obecná složka S) lineárně závisí na vstupním jasu
    ⇒ extrémní světla a stíny jsou oříznuty
  - výhodnější je jejich částečné zachování pomocí nelineárního kódování světlých a tmavých jasů
- doostření – kompenzace nenulové plochy snímacího pixelu

dynamický rozsah 100 : 1
(cca 6,6 EV)



dynamický rozsah 1000 : 1
(cca 10 EV)

ZPRACOVÁNÍ RAW

GAMA KÓDOVÁNÍ

- aplikace vhodné přenosové funkce (viz OETF, přednáška 8)
- kvantizace na požadovaný počet bitů

- může být doplněno převodem na $Y'C_BC_R$, chroma subsamplingem apod.

- raw data obsahují pouze kvantizovaný (typicky na 12–14 bitů) výstup ze znímače
- obvykle jsou „raw formáty" (CR2 – Canon, NEF – Nikon apod.) těsně spjaté s konkrétní kamerou
- formát DNG (Digital Negative) – platformově nezávislý standard pro uložení raw snímků

KOMPRESE OBRAZU

BEZZTRÁTOVÁ KOMPRESE

- run length encoding (RLE)
- entropické kódování
  - Huffmanovo kódování a jiná kódování s proměnnou délkou kódového slova (variable length coding – VLC)
  - aritmetické kódování
- slovníkové metody – LZ77, LZW

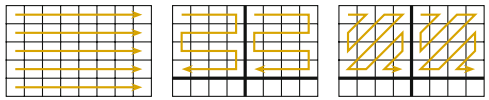
ZTRÁTOVÁ KOMPRESE

- obvykle fourierovská transformace nebo waveletová transformace následovaná kvantizací
- fraktálová komprese, triangularizační přístupy, …
  - v praxi se zatím nepoužívají

BEZZTRÁTOVÁ KOMPRESE

RUN LENGTH ENCODING (RLE)

- de facto jen dekorelace signálu
- redukce opakujících se prvků, např. AAAABBA ⇒ 4A2B1A
  - prvkem mohou být bity, byty nebo pixely
- průchod pixely po řádcích nebo po blocích (různé způsoby)
  - snaha vytvořit co nejdelší sekvence opakujících se hodnot
- varianty pro lepší kompresní poměr, pro speciální použití (např. JPEG – kódování delších sekvencí nul)



průchod po řádcích | standardní průchod po blocích | zig-zag průchod po blocích

BEZZTRÁTOVÁ KOMPRESE

LZW, LZ77 APOD.

- nahrazení známé sekvence znaků kódem
- adaptivní – během kódování se vytváří slovník známých sekvencí, během dekódování jej dekodér rekonstruuje a používá
- efektivní, když se dá výskyt opakujících se sekvencí čekat
  ⇒ prakticky jen počítačová grafika s malým počtem barev (typicky max. 256)
- LZW: patentovaná technologie (dnes už patent vypršel)
  - první křiklavý případ „zvláštního" nakládání s licencí
  - formát GIF je plně založený na LZW; až poté, co se masově rozšířil, začal vlastník patentu požadovat licenční poplatky (reakce: formát PNG založený na obdobě LZW)

BEZZTRÁTOVÁ KOMPRESE

ARITMETICKÉ KÓDOVÁNÍ

- přiřazuje jeden kód celé zprávě (skupině znaků)
  ⇒ jeden znak může „zabírat" necelý počet bitů
- praktické implementace jsou patentované
- příklad: zpráva $AAAB_{EOF}$ (znak EOF = konec zprávy)
  pravděpodobnosti znaků: $p_A = 3/5 \quad p_B = 1/5 \quad p_{EOF} = 1/5$

…	$\in \langle 0 ; 1)$	
A…	$\in \langle 0 ; \frac{3}{5})$	
AA…	$\in \langle 0 ; \frac{9}{25})$	
AAA…	$\in \langle 0 ; \frac{27}{125})$	
AAAB…	$\in \langle \frac{81}{625} ; \frac{108}{625})$	
AAAB	$\in \langle \frac{513}{3125} ; \frac{540}{3125})$	
AAAB ⇒ 00101**1**		⟨0,00101**0**100000…; 0,00101**1**000011…)

ARITMETICKÉ KÓDOVÁNÍ

- čísla p_i vyjádřena zlomky P_1/T, P_2/T, …, $T = \Sigma_i P_i$
  - P_i a T celá čísla s omezenou bitovou hloubkou
- v každém okamžiku kódování je pravděpodobnost zprávy v intervalu ⟨LO, HI), čísla LO, HI s omezenou bit. hloubkou
- jakmile je interval dostatečně malý
  - je zřejmé, jaký bude první bit kódované zprávy – lze jej odeslat na výstup
  - hodnoty LO, HI jsou si blízké, čili nejvýznamnější bity spolu úzce souvisí (mohou být i stejné) ⇒ není třeba je uchovávat
  - hodnoty LO, HI je možné „přenásobit" (normalizace intervalu způsoby E_1, E_2, E_3)

