EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Rozvoj lidských zdrojů TUL pro zvyšování relevance, kvality a přístupu ke vzdělání v podmínkách Průmyslu 4.0

CZ.02.2.69/0.0/0.0/16_015/0002329

Úvod do zpracování obrazů

Mechatronika

Prezentace přednášky č. 3

Grafické formáty, geometrické transformace

doc. Ing. Josef Chaloupka, Ph.D.

TECHNICKÁ UNIVERZITA V LIBERCI
www.tul.cz

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

- Komprese obrazových dat

- Bezztrátové algoritmy >>> efektivního kódování, v obraze existují zákonitosti- podobnost některých sousedních pixelů

- Každou posloupnost opakujících se stejných čísel zapíšeme pouze jednou a k ní kolikrát se má opakovat (run-length encoding RLE)
- Speciální kódy pro skupiny hodnot, které se často opakují (např. Lempel-Ziv-Welch neboli LZW algoritmus)
- Pro jednotlivé hodnoty kódy, které mají různý počet bitů a těm, které se opakují nejčastěji, se přidělí kódy nejkratší, zatímco těm nejméně častým nejdelší (entropy coding, Huffmanův kód, Aritmetické kódování)
- Hodnotu každého pixelu vždy nejprve odhadnout na základě hodnot jednoho nebo více předchozích pixelů a zaznamenat pouze rozdíl mezi odhadnutou a skutečnou hodnotou (predictive coding), dokonalá metoda odhadu hodnot >>> rozdíl mezi skutečnou a odhadovanou metodou nula nebo přinejhorším nějaké hodně malé číslo >>> řetězec čísel, kde převažují nuly a malá čísla >>> kódovaní pomocí předchozích metod
- Matematických transformace (např. Haarovy transformace).

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

- GIF (Graphics Interchange Format) od: Compuserve, rozšířený formát na webu, bezztrátový - LZW kódování (patent spol. Unisys – poplatek z komerčního užívání), pro jednoduchou grafiku - maximálně 8 bitů na pixel (256 barev), pro obrázky s malým počtem barev a výraznými hranami, dvě varianty GIFu – 87, 89a (podporuje průhlednost, animace, ...)

- PNG (Portable Network Graphics) (GIF-patentem Unisysu) bezztrátový - starší Lempel-Ziv komprese, 1-48 bitů na pixel, málo rozšířený

- TIFF (Tagged Image File Format) od: Adobe, požadavek vysoké kvality, komplikovaný formát – uložení bez komprese nebo různě komprimované (LZW, run-length, ztrátová JPEG atd.), ne vždy čitelný formát

RLE kódování

- RLE kódování
- Run-length encoding
- Bezztrátová kompresní metoda, kódování posloupnosti stejných hodnot do dvou čísel >>> délka posloupnosti, znak (číslo…)
- Využití >>> především u komprese obrazových dat, kde se v obraze mohou vyskytovat barevné plochy se stejnou hodnotou >>> jpg, tiff, …
- (-) **Nevýhody** – V nejhorším případě, kdy se data ani jednou neopakují může být kompresní poměr = 2, tj. komprimovaný datový řetězec je 2x větší než původní
- Př.:

datový řetězec:	AAAAAFFFFCHHH	(13 znaků)
kódovaný datový řetězec:	5A4F1C3H	(8 znaků)
kompresní poměr:	0,62 (62%)	

LZW

- **Lempel-Ziv-Welchův algoritmus (1978)**
- Kompresní/dekompresní metoda, využití – přenos dat, GIF, TIFF, postscript
- **(+)** rychlá metoda, **(-)** horší kompresní poměr (cca o 30% horší než nejlepší met.)
- Slovník – ukládají se opakující se znaky
- Pokud se vyskytne několik stejných posloupností znaků – nahrazení stejným číslem
- Slovník se neukládá, ale je znovu vytvořen ze zakódovaného souboru

LZW

- Lempel-Ziv-Welchův algoritmus (1978) - komprese

- Př. řetězec abcabcabcbcba (13 znaků), vstupní abeceda (použité znaky): a b c
  přiřazení čísel a = 1, b = 2, c = 3

