Komprese

Pokročilá analýza obrazu

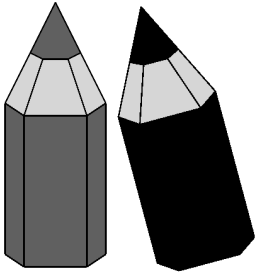
Mgr. Markéta Trnečková, Ph.D.

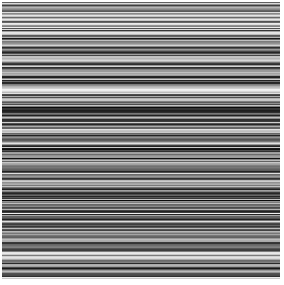
Typy obrazů

- *binární (B/W) – 1 bit/pixel*
- v odstínech šedi (gray scale) – 1 byte/pixel
- *indexový (pseudo color) – 1 byte/pixel*
- indexový (direct color) – 3 byte/pixel
- *plně barevný (color) – 3-4 složky"*
  - *low color (*15 *bit)*
  - *high color (*16 *bit)*
  - *true color (*24 *bit)*
  - *super true color (*32 *bit)*
  - *deep color (*48 *bit)*

Redundance

- *Redundance kódování – informaci kódujeme více bity, než je potřeba*
- Redundance prostorová – korelace mezi pixely
- *Nerelevantní informace – informace, kterou lidské oko nedokáže zpracovat*



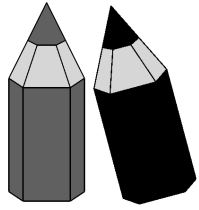


Redundance

- **Redundantní data** – reprezentace obsahuje opakující se nebo nerelevantní informaci
- **Kód nesoucí informaci** – b, b'
- *Relativní redundance dat*
  $R = 1 - \frac{1}{C}$
- Kompresní poměr
  $C = \frac{b}{b'}$

Redundance kódování

- **Délka** – počet bitů každé informace
- 8-**bitový kód** – každá barva je kódována 8 bity
- Pro obrázek obsahující 4 barvy je 8-bitový kód zbytečný
- **Fixní** *m*-**bitový kód** – každý kus informace kódován *m* bity



Příklad

Jaký je kompresní poměr a relativní redundance kódování, pokud obrázek zakódujeme místo 8-bitovým kódem pouze 2-bitovým kódem?

Redundance kódování

- Fixní m-bitový kód není vždy optimální
- **Variabilní délka kódu** – Huffmanovo kódování
- $r_k \in [0, L-1]$ – intenzity v obraze
- velikost obrazu: $M \times N$
- n_k – počet výskytů intenzity r_k
- $P(r_k) = \frac{n_k}{M \cdot N}$ – pravděpodobnost výskytu intenzity
- $L(r_k)$ – počet bitů potřebných k reprezentaci hodnoty r_k
- **Průměrný počet bitů potřebných k reprezentaci každého pixelu** – $L_{avg} = \sum_{k=0}^{L-1} L(r_k)P(r_k)$
- **Celkový počet bitů potřebných k reprezentaci každého pixelu** – $M \cdot N \cdot L_{avg}$

Příklad

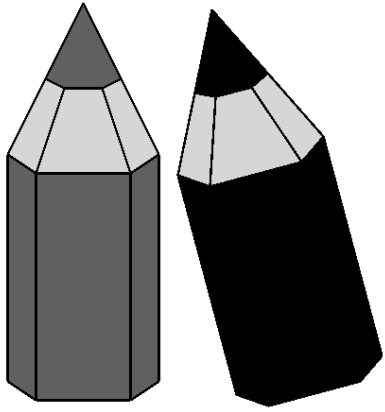
Jaký je průměrný počet bitů potřebných k reprezentaci každého pixelu, pokud použijeme fixní m-kód?