  ⇒ omezení počtu bitů nutných k uchování LO, HI

ARITMETICKÉ KÓDOVÁNÍ

Normalizace typu E_1, E_2, E_3 může proběhnout, pokud HI – LO < 1/2 (interval je dostatečně krátký)

E_1 proběhne, pokud HI < 1/2

⇒ první bit bude 0

⇒ LO := LO × 2; HI := HI × 2

E_2 proběhne, pokud LO > 1/2

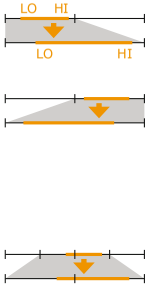
⇒ první bit bude 1

⇒ LO := LO × 2 – 1; HI := HI × 2 – 1

E_3 proběhne, pokud 1/4 < LO < HI < 3/4

⇒ hodnotu výstupního bitu sice neznáme, ale jakmile časem proběhne E_1 nebo E_2, tj. na výstup odešleme bit x, bude další bit určitě negace(x)

⇒ LO := LO × 2 – 1/2; HI := HI × 2 – 1/2



ARITMETICKÉ KÓDOVÁNÍ

KOMPRESE

LO := 0; HI := 1; E_3bitů := 0;
A := vstup(); /* EOF ⇒ vnější smyčka už se nevrátí */
zkrať LO, HI podle pravděpodobnosti A
- lze provést E_1?
  E_1; výstup(0); E_3bitů × výstup(1); E_3bitů := 0;
- lze provést E_2?
  E_2; výstup(1); E_3bitů × výstup(0); E_3bitů := 0;
- lze provést E_3?
  E_3; E_3bitů++;

výstup bitů pro výsledný interval a zbylých E_3 bitů

ARITMETICKÉ KÓDOVÁNÍ

DEKOMPRESE

LO := 0; HI := 1;
do C načti maximum bitů (např. 31)
(⇒ připojím-li 0/1, patří C stále stejnému znaku A)
- výstup(A)
  zkrať LO, HI podle pravděpodobnosti A:
  - lze provést E_1 nebo E_2?
    E_1 ⇒ E_1; C := C*2+vstup()
    E_2 ⇒ E_2; nuluj nejvyšší bit C; C := C*2+vstup()
  - lze provést E_3?
    E_3; nuluj druhý nejvyšší bit C; C := C*2+vstup()

ARITMETICKÉ KÓDOVÁNÍ

DALŠÍ VLASTNOSTI

- výpočet odhadu pravděpodobnosti znaku lze po každém znaku adaptovat
- po každém znaku lze měnit vstupní abecedu
- v současnosti nejpoužívanější algoritmus: CABAC (Content Adaptive Binary Arithmetic Coding)

- výsledný interval lze kódovat i v jiné než dvojkové soustavě
  – šlo by použít např. pro kódování dlouhých URL adres s mnoha parametry – tvorba „krátkých URL“ (ale v praxi se to dělá jinak)

DCT

DISKRÉTNÍ KOSINOVÁ TRANSFORMACE

- pro připomenutí: signál $x[0]$, $x[1]$, …, $x[N-1]$ převedeme do frekvenční podoby diskrétní Fourierovou transformací:

$$X[k] = \frac{1}{N} \sum_{j=0}^{N-1} x[j] \exp\left(-\mathrm{i}2\pi \frac{jk}{N}\right)$$

- de facto odpovídá stavu, kdy je signál $x[]$ periodický s periodou N a zkoumáme jen jednu jeho periodu

⇒ po vzorku $x[N-1]$ de facto následuje vzorek $x[0]$
⇒ mezi vzorky $x[N-1]$ a $x[0]$ se dá očekávat „skok“
⇒ tento „skok“ se projevuje zvýšením amplitud vysokých frekvencí
⇒ pro uchování kvalitní podoby $x[]$ nesmíme kompresí příliš poškodit vysoké frekvence ⇒ nevýhodné

DCT

- diskrétní Fourierovu transformaci můžeme zapsat i bez komplexních čísel; inverzní transformace:

$$x[j] = \sum_{k=0}^{N/2} a_C[k] \cos\left(2\pi \frac{jk}{N}\right) + a_S[k] \sin\left(2\pi \frac{jk}{N}\right)$$

- koeficienty $a_C[k]$ jsou dány dopřednou transformací:

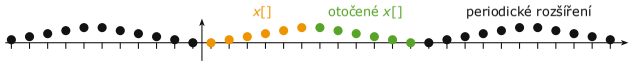
$$a_C[k] = \begin{cases} a'_C[0] & \text{pro } k = 0 \\ a'_C[N/2] & \text{pro } k = N / 2 \\ 2a'_C[k] & \text{jinak} \end{cases}$$

$$\text{kde} \quad a'_C[k] = \frac{1}{N} \sum_{j=0}^{N-1} x[j] \cos\left(2\pi \frac{jk}{N}\right)$$

- koeficienty $a_S[k]$ jsou dány obdobným vztahem, ve výpočtu $a'_S[k]$ figuruje funkce sin()