1. Nalezení nejdelší fráze ze slovníku shodné se vstupem, index na výstup, odebrání
2. Nová fráze = nalezená fráze + jeden další znak

Krok	text vstupu	nalezená fráze	výstup	nová fráze	Index nov. fráze
1	abcabcabcbcba	a	1	ab	4
2	bcabcabcbcba	b	2	bc	5
3	cabcabcbcba	c	3	ca	6
4	abcabcbcba	ab	4	abc	7
5	cabcbcba	ca	6	cab	8
6	bcbcba	bc	5	bcb	9
7	bcba	bcb	9		
8	a	a	1		

Výstup: řetězec 12346591 (8 čísel), kompresní poměr 61,5%

LZW

- Lempel-Ziv-Welchův algoritmus (1978) - dekomprese

- Př. řetězec 12346591, přiřazení čísel a = 1, b = 2, c = 3
- K číslu přiřazená fráze
- Nová fráze = fráze z předchozího kroku + první znak fráze výstupu
- Pokud číslo na vstupu je větší než současné číslo ve vytvářeném slovníku: výstup = nová fráze = minulá fráze + její první znak

Krok	vstup	výstup	nová fráze	Index nov. fráze
1	1	a		
2	2	b	ab	4
3	3	c	bc	5
4	4	ab	ca	6
5	6	ca	abc	7
6	5	bc	cab	8
7	9	bcb	bcb	9
8	1	a		

- Výstup: řetězec abcabcabcbcba

Huffmanovo kódování

- Huffmanovo kódování
- Algoritmus navržen Davidem Huffmanem (1952)
- Využití prefixového kódu - kód žádného znaku není prefixem jiného znaku, neprefixový kód: Morseova abeceda
- Proměnná délka kódových slov >>> „znaky“, které jsou nejvíce četné mají nejkratší délku a naopak
- **Algoritmus kódování**:

1. Zjištění četnosti jednotlivých „znaků“
2. Vytvoření jednotlivých kódů na základě četností
3. Nahrazení jednotlivými znaků v datovém souboru nalezenými kódy

- (+) **Výhody** – rychlá komprese a dekomprese, nenáročné na paměť
- (-) **Nevýhody** – nutnost nalezených kódů, menší kompresní poměr

Huffmanovo kódování

- **Příklad:** znakový řetězec ABRAKADABRA
- 1) četnosti A (5x – 0,46), R (2x – 0,18), B (2x – 0,18), K (1x – 0,09), D (1x – 0,09)
- 2) vytvoření tabulky dle četností
- 3) poslední dvě četnosti se sečtou a zařadí se do tabulky, sčítá se až do 1

A 0,46		A 0,46		A 0,46		KDBR 0,54	1
R 0,18		R 0,18		KDB 0,36	1	A 0,46	0
B 0,18		B 0,18	1	R 0,18	0		
K 0,09	1	KD 0,18	0				
D 0,09	0						

- 4) Posledním dvěma slovům v každém sloupci tabulce přiřadíme 1 (vyšší četnost) a 0 (nižší četnost)
- 5) Výsledný kód znaku >>> posloupnost 0 a 1 dle toho jak se znak seskupoval s dalšími znaky, např. pro znak K: (1) – KDBR, (1) KDB, (0) KD a (1) K. Výsledné kódy A (0), R (10), B (111), K (1101), D (1100)
- 6) Výsledný řetězec pro ABRAKADABRA: 0 111 10 0 1101 0 1100 0 111 10 0, tj.: 01111001101011000111100
- Kompresní poměr (pokud 1 znak = 8 bit.): 0,26 (26%)

Aritmetické kódování

- Aritmetické kódování

- Huffmanův kód >>> problém při stejné pravděpodobnosti výskytu >>> možné řešení >>> aritmetické kódování
- Pro bezztrátovou kompresi dat, proměnná délka kódových slov jako u Huffmanova kódování, při kódování se však vstupní znak nenahrazuje specifickým kódem, ale výsledek, tj. vstupní datový řetězec se nahradí reálným číslem z intervalu <0,1).