Redundance kódování – Příklad

- Velikost: 440×440
- Intenzity: 0, 96, 214 a 255

Fixní 8-kód

intenzita	kód
r_0	00000000
r_{96}	01100000
r_{214}	11010110
r_{255}	11111111



Příklad

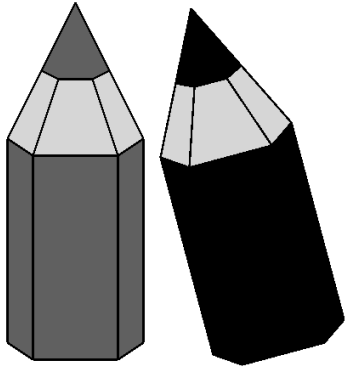
Jak velký fixní m-kód potřebujeme pro zakódování tohoto obrázku? Jak by vypadal?

Redundance kódování – Příklad

Kód s proměnlivou délkou

intenzita	$P(r_k)$	kód	délka kódu
r_0	0.26	01	2
r_{96}	0.2	000	3
r_{214}	0.11	001	3
r_{255}	0.43	1	1

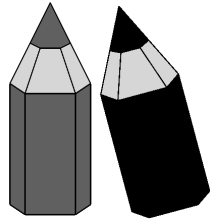
- Průměrná délka: $L_{avg} =?$
- Komprese a relativní redundance 8-kódu:
  $C =?$
  $R =?$



Redundance kódování – Příklad

Kód s proměnlivou délkou

intenzita	$P(r_k)$	kód	délka kódu
r_0	0.26	01	2
r_{96}	0.2	000	3
r_{214}	0.11	001	3
r_{255}	0.43	1	1



- Průměrná délka:
  $L_{avg} = 0.26 \cdot 2 + 0.2 \cdot 3 + 0.11 \cdot 3 + 0.43 \cdot 1 = 1.88$
- Komprese a relativní redundance 8-kódu:
  $C = \frac{440 \cdot 440 \cdot 8}{440 \cdot 440 \cdot 1.88} = \frac{8}{1.88} \approx 4.25$
  $R = 1 - \frac{1}{4.26} \approx 0.76$

Příklad

Jaký je průměrný počet bitů potřebných k reprezentaci každého pixelu, pokud použijeme fixní m-kód?

Huffmanův kód

- Každý znak (barva) je kódován jedním kódem
- Znaky vyskytující se častěji jsou kódovány kratším kódem
- Znaky se seřadí dle pravděpodobnosti výskytu a postupně se shlukují
- Poté se vytváří kód
- Viz příklad
- Kód je jednoznačně dekódovatelný – není možné ho dekódovat jinak
- Dekódování – čteme zleva doprava a hledáme nejdelší možný kód

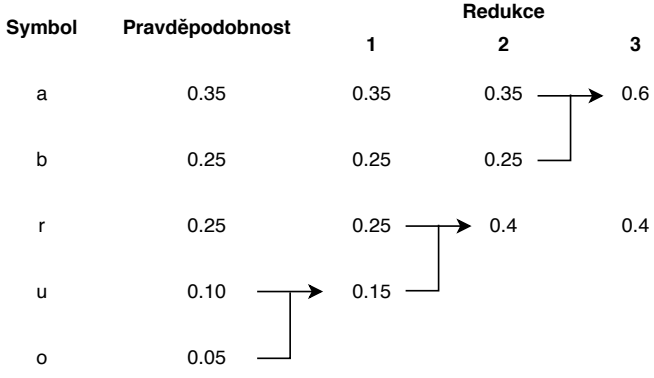
Příklad

Vytvořte Huffmanův kód pro řetězec 'barbaraabarboraubaru'.

Příklad – Huffmanův kód

Vytvořte Huffmanův kód pro řetězec 'barbaraabarboraubaru'.