DCT

- udělejme ze signálu $x[]$ délky N signál $y[]$ délky $2N$:
  $x[0], x[1], \ldots, x[N-1], x[N-1], x[N-2], \ldots, x[1], x[0]$
  a posuňme jej o 1/2 vzorkovací vzdálenosti „doprava"



- periodicky rozšířený signál $y[]$ je sudý

⇒ v jeho reálné DFT nemohou figurovat funkce sinus

⇒ výstupem reálné DFT je N koeficientů a_c (všechny $a_S = 0$)

⇒ *diskrétní kosinová transformace* (DCT) signálu $x[]$

- existuje více způsobů, jak sudé periodické rozšíření udělat

⇒ existuje více typů DCT

DCT

DVOUROZMĚRNÁ DCT

- aplikace DCT na všechny sloupce, pak na všechny řádky výsledku
- dá se přepsat do jediného vztahu, např. pro vstupní signál $s[i, j]$ je DCT rovna $S[u, v]$:

$$S[u, v] = \sqrt{\frac{2}{M}}\sqrt{\frac{2}{N}}\, c_u\, c_v \times \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} s[i, j] \cos\left(\pi\frac{(2i+1)u}{2N}\right) \cos\left(\pi\frac{(2j+1)v}{2N}\right)$$

kde $c_u = \begin{cases} 1/\sqrt{2} & \text{pro } u = 0 \\ 1 & \text{jinak} \end{cases}$ $c_v = \begin{cases} 1/\sqrt{2} & \text{pro } v = 0 \\ 1 & \text{jinak} \end{cases}$

DCT

- IDCT je v tomto případě rovna:

$$s[i, j] = \sqrt{\frac{2}{M}}\sqrt{\frac{2}{N}} \times \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} c_u\, c_v\, S[u, v] \cos\left(\pi\frac{(2i+1)u}{2N}\right) \cos\left(\pi\frac{(2j+1)v}{2N}\right)$$

kde $c_u = \begin{cases} 1/\sqrt{2} & \text{pro } u = 0 \\ 1 & \text{jinak} \end{cases}$ $c_v = \begin{cases} 1/\sqrt{2} & \text{pro } v = 0 \\ 1 & \text{jinak} \end{cases}$

- obdobně se dá udělat vícerozměrná DCT, resp. DFT
- pro výpočet DCT se dá využít rychlý algoritmus FFT

JPEG

- správný název JFIF = JPEG File Interchange Format
  JPEG = Joint Photographic Experts Group
- ztrátový kodér obrazu zaměřený na fotografie
  - obrázek typicky desítky kilopixelů a více
  - obrázek obsahuje především hladké přechody a minimum ostrých hran
  - obrázek obsahuje mírný šum
- původní standard definován pro šedotónové a RGB obrázky, 8bitová kvantizace, gama kódování
- pro typický RGB obrázek očekávaný kompresní poměr cca 1 : 10 až 1 : 20 (cca 1–2 bity/pixel) pro vizuálně bezztrátovou kompresi
  - kompresní poměr je vždy nutné hodnotit v souvislosti s kvalitou obrazu

JPEG

PRŮBĚH KOMPRESE

- převod RGB → $Y'C_BC_R$
- chroma subsampling; dále práce s kanály $Y'C_BC_R$ nezávislá
  - chroma subsampling je velmi mírná ztrátová komprese
- rozdělení na bloky 8 × 8 vzorků (velikost kompromis mezi výpočetní náročností a efektivitou komprese)
- aplikace DCT na blok
- kvantizace („zaokrouhlení") výsledku DCT
  - jediný významný ztrátový krok
  - volbou kvantizační tabulky je možné ovlivňovat kvalitu výsledku, resp. kompresní poměr
- převod 8 × 8 koeficientů zig-zag průchodem na posloupnost 64 koeficientů
- speciální RLE kódování + entropické kódování

JPEG

původní obrázek

obrázek po kvantizaci výsledku DCT a dekompresi (dekvantizaci a IDCT)

šipka ukazuje typické artefakty – ztráta detailů, rozpad do bloků 8×8 pixelů

JPEG

blok z původního obrázku (na minulém snímku vyznačen čtverečkem)

blok po kvantizaci DCT koeficientů a následné dekompresi

tabulka kvantizačních koeficientů

◄ *ilustrativní*
reálná ►

1	2	3	4	5	6	7	8
2	4	6	8	10	12	14	16
3	6	9	12	15	18	21	24
4	8	12	16	20	24	28	32
5	10	15	20	25	30	35	40
6	12	18	24	30	36	42	48
7	14	21	28	35	42	49	56
8	16	24	32	40	48	56	64

malé hodnoty — velké hodnoty

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	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

JPEG

31	46	62	88	97	95	92	82
83	108	98	126	158	111	97	90
103	131	124	116	151	144	104	104
136	144	133	127	134	151	133	134
191	216	196	148	136	148	143	117
139	203	216	240	237	147	107	65
122	118	176	255	255	161	58	21
141	122	131	172	166	99	29	3

blok z původního obrázku (stejný jako na minulém snímku)

