

Počítačová grafika

Reprezentace rastrového obrazu

Eduard Bartl



KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLMOUCI



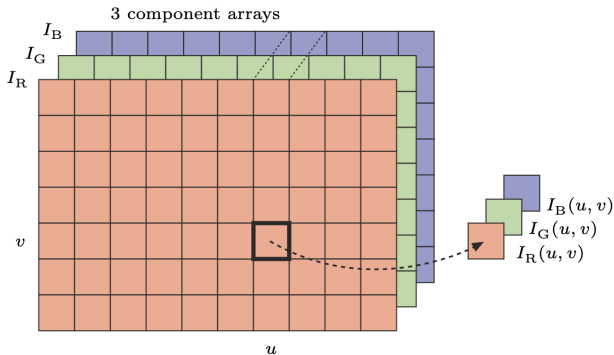
- dvourozměrné pole
- paleta barev (indexový mód)
- kvadrantový strom

Reprezentace pomocí dvourozměrného pole

- nejjednodušší reprezentace s velkou paměťovou náročností
- informace o barvě každého pixelu je uložena v dvourozměrném poli:
 - component ordering
 - packed ordering

Reprezentace pomocí dvourozměrného pole – component ordering

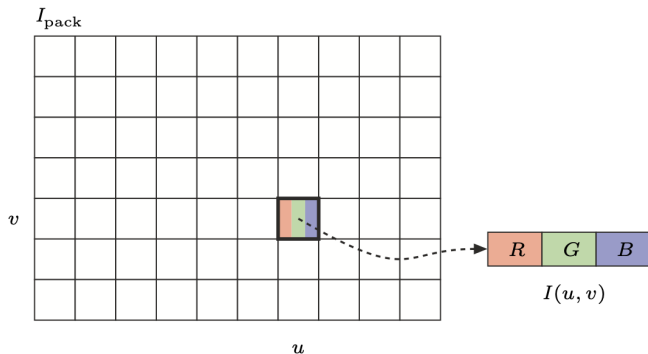
- každý barevný kanál je uložen v samostatném poli



Obrázek: zdroj [2].

Reprezentace pomocí dvourozměrného pole – packed ordering

- používá se pouze jedno pole
- v každé buňce kompletní informace o barvě (obvyčejně 32bitová hodnota typu integer, alfa kanál a 3 barevné kanály)



Obrázek: zdroj [2].

Reprezentace pomocí dvourozměrného pole

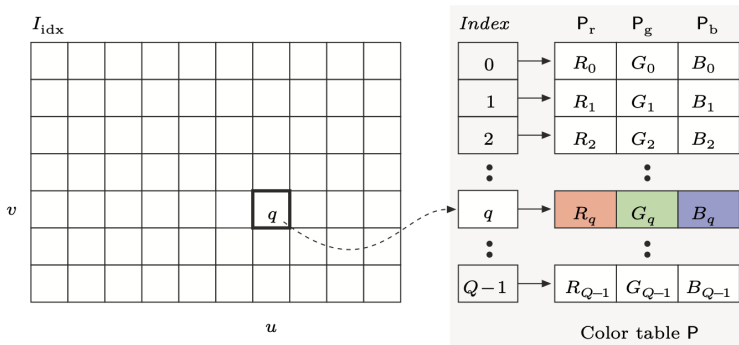
- parametry obrazu:
 - rozlišení
 - barevná hloubka