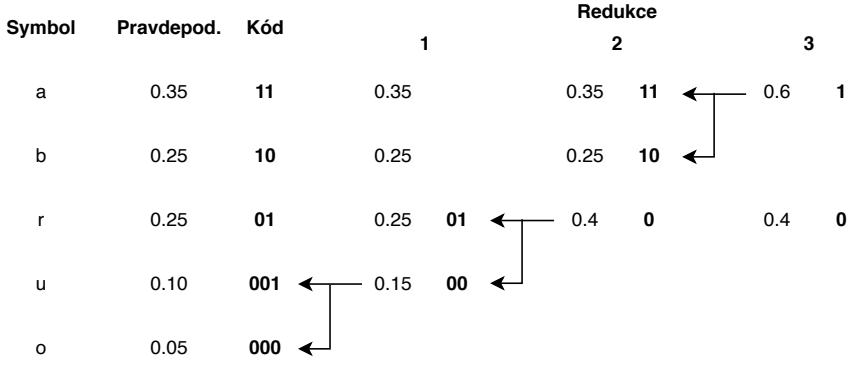
- Postupné shlukování:



Příklad – Huffmanův kód

Vytvořte Huffmanův kód pro řetězec 'barbaraabarboraubaru'.

- Vytvoření kódu:



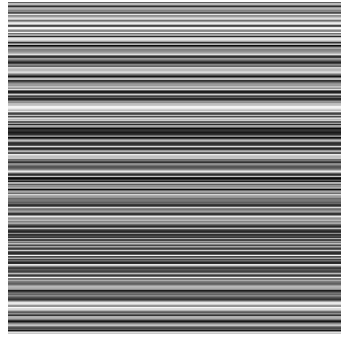
Příklad – Huffmanův kód

Vytvořte Huffmanův kód pro řetězec 'barbaraabarboraubaru'.

- **Kód**:
  - a – '11'
  - b – '10'
  - r – '01'
  - u – '001'
  - o – '000'
- **Výsledný řetězec**: '10110110110111111011011000001110011011010 01'
- **Dekódování**:
  - 1. validní kód zleva – '10' = b
  - 2. – '11' = a
  - 3. – '01' = r

Redundance prostorová

- Velikost: 256×256
- Intenzity: $0, \ldots, 255$
- Kód: Každý řádek intenzita + počet opakování
- Každý pixel kódován 2 byty
- RLE komprese



Příklad

Spočítejte kompresi a relativní redundanci 8-bitového kódu vůči tomuto kódu.

RLE

- **run-length pair** – dvojice počet opakování a intenzita (nejčastěji index do palety)
- 2 bytová reprezentace
- speciální kódy 1. byte = 0:
  - 2. byte 0 = konec řádku
  - 2. byte 1 = konec obrázku

Příklad

Jak bude vypadat zakódovaný řetězec 'aaaabbcccaabb'?

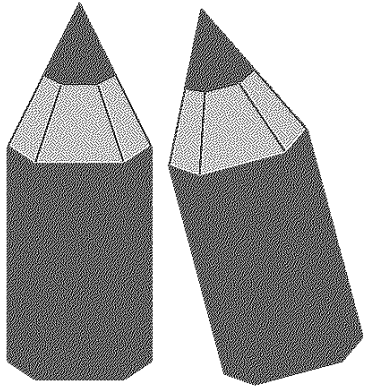
Příklad

Existuje příklad, kdy by RLE dopadla špatně?

Nerelevantní informace

- Velikost: 440×440
- Fixní 8-kód – celková délka: $440 \cdot 440 \cdot 8$
- Zaokrouhlení: 1 intenzita

Nerelevantní informace



JEPG komprese

- Ztrátová kompresní metoda
- **Kroky**:
  - Obrázek převedeme do barevného modelu YCbCr – jednotlivé složky pak zpracováváme zvlášť
  - Podvzorkujeme barevné složky (jasovou ne)
  - Rozdělíme na 8×8 bloky
  - Aplikujeme DCT
  - Provedeme kvantizaci (vydělíme kvantizační tabulkou – různé tabulky pro různé kvality výsledku) a výsledek zaokrouhlíme
  - Bloková data se pomocí zik-zak metody převedou na sekvenci
  - Sekvenci kódujeme pomocí bezztrátové kompresní metody, např. RLE

Příklad – JEPG komprese

Pomocí JPEG komprese zakódujte následující blok

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

Příklad – JEPG komprese

Diskrétní kosinová transformace

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

DCT (zpětná)