128.30	19.57	-34.22	5.22	2.76	-5.36	-2.64	0.35
-23.72	-38.10	30.56	-2.97	-8.02	-2.80	-0.62	0.16
-30.40	-3.40	-22.22	4.72	13.45	2.99	-0.53	0.13
9.86	1.54	-8.32	-7.19	-3.15	2.54	-4.25	1.23
-11.60	-2.46	26.32	-0.27	-6.22	4.87	3.20	-2.16
-3.22	-12.21	-12.13	2.29	4.46	2.22	4.11	-1.82
-2.65	1.89	1.90	-3.03	-4.19	-2.38	-1.40	0.47
-0.10	1.91	3.35	1.91	-3.33	0.89	3.91	-0.19

jeho DCT koeficienty (tj. výsledek DCT; normalizační konstanta v definici DCT)

je pro přehlednost volena tak, aby prvek na pozici [0,0], tj. v levém horním rohu, vyšel roven aritmetickému průměru hodnot bloku)

JPEG

128.30	19.57	-34.22	5.22	2.76	-5.36	-2.64	0.35
-23.72	-38.10	30.56	-2.97	-8.02	-2.80	-0.62	0.16
-30.40	-3.40	-22.22	4.72	13.45	2.99	-0.53	0.13
9.86	1.54	-8.32	-7.19	-3.15	2.54	-4.25	1.23
-11.60	-2.46	26.32	-0.27	-6.22	4.87	3.20	-2.16
-3.22	-12.21	-12.13	2.29	4.46	2.22	4.11	-1.82
-2.65	1.89	1.90	-3.03	-4.19	-2.38	-1.40	0.47
-0.10	1.91	3.35	1.91	-3.33	0.89	3.91	-0.19

DCT koeficienty bloku (pro přehlednost opsáno z minulého snímku)

128.00	20.00	-33.00	4.00	5.00	-6.00	0.00	0.00
-24.00	-40.00	30.00	0.00	-10.00	0.00	0.00	0.00
-30.00	-6.00	-18.00	0.00	15.00	0.00	0.00	0.00
8.00	0.00	-12.00	0.00	0.00	0.00	0.00	0.00
-10.00	0.00	30.00	0.00	0.00	0.00	0.00	0.00
-6.00	-12.00	-18.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

DCT koeficienty po kvantizaci pomocí ilustrativní kvantizační tabulky

(pro názornost jsou čísla dekvantizována, tj. vynásobena kvantizačními koeficienty; např. prvek –34,22 na pozici [3,1] je kvantizován na round(–34,22 / 3) = –11, po dekvantizaci –11 × 3 = –33)

JPEG

31	46	62	88	97	95	92	82
83	108	98	126	158	111	97	90
103	131	124	116	151	144	104	104
136	144	133	127	134	151	133	134
191	216	196	148	136	148	143	117
139	203	216	240	237	147	107	65
122	118	176	255	255	161	58	21
141	122	131	172	166	99	29	3

blok z původního obrázku

31	38	51	83	110	96	80	92
91	102	105	116	132	119	98	98
96	123	130	132	148	143	116	101
136	154	140	118	130	142	131	121
207	211	183	152	149	145	129	122
151	181	214	240	237	172	93	59
93	122	183	251	250	149	47	16
157	133	136	173	164	79	23	39

blok po kvantizaci DCT koeficientů a následné dekompresi

(dekomprese = dekvantizace, tj. vynásobení kvantizačními koeficienty, a IDCT)

JPEG

KÓDOVÁNÍ AC KOEFICIENTŮ

- tj. kódování koeficientů vyjma koeficientu na pozici [0,0]
- kódování pomocí sekvence bitů <zzzz><kkkk><bb…b>
  - kkkk – 4bitové číslo udávající kategorii koeficientu (a tím pádem počet bitů b)
    1 = {–1, 1} 2 = {–3, –2, 2, 3} 3 = {–7…–4, 4…7}
    4 = {–15…–8, 8…15} 5 = {–31…–16, 16…31} …
  - zzzz – počet nulových koeficientů před kódovaným
  - b – bity pro přesnou hodnotu koeficientu
- speciální kódy:
  - <zzzz><kkkk> = 1111 0000 ⇒ 16 nul za sebou
  - <zzzz><kkkk> = 0000 0000 ⇒ do konce sekvence následují jen nuly (tzv. kód EOB = end of block)

PŘÍKLAD KÓDOVÁNÍ AC KOEFICIENTŮ

- zig-zag průchod:
  10, −12, −10, −10, −11,
  1, 5, −1, 2, −2, 0, −2, 0, 1,
  −1, −1, 0, −1, 0, −1, 0, −1,
  2, 0, 1, 0, 0, 0, 0, 0, 0, 0,
  −1, 0, 0, …, 0