- **Algoritmus kódování:**

1. Zjištění četnosti (pravděpodobnosti výskytu) jednotlivých „znaků“
2. Dle pravděpodobnosti výskytu znaků se umístí znak v intervalu <0,1)
3. Celý interval <0,1) je postupně omezován z obou stran na základě přicházejících znaků.
4. Každý znak vybere z aktuálního intervalu odpovídající poměrnou část >>> nový základ pro následující symbol.
5. Po průchodu (načtení) všech znaků dostáváme podinterval z intervalu <0,1), výsledkem je pak libovolné reálné číslo z tohoto intervalu.
6. Na konec kódované zprávy dáme speciální znak, jinak při dekódování není možné určit konec datového toku, nebo uložíme délku původní posloupnosti znaků

Aritmetické kódování

- Aritmetické kódování – př.
- Datový řetězec CBAABCADAC (10 znaků)
- 1) Pravděpodobnosti výskytu A – 0.4 (P1), B – 0.2 (P2), C – 0.3 (P3), D – 0.1 (P4)
- 2) rozdělení v intervalu <0, 1):

<0, P1), <P1, P1 + P2), <P1 + P2, P1 + P2 + P3), <P1 + P2 + P3, P1 + P2 + P3 +P4)

Kumulativní pravděpodobnosti:

Q0 = 0, Q1= P1, Q2 = P1 + P2, … ,QN = P1 + P2 + … + PN = 1

<0, Q1), <Q1, Q2), <Q2, Q3), <Q3, Q4)
<0, 0.4), <0.4, 0.6), <0.6, 0.9), <0.9, 1)
A B C D

Aritmetické kódování

- Aritmetické kódování – př.

- Datový řetězec CBAABCADAC (10 znaků)

- Rozdělení intervalu

<0, 0.4), <0.4, 0.6), <0.6, 0.9), <0.9, 1)

A B C D

- 3) Kódování >>> postupné omezování intervalu I = <0, 1), I = <L, H)
- >>> Postupně jsou brány znaky z datového řetězce, k nim známe IZ = <ZL, ZH)
- >>> Nová hodnota intervalu IN = <L + ZL*(H - L), L + ZH*(H - L))

C >>> I = <0, 1), IN = <0 + 0.6*(1 – 0), 0 + 0.9*(1 - 0)) = <0.6, 0.9)
B >>> I = <0.6, 0.9), IN = <0.6 + 0.4*(0.9 – 0.6), 0.6 + 0.6*(0.9 – 0.6)) = <0.72, 0,78)
A >>> I = <0.72, 0.78), IN = <0.72 + 0*(0.78 – 0.72), 0.72 + 0.4*(0.78 – 0.72)) = <0.72, 0,744)
A >>> I = <0.72, 0.744), IN = <0.72 + 0*(0.744 – 0.72), 0.72 + 0.4*(0.744 – 0.72)) = <0.72, 0,7296)
B >>> I = <0.72, 0.7296), IN = <0.72 + 0.4*(0.7296 – 0.72), 0.72 + 0.6*(0.7296 – 0.72)) = <0.72384, 0.72576)
….
….
B >>> I = <0.72519936, 0.725208576), IN = <0.7252048896, 0.7252076544), C = 0.725205

Aritmetické kódování

- Aritmetické dekódování – př.

- Datový řetězec CBAABCADAC (10 znaků)

- Rozdělení intervalu

<0, 0.4), <0.4, 0.6), <0.6, 0.9), <0.9, 1)

A B C D

- Výsledek kódování >>> C = 0.725205

- 1) Počáteční hodnota intervalu dekódování I = <0, 1)
- 2) dekódování znaku: K = ((C – L) / (H – L)); ZL <= K < ZH >>> nalezneme odpovídající znak
- 3) počítáme nový interval IN = <L + ZL*(H - L), L + ZH*(H - L))
- I = <0, 1), K = 0.725205, znak = C, IN = <0 + 0.6*(1 – 0), 0 + 0.9*(1 – 0) = <0.6, 0.9)
- I = <0.6, 0.9), K = 0.41735, znak = B, IN = <0.6 + 0.4*(0.9 – 0.6),0.6 + 0.6*(0.9 – 0.6)) =<0.72, 0.78)
- I = <0.72, 0.78), K = 0.08675, znak = A, IN = <0.72, 0.744)
- I = <0.72, 0.744), K = 0.216875, znak = A, IN = <0.72, 0.7296)
- …