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

$c(u), c(v) = \frac{1}{\sqrt{2}}$ pro $u, v = 0$
$= 1$ jinak

1259.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3
-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2
-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1
-7.1	-1.9	0.2	1.5	0.9	-0.1	-0.0	0.3
-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3
1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0
-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8
-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4

Příklad – JEPG komprese

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	78	87	103	121	120	101
72	92	95	98	112	100	103	99

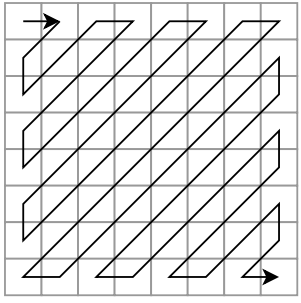
Příklad – JEPG komprese

Po kvantizaci

79	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Příklad – JEPG komprese

Zig zag



79 0 -2 -1 -1 -1 0 0 -1 -1 0 0 0 . . . 0

Příklad – JEPG komprese

Expandované koeficienty před ICDT

1264	0	-10	0	0	0	0	0
-24	-12	0	0	0	0	0	0
-14	-13	0	0	0	0	0	0
-14	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Příklad – JEPG komprese

Rekonstruovaná data

142	144	147	150	152	155	155	155
149	150	153	155	156	157	156	156
157	158	159	161	161	160	159	158
162	162	163	163	162	160	158	157
162	162	162	162	161	158	156	155
160	161	161	161	160	158	156	154
160	160	161	162	161	160	158	157
160	161	163	164	164	163	161	160

JEPG komprese

Rekonstruovaná data

Původní data

Nová data

JEPG komprese

Bez komprese
1286360 byte

Kvalita 80
526923 byte

Kvalita 50
348090 byte

Kvalita 10
154793 byte

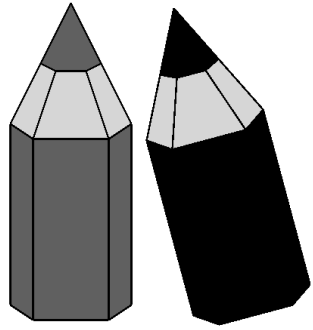
Kvalita 5
86691 byte

Informace

- **Náhodná událost** – E
- **Pravděpodobnost náhodné události** – E
- **Informace** – $I(E) = \log \frac{1}{P(E)} = -\log P(E)$
- **Základ logaritmu** = jednotky (v obraze 2)
- **Entropie** = průměrná informace
  $H = -\sum_{j=1}^{J} P(a_j) \log P(a_j)$
- **Matlab**: `J = entropy(I)`

Redundance kódování – Příklad

intenzita	$P(r_k)$
r_0	0.26
r_{96}	0.2
r_{214}	0.11
r_{255}	0.43

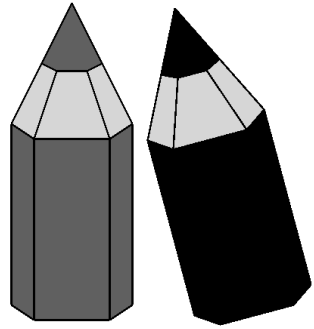


Příklad

Jaká je entropie tohoto obrazu?

Redundance kódování – Příklad

intenzita	$P(r_k)$
r_0	0.26
r_{96}	0.2
r_{214}	0.11
r_{255}	0.43



$$H = -[0.26 \cdot \log_2 0.26 + 0.2 \cdot \log_2 0.2 + 0.11 \cdot \log_2 0.11 + 0.43 \cdot \log_2 0.43] \approx 1.843 \text{ bit/pixel.}$$

Měření kvality komprese

- Objektivní hodnocení
- Subjektivní hodnocení – fidelity kriteria

Objektivní měření

Chybové metriky

- **Hodnota vstupního obrazu**: $f(x, y)$
- **Hodnota výstupního obrazu**: $\hat{f}(x, y)$
- **Chyba**: $e(x, y) = |\hat{f}(x, y) - f(x, y)|$
- **Celková chyba**: $\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} |\hat{f}(x, y) - f(x, y)|$
- **mean-square error**
  $e_{ms} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2$
- **root-mean-square error**
  $e_{rms} = \left[\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2 \right]^{\frac{1}{2}}$
- **mean-square signal-to-noise ratio**
  $SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x,y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x,y) - f(x,y)]^2}$