128	10	-11	1	1	-1	0	0
-12	-10	5	0	-1	0	0	0
-10	-1	-2	0	1	0	0	0
2	0	-1	0	0	0	0	0
-2	0	2	0	0	0	0	0
-1	-1	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

kvantizované koeficienty DCT

- kódy některých AC koeficientů
  10 → 0000 0100 0010 (kategorie 4, 10 − 8 = 2)
  −12 → 0000 0100 1011 (kategorie 4, −12 + 8 + 15 = 11)
  …
  (převod na dvojkový doplněk)
  5 → 0000 0011 001 (kategorie 3, 5 − 4 = 1)
  …
  (ofset pro kategorii 3)
  −2 → 0001 0010 11 (kategorie 2, −2 + 2 + 3 = 3)

POSLEDNÍ KROKY KÓDOVÁNÍ

- v kódech AC koeficientů se dá předpokládat, že sekvence <zzzz><kkkk> se budou vyskytovat s různou pravděpodobností ⇒ nezapisují se na výstup přímo, ale ještě se na ně aplikuje Huffmanovo nebo aritmetické kódování
- DC koeficienty (koeficienty na pozici [0,0] v DCT každého bloku) představují průměrnou hodnotu bloku
  ⇒ dá se očekávat, že u sousedních bloků budou podobné
  ⇒ kódují se technikou DPCM (tj. ukládá se jen rozdíl oproti předchozímu)

JPEG

- komprese JPEG je z principu ztrátová – pro bezztrátový běh by bylo nutné uložit koeficienty DCT přesně
- existuje bezztrátová varianta JPEG, která nepoužívá DCT, ale predikční schéma; v praxi se příliš nepoužívá
- existuje i schéma určené pro obrázky s 1 bitem/pixel
  - někdy označováno JBIG (Joint Bi-level Image experts Group)
  - založeno na aritmetickém kódování
  - v praxi se často využívá pro kompresi proudu bitů v raw formátech

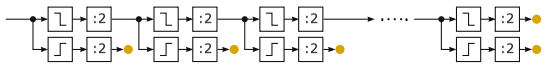
DWT

DISKRÉTNÍ WAVELETOVÁ TRANSFORMACE

- zásadní nevýhoda DCT a obdobných transformací – nutnost volit velikost bloku (v JPEG 8 × 8)
  - příliš velký blok – kvantizace DCT koeficientů vede ke globálnímu poškození celého bloku ⇒ je-li blok jen lokálně členitý, dotkne se kvantizace i jeho hladkých částí
  - příliš malý blok – neefektivní dekorelace v hladkých plochách obrazu ⇒ špatný kompresní poměr
- možnosti nápravy
  - kodér si velikost bloku odhaduje podle charakteru obrazu
  - kodér založen na diskrétní waveletové transformaci, která dělení na bloky nevyžaduje

DWT

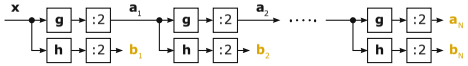
- vychází ze spojité waveletové transformace, pro kompresi obrazu se používá její speciální formulace – nebudeme se spojitou variantou vůbec zabývat
- princip podobný hierarchickému sub-band kódování
  - rozdělení signálu na vysokofrekvenční a nízkofrekvenční část
  - podvzorkování obou částí na polovinu vzorků
  - aplikace téhož postupu na nízkofrekvenční část (v hierarchickém sub-band kódování se aplikuje i na vysokofrekvenční část)
  - koeficienty DWT = souhrn koncových výstupů



DWT

DOPŘEDNÁ DWT

- vstup: signál **x**[]
- aplikace FIR filtru **g** (dolní propust) na **x**[]
  podvzorkování = vynechání lichých vzorků (číslují se od 0)
  výsledek: signál $\mathbf{a}_1$[]
- aplikace FIR filtru **h** (horní propust) na **x**[]
  podvzorkování
  výsledek: signál $\mathbf{b}_1$[]
- aplikace téhož postupu na $\mathbf{a}_1$[] ⇒ výstup $\mathbf{a}_2$[], $\mathbf{b}_2$[]; atd.
- výstup DWT: signály $\mathbf{b}_1$[], $\mathbf{b}_2$[], …, $\mathbf{b}_N$[] a signál $\mathbf{a}_N$[]



DWT

ZPĚTNÁ DWT

- opačný postup k dopředné DWT
- vstup: signály $\mathbf{a}_k[]$, $\mathbf{b}_k[]$
- zdvojnásobit počet vzorků signálu $\mathbf{a}_k[]$ vložením nul:
  $\check{\mathbf{a}}_k[2i] = \mathbf{a}^k[i]$, $\check{\mathbf{a}}_k[2i+1] = 0$
- zdvojnásobit počet vzorků signálu $\mathbf{b}_k[]$ vložením nul
- aplikace FIR filtru $\tilde{\mathbf{g}}$ na $\check{\mathbf{a}}_k[] \Rightarrow$ signál $\mathbf{A}_k[]$
  aplikace FIR filtru $\tilde{\mathbf{h}}$ na $\check{\mathbf{b}}_k[i] \Rightarrow$ signál $\mathbf{B}_k[]$
- vytvoření signálu $\mathbf{a}_{k-1}[]$:
  $\mathbf{a}^{k-1}[i] = \mathbf{A}^k[i] + \mathbf{B}^k[i]$
- pokračování postupu na $\mathbf{a}_{k-1}[]$, $\mathbf{b}_{k-1}[]$
- koncový výsledek $\mathbf{a}_0[]$ je roven původnímu signálu $\mathbf{x}[]$