Barevná hloubka

- **barevná hloubka** – počet bitů potřebný pro vyjádření barvy každého pixelu (jednotka bpp = bit per pixel)
- často používané barevné hloubky:
 - 1 bpp ... $2^1 = 2$ barvy (monochromatický systém), typicky černá a bílá, použití: fax
 - 4 bpp ... $2^4 = 16$ barev; použití: cartoons, line drawings
 - 6 bpp ... $2^6 = 64$ barev; použití: transreflexní displeje a displeje s nízkou spotřebou (např. MIP displeje v chytrých hodinkách)
 - 8 bpp ... $2^8 = 256$ barev; použití: displeje s nízkou spotřebou (např. GPS navigace, měřicí přístroje, některé automobilové displeje)
 - 12 bpp ... $2^{12} = 4096$ barev; použití: šedotónové obrázky v medicíně
 - 16 bpp ... $2^{16} = 65\,535$ barev (high colors), typicky 3 barevné kanály (5-6-5 bitů pro jednotlivé kanály)
 - 24 bpp ... $2^{24} = 16\,777\,216$ barev (true colors), typicky 3 barevné kanály (8-8-8 bitů pro jednotlivé kanály)
- velká paměťová náročnost:
 - běžný obrázek, rozlišení 1024×768 pixelů, true colors:
 $1024 \cdot 768 \cdot 24 = 18\,874\,368 \text{ b} = 2,25 \text{ MB}$

Reprezentace pomocí palety barev

- také se používá dvourozměrné pole, ale hodnota v poli nereprezentuje barvu, nýbrž je ukazatelem do **palety barev** (colormap)
- v paletě jsou uloženy hodnoty všech tří barevných kanálů
- paleta bývá často uložena spolu s obrazem, typická velikost palety je 1B



Obrázek: zdroj [2].

Uvažujme konkrétní scénář:

- na vstupu 24bpp RGB obrázek
- na výstupu 8bpp obrázek (každá hodnota je ukazatelem do palety, která obsahuje $2^8 = 256$ barev)

Problém:

- 1 jak vybrat vhodné barvy do palety – kvantizace barev
- 2 jak snížit počet barev vstupního obrazu na 256 – omezení barevného prostoru

Dva přístupy:

Pevně zadaná paleta

- dopředu neznáme obraz, který budeme převádět do paletové reprezentace
- obecně poskytuje horší výsledky
- rychlejší přístup, převedení je možné provádět v době načítání obrazu
- nejjednodušší řešení: převedení na šedotónový obrázek $\langle r, g, b \rangle \mapsto \langle y, y, y \rangle$, kde

$$y = 0,299r + 0,587g + 0,144b$$

- přijdeme ale o barvy...
- lepší přístup: 3-3-2 paleta

Adaptivně vytvářená paleta

- je potřeba předem znát převáděný obraz
- např: popularity algoritmus, median-cut algoritmus

Pevně zadaná paleta (3-3-2 paleta)

- barevné kanály kódujeme pomocí 3-3-2 bitů
- ukážeme na RGB modelu; RGB krychle se rozdělí na:
 - $2^3 = 8$ řezů ve směru kanálu R
 - $2^3 = 8$ řezů ve směru kanálu G
 - $2^2 = 4$ řezy ve směru kanálu B
- body v průsečících řezů určují barvy RGB palety
- analogicky se postupuje i v případě jiných barevných modelů (např. HSV)
- jak efektivně počítat tyto průsečíky řezů?

Pevně zadaná paleta (3-3-2 paleta)

- chceme vypočítat i tou barvu palety
- převedeme i do binárního tvaru $b_7b_6b_5b_4b_3b_2b_1b_0$
 - bity b_0, b_1 určují modrou barvu
 - bity b_2, b_3, b_4 určují zelenou barvu
 - bity b_5, b_6, b_7 určují červenou
- bity je tedy potřeba
 - 1 posunout
 - 2 maskovat
 - 3 získanou hodnotu přeškálovat

Pevně zadaná paleta (3-3-2 paleta)

- např. $i = 15 = (\text{000}\text{0111})_2$

$$i \text{ shr } 5 = (00000\text{00})_2 = 0$$
$$(0 * 255) \text{ div } 7 = 0$$

$$i \text{ shr } 2 = (00\text{000}\text{011})_2$$
$$(\text{00}\text{000}\text{011})_2 \text{ and } (00000111)_2 = (00000\text{011})_2 = 3$$
$$(3 * 255) \text{ div } 7 = 109$$

$$(\text{000}\text{0111})_2 \text{ and } (00000011)_2 = (000000\text{11})_2 = 3$$
$$(3 * 255) \text{ div } 3 = 255$$