Objektivní měření
Peak Signal-to-Noise Ratio

- **Peak Signal-to-Noise Ratio** (PSNR) – hodně se používá v kompresi
  $PSNR = 10 \log_{10} \frac{L^2}{e_{ms}}$ kde L je maximální hodnota (běžně 1 nebo 255)
- Penalizuje kvadratickou chybu (silně trestá velké odchylky)
- Nezohledňuje strukturální ani percepční vlastnosti
- Hodnocení:
  - < 20 Silná degradace
  - $20 - 30$ Viditelná degradace
  - $30 - 35$ Přijatelná kvalita
  - $35 - 40$ Vizuálně velmi dobrá
  - > 40 Nepostřehnutelný rozdíl

Objektivní měření

Strukturální metriky – SSIM

- **Structural Similarity Index** (SSIM) – metrika založená na porovnání struktury
- Modeluje tři složky: luminanci, kontrast a strukturu

$$SSIM(x, y) = [l(x, y)]^{\alpha}[c(x, y)]^{\beta}[s(x, y)]^{\gamma}$$

- Praktická forma:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

- Hodnota v intervalu $[-1, 1]$, v praxi $[0, 1]$
- $1 =$ identické obrazy
- Lepší korelace s lidským vnímáním než PSNR

Objektivní měření
SSIM

- Počítá se lokálně v klouzavém okně (např. 11×11)
- Výsledkem je mapa kvality + průměrná hodnota
- Méně citlivý na globální změnu jasu
- Citlivější na strukturální degradace (blur, blocking)
- Hodnocení:
  - < 0.5 výrazné strukturální poškození
  - $0.5 - 0.8$ zřetelná degradace
  - $0.8 - 0.95$ dobrá kvalita
  - > 0.95 velmi vysoká kvalita
- Často se používá spolu s PSNR

Objektivní měření
SSIM

Původní obraz

Upravený

Mapa

Objektivní měření
SSIM

- **Multi-Scale SSIM** (MS-SSIM)
  - Vyhodnocuje strukturu na více měřítkách (Gaussian pyramid)
  - Lepší model lidského vnímání detailů a kontrastu
- **Feature Similarity Index** (FSIM)
  - Vychází z významu hran a fázové konzistence
  - Používá Phase Congruency a Gradient magnitude
  - Lépe zachycuje perceptuálně důležité struktury

Objektivní měření

Percepční a hluboké metriky

- **Percepční metriky** – snaží se modelovat lidské vizuální vnímání
- Nepracují přímo v pixelovém prostoru, ale ve **feature prostoru**
- Využívají hluboké neuronové sítě (CNN)
- Porovnávají aktivace v jednotlivých vrstvách sítě
- Obecný princip:

$$d(x, y) = \|\phi(x) - \phi(y)\|$$

  kde $\phi(\cdot)$ je zobrazení do feature prostoru
- Lepší korelace s lidským hodnocením než PSNR/SSIM

Objektivní měření

Percepční a hluboké metriky

- **Learned Perceptual Image Patch Similarity** (LPIPS)
  - Využívá předtrénované CNN (např. VGG, AlexNet)
  - Vážená vzdálenost mezi normalizovanými feature mapami
  - Nižší hodnota = vyšší perceptuální podobnost
  - Citlivá na texturu a strukturální detaily
- **Fréchet Inception Distance** (FID)
  - Porovnává distribuce feature vektorů dvou množin obrazů
  - Feature extrahovány z Inception sítě
  - Měří vzdálenost mezi dvěma Gaussovskými rozděleními

$$FID = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2})$$

  - Nižší hodnota = lepší shoda distribucí
  - Používá se pro hodnocení generativních modelů