DWT

PŘÍKLAD

$\mathbf{g} = [\underline{½}; ½]$ $\mathbf{h} = [\underline{½}; -½]$ $\tilde{\mathbf{g}} = [1; \underline{1}]$ $\tilde{\mathbf{h}} = [-1; \underline{1}]$

poloha podtržení značí, z kterých vzorků vstupního signálu se má vytvořit vzorek výstupního signálu

	signál **x**	0	16	8	32	8	4	4	8
			aplikace **g**						
jeden krok dopředné DWT	**x** ⊗ **g**	8	12	20	20	6	4	6	4
	$\mathbf{a}_1$	8	20	6	6				
	x ⊗ **h**	−8	4	−12	12	2	0	−2	4
	$\mathbf{b}_1$	−8	−12	2	−2				
jeden krok zpětné DWT	$\check{\mathbf{a}}_1$	8	0	20	0	6	0	6	0
			aplikace $\tilde{\mathbf{g}}$						
	$\mathbf{A}_1$	8	8	20	20	6	6	6	6
	$\check{\mathbf{b}}_1$	−8	0	−12	0	2	0	−2	0
	$\mathbf{B}_1$	−8	8	−12	12	2	−2	−2	2
	$\mathbf{A}_1 + \mathbf{B}_1$	0	16	8	32	8	4	4	8

výsledek zpětné transformace: rekonstrukce signálu **x**

DWT

PŘÍKLADY FILTRŮ

- filtrům **g**, **h** se říká analyzační, filtrům $\tilde{\mathbf{g}}$, $\tilde{\mathbf{h}}$ syntetizační
- Haarovy ortonormální filtry:
  $\mathbf{g} = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$ $\mathbf{h} = [\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}]$ $\tilde{\mathbf{g}} = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$ $\tilde{\mathbf{h}} = [\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$
- Daubechies-4 ortonormální filtry:
  g = [0,483; 0,837; 0,224; –0,129]
  h = [–0,129; –0,224; 0,837; –0,483]
  $\tilde{\mathbf{g}}$ = [–0,129; 0,224; 0,837; 0,483]
  $\tilde{\mathbf{h}}$ = [–0,483; 0,837; –0,224; –0,129]
- ortogonální/ortonormální filtry: koeficienty filtrů stejné, liší se znaménky a pořadím
- biortogonální filtry – liší se koeficienty nebo i délkou

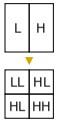
DWT

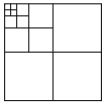
- podobně jako v teorii FIR filtrů platí:
  filtr dlouhý
  ⇒ může lépe oddělit vysoké a nízké frekvence (to je dobře)
  ⇒ hůře lokalizuje detaily v signálu (to je spíš špatně)
- délka filtrů tedy plyne z kompromisu
- výhoda DWT: k signálu lze vyhledat vhodný tvar a délku filtrů (naproti tomu u DFT, DCT apod. jsou bázové funkce sinus a kosinus pevně dané)

DWT

DVOJROZMĚRNÁ DWT

- jeden krok dopředné DWT na všechny řádky, uložení výsledných nízko- (L) a vysokofrekvenčních (H) signálů za sebe
- jeden krok dopředné DWT na všechny sloupce výsledku, uložení výsledných nízko- a vysokofrekvenčních signálů pod sebe
  - sloupce vzniklé z části L: LL, LH
  - sloupce vzniklé z části H: HL, HH
- aplikace stejného postupu na 2D signál LL
- algoritmická složitost: O(*N*), kde *N* je délka řádky, resp. sloupce
  (pozn.: složitost DCT je O(*N* log *N*))





výsledek
4 kroků 2D DWT

DWT

původní obrázek

2D DWT, Haarova báze

Aplikace 8 kroků DWT na vstupní obrázek 256 × 256 pixelů. Koeficienty HL pásem ukazují, kde měl původní obrázek výrazné svislé hrany (H signál po DWT řádky výrazný), koeficienty LH pásem ukazují, kde měl výrazné vodorovné hrany (H signál po DWT sloupců výrazný).

DWT

2D DWT, báze Daubechies4

2D DWT, Haarova báze

Rozdíl mezi různými typy analyzačních filtrů. Zřetelně je vidět, že Daubechies4 filtry signál dekorelují lépe – v rozkladu Haarovými filtry je v pásmech HL, LH stále vidět podoba původního obrázku (tj. dekorelace není moc dobrá).