- 15. barva (indexováno od nuly) v paletě má tedy kód $\langle 0, 109, 255 \rangle$: ■ jasná modrá barva s mírně zeleným nádechem

Pevně zadaná paleta (3-3-2 paleta)

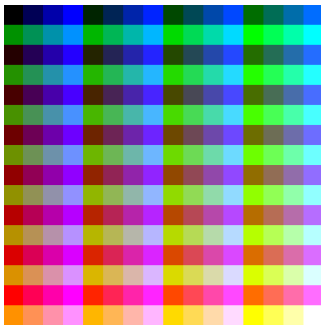
- obecně, i tá barva palety je vypočítána pomocí vztahů:

$$r = ((i \text{ shr } 5) * 255) \text{ div } 7,$$

$$g = ((i \text{ shr } 2) \text{ and } 7) * 255) \text{ div } 7,$$

$$b = ((i \text{ and } 3) * 255) \text{ div } 3,$$

kde shr je operace bitového posunu doprava, and log. součinu, div celočíselného dělení



Pevně zadaná paleta (3-3-2 paleta)



Adaptivně vytvářená paleta (Popularity algoritmus)

- princip: často se vyskytující barvy by měly být v paletě
- vypočítá se histogram (viz později), 256 nejčastěji se vyskytujících barev se zařadí do palety
- co když se v obraze vyskytují rozsáhlé oblasti obarvené velmi podobnými barvami?
- pak algoritmus plýtvá barvami v paletě

Adaptivně vytvářená paleta (hierarchické shlukování podobných barev)

- jak měřit vzdálenost $d(c_1, c_2)$ mezi barvami $c_1 = \langle r_1, g_1, b_1 \rangle$ a $c_2 = \langle r_2, g_2, b_2 \rangle$ v RGB modelu?
- nejlépe Euklidovská vzdálenost v RGB krychli:

$$d(c_1, c_2) = \sqrt{(c_1 - c_2)^2} = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}$$

- vzdálenost shluků: vzdálenost jejich středů
- výpočet středu shluku: aritmetický nebo vážený průměr; výhody, nevýhody

Postup:

- 1 na začátku jsou shluky tvořeny jednotlivými barvami
- 2 spojíme nejbližší dvojice shluků do jednoho
- 3 opakuje bod 2 dokud nedostaneme požadovaný počet shluků (často 256)

Adaptivně vytvářená paleta paleta (Median-cut algoritmus)

- jinak zvaný Heckbertův algoritmus

Postup:

- 1 vybereme barevný kanál s největším rozsahem hodnot
 - 2 vypočítáme medián těchto hodnot
 - 3 barevný prostor rozdělíme na 2 části podle mediánu
 - 4 rekurzivně opakujeme výpočet na obou částech dokud nedostaneme požadovaný počet shluků
 - 5 barva v paletě se určí jako střed shluku, aritmetický nebo vážený průměr
- algoritmus dokáže vytvořit paletu s počtem barev, který je mocninou čísla 2

Zadání 2. miniúkolu



- 1 vytvořte program, který zobrazí pevnou 8bitovou paletu barev pro libovolnou barevnou hloubku jednotlivých kanálů; tzn., na vstupu budou 3 parametry, p_r , p_g a p_b ($p_r + p_g + p_b = 8$), které udávají počet bitů pro kódování červeného, zeleného a modrého kanálu; program pak zobrazí p_r - p_g - p_b paletu
- 2 vytvořte program, který převede vstupní RGB obrázek (32 bpp, component nebo packed ordering) do indexového módu s využitím pevné palety z předchozího bodu