Objektivní měření

No-Reference (NR) – Blind IQA

- **No-Reference Image Quality Assessment** (NR-IQA)
- Nevyžaduje referenční obraz
- Hodnotí kvalitu pouze z degradovaného obrazu
- Klíčové pro:
  - reálné snímky bez ground-truth
  - medicínské a bezpečnostní aplikace
  - monitoring přenosu obrazu
- Obtížnější úloha než FR metriky

Objektivní měření

NR – statistické přístupy

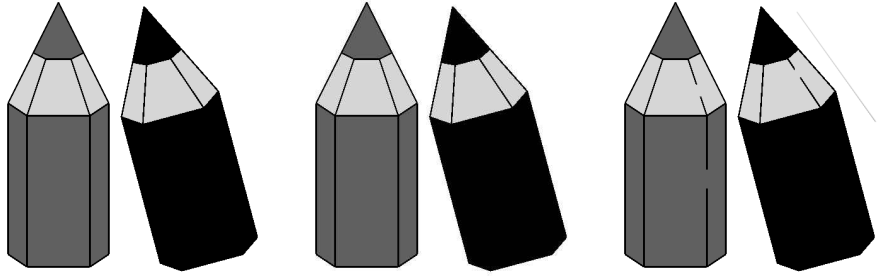
- Vycházejí z **Natural Scene Statistics** (NSS)
- Přirozené obrazy mají specifické statistické vlastnosti
- Degradace tyto statistiky narušuje
  - šum rozšíří rozdělení,
  - rozmazání sníží energii vysokých frekvencí a mění statistiky gradientu
  - JPEG zavádí periodické a blokové artefakty a tím dojde ke změně lokálních statistik
- Příklady:
  - BRISQUE
  - NIQE
  - BLIINDS
- Výstup: predikce subjektivního skóre

Objektivní měření
NR – hluboké modely

- CNN trénované na databázích se subjektivním hodnocením lidí
- Učí se přímo mapování obraz $\rightarrow$ kvalita
- Příklady:
  - NIMA
  - CNN-based BIQA modely
- Výhody:
  - lepší generalizace na komplexní degradace
  - dobrá korelace s lidským hodnocením
- Nevýhoda: závislost na trénovacích datech

Subjektivní hodnocení

- **MOS** (Mean opinion score) – průměrné hodnocení skupiny uživatelů
- **Pairwise comparison** – porovnávání dvou obrázků



Watermarking

- **Watermarking** (vodoznak) – vkládání skryté informace do obrazu
- Cíl:
  - ochrana autorských práv
  - autentizace obsahu (že nebyl obraz změněn)
  - sledování distribuce
- Požadavky:
  - nejde jednoduše odstranit
  - robustnost vůči degradacím
  - dostatečná kapacita
  - někdy neviditelnost (imperceptibility)
- Kompromis mezi robustností a neviditelností
- Vkládání vodoznaku má společné rysy s kompresí

Watermarking

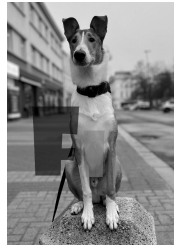
Viditelné vodoznaky

- neprůhledné nebo průhledné
- obrázek, který je přidán k obrazu

Watermarking
Viditelné vodoznaky


Původní


Vodoznak


S vodoznakem


Rozdíl

Příklad

Jak přidáme do obrázku průhledný vodoznak?

Watermarking
Neviditelné vodoznaky

- Nejsou vidět pouhým okem
- Je možné je však získat vhodným dekódovacím algoritmem
- Vodoznak je redundantní informace v obraze, kterou lidský vizuální systém ignoruje