DWT KOMPRESE

- komprese obrazu pomocí DWT založena na podobném principu jako komprese pomocí DCT:
  - transformace obrazu
  - kvantizace koeficientů
  - kódované uložení kvantizovaných koeficientů
- varianta 1: pro všechny koeficienty daného frekvenčního pásma kvantizace stejná
  - snadná implementace
  - koeficienty v daném frekvenčním pásmu mají velké i malé hodnoty ⇒ neexistuje univerzální ideální nastavení kvantizace

  ⇒ nepříliš kvalitní výsledek, resp. kompresní poměr

DWT KOMPRESE

Haarova báze, 1/64 koeficientů kvantizována s velkou přesností, zbytek s „přesností" 0 bitů (tj. koeficient byl vyhozen, indikace šedou barvou)

Haarova báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty v pásmech nejnižších frekvencí)

DWT KOMPRESE

- varianta 2: největší koeficienty kódovány největším počtem bitů
  - vede k výrazně lepším výsledkům rekonstrukce, pokud koeficienty zaberou stejné množství bitů jako u varianty 1
  - problém: je třeba uložit informaci, kolik bitů každý koeficient zabírá ⇒ zhoršení kompresního poměru
  - ukazuje se, že uložení informace o počtu bitů každého koeficientu zabírá víc místa než samotné koeficienty

DWT KOMPRESE

Haarova báze, 1/64 největších koeficientů kvantizována s velkou přesností, zbytek s „přesností" 0 bitů (tj. koeficient byl vyhozen, indikace šedou barvou)

Haarova báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty s největší absolutní hodnotou)

DWT KOMPRESE

Haarova báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty v pásmech nejnižších frekvencí, viz snímek 49)

Haarova báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty s největší absolutní hodnotou, viz snímek 51)

DWT KOMPRESE

Daubechies-4 báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty v pásmech nejnižších frekvencí)

Daubechies-4 báze, rekonstrukce z 1/64 celkového množství původních koeficientů (zachovány koeficienty s největší absolutní hodnotou)

DWT KOMPRESE

Haarova báze, rekonstrukce z 1/16 celkového množství původních koeficientů (zachovány koeficienty s největší absolutní hodnotou)

Daubechies-4 báze, rekonstrukce z 1/16 celkového množství původních koeficientů (zachovány koeficienty s největší absolutní hodnotou)

EZW KOMPRESE

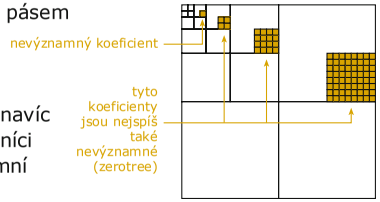
EMBEDDED ZEROTREE WAVELET

- vtipné řešení problému ukládání „mapy koeficientů“, tj. bitové délky jednotlivých koeficientů
  - žádná mapa se neukládá (jistá podobnost: slovníkové komprese typu LZW také žádný slovník neukládají)
  - koeficienty se neukládají jeden za druhým, ale „všechny najednou“
- první kompresní schéma založené na DWT, které může konkurovat technikám založeným na DCT
- principy EZW jsou dále rozvinuty ve schématech SPIHT nebo JPEG2000
  - EZW je nejjednodušší a pro výklad principu asi nejvýhodnější

EZW KOMPRESE

PRINCIP

- dominantní průchod: uložení informace, které koeficienty jsou „významné“ (absolutní hodnota větší než práh T)
  - zásadní trik: je-li koeficient „nevýznamný“, pak jsou nejspíš „nevýznamné“ koeficienty v příslušných částech vysokofrekvenčních pásem
  - kód koeficientu K:
    POS pro $K > T$
    NEG pro $-K > T$
    ZTR pro $|K| \leq T$ a navíc všichni následníci jsou nevýznamní (zerotree)
    IZ v ostatních případech (isolated zero)



EZW KOMPRESE

- vedlejší průchod: uložení nejvýznamnějšího bitu „významných" koeficientů metodou postupných aproximací
- snížení prahu T na polovinu
- další iterace dominantního průchodu: doplní informaci, které koeficienty začaly být významné na nové hodnotě T
- další iterace vedlejšího průchodu: doplní další bit koeficientům, které byly dosud označeny za významné
- snížení prahu T na polovinu atd.

- algoritmus lze kdykoliv ukončit (typicky po dosažení požadované délky souboru) ⇒ dosud vygenerovaný výstup poskytuje nejlepší aproximaci o koeficientech DWT
  – výhoda: přesné řízení kompresního poměru

EZW KOMPRESE

METODA POSTUPNÝCH APROXIMACÍ

- postupné kódování reálného čísla K z rozsahu [0, M)
- princip: chceme-li K kódovat jedním bitem, budeme bit 0 interpretovat jako hodnotu M/4, bit 1 jako hodnotu 3M/4
  – maximální chyba aproximace je M/4
- chceme-li K kódovat 2 bity ⇒ max. chyba aproximace M/8
- příklad: M = 50, K = 42

práh T	výstupní bit	interpretace dosavadního výstupu	maximální chyba dosavadního výstupu (absolutní hodnota)
25,0000	1	37,5000	12,5000
37,5000	1	43,7500	6,2500
43,7500	0	40,6250	3,1250
40,6250	1	42,1875	1,5625