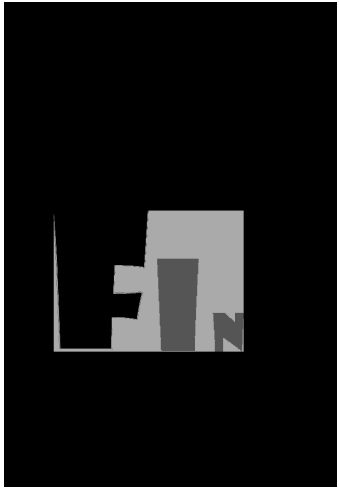
Watermarking

Neviditelné vodoznaky

Příklad: Vodoznak je uložen do dvou nejméně významných bitů

Původní

S vodoznakem



2 nejméně významné bity

Watermarking
Neviditelné vodoznaky

- Důležitou vlastností neviditelných vodoznaků je zabránění jejich odstranění (náhodně nebo úmyslně)
- **Křehké vodoznaky** = jsou odstraněny jakoukoliv modifikací obrázku
- Poznámka: U vodoznaků sloužících k autentifikaci to je žádoucí jev

Příklad

Je použití dvou nejméně významných bitů pro vložení vodoznaku křehké?

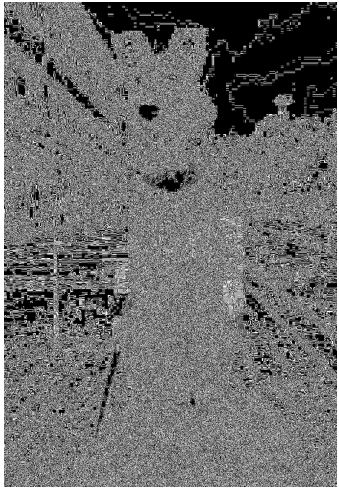
Watermarking

Neviditelné vodoznaky

Příklad: Vodoznak je uložen do dvou nejméně významných bitů je křehký. Níže je aplikovaná jpeg komprese.

Původní

S vodoznakem



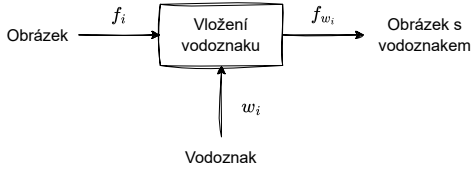
2 nejméně významné bity

Watermarking
Neviditelné vodoznaky

- **Robustní vodoznaky** = takové, které modifikací obrázku neodstraníme
- Běžně jsou odolné vůči:
  - ztrátové kompresi
  - lineárnímu i nelineárnímu filtrování
  - ořezávání, otáčení, převzorkování
  - vytištění a naskenování
- Samozřejmě chceme, aby nebyl původní obrázek modifikovaný viditelně
- Vodoznaky vkládáme buď v prostorové doméně, ale i ve frekvenční

Watermarking
Neviditelné vodoznaky

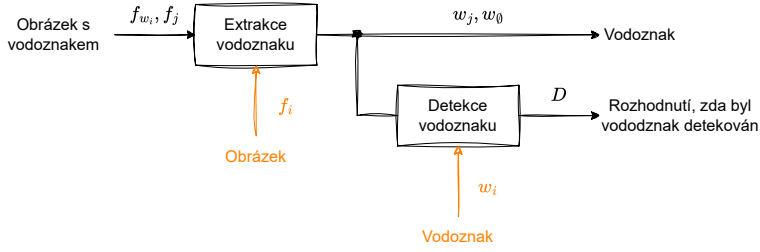
- **Kódování**:
  - Vložíme vodoznak w_i do původního obrázku f_i
  - Dostaneme obrázek s vodoznakem f_{w_i}



Watermarking

Neviditelné vodoznaky

- **Dekódování**:
  - Extrahujeme vodoznak buď z obrázku s vodoznakem f_{w_i} nebo bez vodoznaku f_j
  - Buď pouze detekujeme přítomnost nějakého vodoznaku (k tomu můžeme ale nemusíme použít původní obraz)
  - Případně validujeme vodoznak (k tomu můžeme potřebovat původní vodoznak)
- Pokud použijeme f_i nebo w_i nazýváme tento dekódovací systém **private** nebo **restricted-key**
- Jinak nazýváme tento dekódovací systém **public** nebo **unrestricted-key**



Watermarking

DCT neviditelný robustní vodoznak

1. Spočítáme 2D DCT původního obrázku
2. Najdeme K největších koeficientů $c_1, c_2, \ldots, c_K$
3. Vytvoříme vodoznak obsahující K prvkovou pseudonáhodnou sekvenci čísel $\omega_1, \omega_2, \ldots, \omega_K$ (Gaussovské rozložení, s průměrem $\mu = 0$ a variancí $\sigma^2 = 1$)
4. Vložíme vodoznak (z kroku 3) do koeficientů (z kroku 2)
   $c_i' = c_i \cdot (1 + \alpha\omega_i)$
   α je kladná nenulová konstanta (definuje, jak moc jsou koeficienty modifikované)
   Nahradíme původní koeficienty novými
5. Spočítáme inverzní DCT

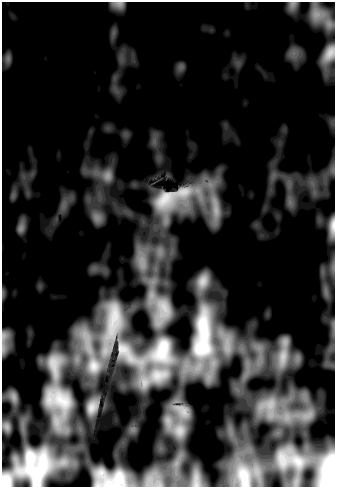
Watermarking

Neviditelné vodoznaky

Příklad: $K = 1000$, $\alpha = 0.1$

Původní

S vodoznakem



Rozdíl (max = 20)

Watermarking

Neviditelné vodoznaky

Původní

$\alpha = 0.05$

$\alpha = 0.1$

$\alpha = 0.5$

Watermarking

DCT neviditelný robustní vodoznak

- **Detekce přítomnosti vodoznaku**

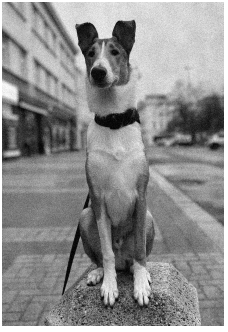
1. Spočítáme 2D DCT obrázku
2. Najdeme K největších koeficientů $\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_K$
   Pokud se jedná o nemodifikovaný obrázek s vodoznakem $\hat{c}_i = c'_i$
   Pokud došlo k modifikaci obrázku $\hat{c}_i \approx c'_i$
   Pokud vodoznak neobsahuje, nebudou si podobné
3. Spočítáme podobnost $\hat{\omega}_1, \hat{\omega}_2, \ldots, \hat{\omega}_K$ s $\omega_1, \omega_2, \ldots, \omega_K$
   $$\gamma = \frac{\sum_{i=1}^{K} (\hat{\omega}_i - \overline{\hat{\omega}})(\omega_i - \overline{\omega})}{\sqrt{\sum_{i=1}^{K} (\hat{\omega}_i - \overline{\hat{\omega}})^2 (\omega_i - \overline{\omega})^2}}$$
   $\overline{\omega}$ a $\overline{\hat{\omega}}$ jsou průměrné hodnoty
4. Porovnáme γ se zvoleným prahem T a dle toho rozhodneme
5. Spočítáme inverzní DCT

Watermarking
Neviditelné vodoznaky

Původní
$\gamma = 9958$

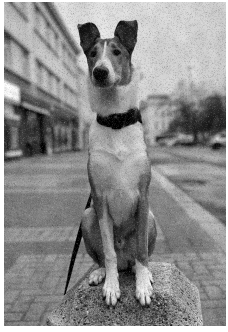
jpeg komprese
$\gamma = 9947$

jpeg komprese (větší)
$\gamma = 9099$



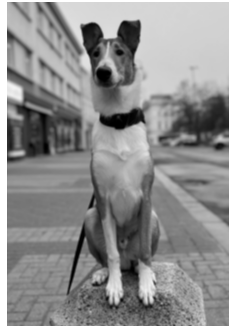
Gauss šum
$\gamma = 9008$

Watermarking
Neviditelné vodoznaky



Šum sůl a pepř
$\gamma = 0.8971$

Bez vodoznaku
$\gamma = NaN$



Rozostření
$\gamma = 0.0048$