- obecně: číslo $K \geq 0$ reprezentujeme N bity číslem $a = \text{floor}(K / M \times 2^N)$, interpretujeme číslem $K' = (a + 0{,}5) \times M / 2^N$, max. chyba je $0{,}5 \times M / 2^N$

EZW KOMPRESE

PŘÍKLAD BĚHU

63	-51	10	8	40	6	5	6
18	44	49	20	8	−9	3	−3
6	20	2	5	7	5	−7	−2
15	12	−9	19	−8	2	0	0
8	5	−4	6	2	3	2	1
9	6	1	0	1	1	1	1
−3	8	10	12	0	3	2	1
16	18	6	2	1	2	3	3

DWT koeficienty.
Prochází se jimi postupně přes pásma LL, HL, LH, HH, každé pásmo řádku po řádce.

První běh: $M = 64$, $T = 32$

Dominantní průběh		Vedlejší průběh		
koeficient	výstup	práh	výstup	interpretace
63	*POS*	48	*1*	56
−51	*NEG*	−48	*1*	−56
18	*ZTR*			
44	*POS*	48	*0*	40
10	*IZ*			
8	*ZTR*			
49	*POS*	48	*1*	56
20	*ZTR*			
2	*ZTR*			
5	*ZTR*			
−9	*ZTR*			
19	*ZTR*			
...				

nevýznamný koeficient, podřízené koeficienty rovněž ⇒ mohou se z průchodu vynechat

EZW KOMPRESE

- výstup dominantního průběhu dále kódován aritmetickým kodérem (vstupní abeceda POS, NEG, IZ, ZTR pro většinu pásem, jen pásma s nejvyššími frekvencemi nemohou obsahovat kód IZ, protože neexistují podřízené koeficienty)
- výstup vedlejšího průběhu rovněž kódován aritmetickým kodérem (vstupní abeceda 0, 1)
- každý bit výstupu zlepšuje kvalitu obrazu, kódování/dekódování lze kdykoliv ukončit

- další výhoda: použitím vhodných analyzačních/syntetizačních filtrů a dostatečně dlouhým kódováním lze docílit bezztrátové komprese

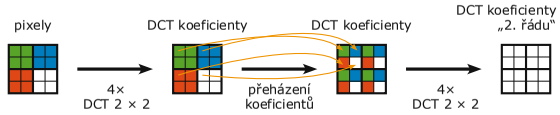
DALŠÍ TECHNIKY V KOMPRESI OBRAZU

- současné trendy:
  - využívání algoritmů kódování tzv. I-snímků ve videu (např. z kodéru H.264, VP-9 apod.)
  - návrh kódování/dekódování uzpůsoben paralelní a/nebo hardwarové implemenzaci
  - maximální snaha využívat celočíselnou aritmetiku
- užití jiného barevného prostoru než $Y`C_BC_R$
  - $Y`C_GC_O$ má podobné vlastnosti, výpočet v celočíselné aritmetice:
    $Y` = [G + (R + B) / 2] / 2$
    $C_G = [G – (R + B) / 2] / 2$, $C_O = (R – B) / 2$

- používání transformací s obdobnými vlastnostmi jako DCT, ale s výpočty s celými čísly
- bloky pro DCT jiné než 8 × 8 pixelů
  - obrázek typicky rozdělen na velké bloky (např. 32×32 px)
  - kodér se automaticky rozhodne pro „velkou DCT", nebo velký blok rozdělí na menší bloky
  - hierarchické dělení bloků až do velikosti 4×4 px)

DALŠÍ TECHNIKY

- hierarchické užití DCT
  - stejnolehlé koeficienty DCT z několika sousedních bloků jsou typicky podobné
  ⇒ sdruží se do bloků a ty se opět transformují DCT



Ilustrace principu:
1. vstupní obrázek rozdělen na bloky 2 × 2 px
2. bloky transformovány DCT velikosti 2 × 2
3. koeficienty DCT [0,0] sdruženy do bloku 2 × 2; obdobně koeficienty [0, 1] atd.
4. nově vytvořené bloky opět transformovány DCT velikosti 2 × 2, výsledek kvantizován a odeslán na výstup

DALŠÍ TECHNIKY

- predikce bloku z okolních bloků
  - před kódováním bloku se kodér rozhodne, zda lze jeho obsah odhadnout z pixelů bloků „nad" a „vlevo"
  - pokud ano, odečte odhad od hodnot bloku a kóduje rozdíl (reziduum), na výstup odešle i informaci o typu odhadu
  - pokud ne, kóduje pixely standardním způsobem

	2	4	6	8
2	2	4	6	9
3	2	5	6	8
2	3	4	7	7
2	2	5	6	9

kódovaný blok a jeho okolí

2	4	6	8
2	4	6	8
2	4	6	8
2	4	6	8

kodér „si všiml", že řádky bloku jsou podobné jeho hornímu okolí – toto je „odhad bloku"

0	0	0	1
0	1	0	0
1	0	1	-1
0	1	0	1

kodér bude kódovat jen rozdíl bloku a odhadu