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

- Ztrátové algoritmy >>> část informace obsažené v obrázku lze neuvažovat, bez viditelného rozdílu >>> citlivost lidského oko

- Transformace obrazu, separace jasu a barvy HSI (Hue, Saturation a Intensity) barevnou informaci zaznamenat jen v hrubším rozlišení

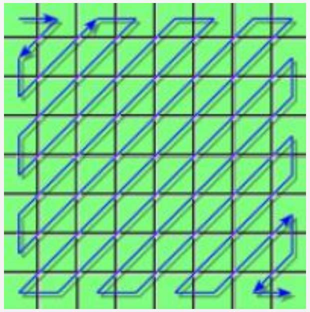
- Transformace obrazu do frekvenční oblasti (Fourierova nebo kosinová transformace, wavelety apod.) – zanedbání vyšších frekvencí (v diagonálním směru-malá citlivost lidského oka), zbylé frekvence se různě hrubě kvantizují (zaokrouhlují) podle důležitosti **(+)** větší komprese, **(-)** nevratné ztráty

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

- JPG (JPEG, přesněji JFIF) od: Joint Photography Experts Group, ukládání fotografií, hlavní formát pro prezentaci fotografií na webu, digitální fotoaparáty, ztrátový formát založený na diskrétní kosinové transformaci >>> malé čtverečky o rozměrech 8x8 pixelů (rychlost), velká komprese >>> viditelné artefakty tvaru čtverečků, kontury místo plynulých přechodů barev, vzorečky v oblastech s drobnou texturou, „duchové" kolem hran; náhled z nízkých frekvencí

- **Postup JPG komprese**:

- 1) Převod do barevného prostoru YCbCr
- 2) Převzorkování (4:4:4, 4:2:2, 4:2:0)
- odstranění části barevné informace
- 3) Rozdělení obrázku do makrobloků 8x8
- 4) Výpočet DCT: Diskrétní kosinová transformace
- 5) Kvantizace pomocí kvantizační tabulky
- zahození části informace – vysoké frekvence
- 6) Linearizace (zig-zag)
- 7) RLE kódování
- 8) Huffmanovo nebo aritmetické kódování



GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

- **FPX** (Flashpix) od: Kodak, Hewlet Packard, Microsoft a Live Picture, prohlížení velkých obrázků webovým browserem, formát vlastní nezávislé konsorcium Digital Imaging Group, obrázek je v souboru uložený jako pyramida různých rozlišení – zoomování, ztrátová komprese je ztrátová, malé rozšíření
- **PCD** (PhotoCD) od: Kodak, čitelnost formátu, málokterý editor umí ukládat, bezztrátová komprese je bezztrátová, obrázek ve formě pyramidy, profesionální skenování fotografií
- **PSD** pro: Adobe Photoshop, bezztrátový formát, podporuje vrstvy, alfa kanály, cesty apod.
- **PDF** (Portable Document Format) od: Adobe, pro elektronické publikace, ztrátová, bezztrátová komprese, prohlížeč zdarma
- **EPS** (Encapsulated PostScript) od: Adobe, pro přidávání obrázku do dokumentů, vektorová grafika, rastrové obrázky.
- **BMP** bitmapa, indexované a RGB obrázky, 1, 4, 8 nebo 24... bitů na pixel, nekomprimovaná nebo bezztrátově komprimovaná data pomocí RLE.

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU

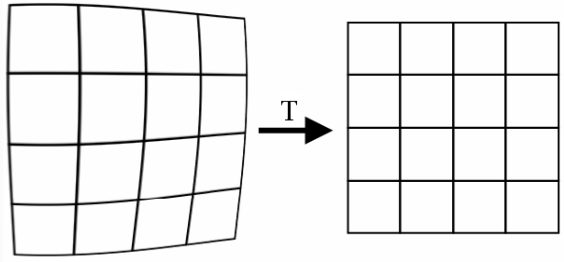
- XBM (X BitMap), bitmapa pro X Windows (Unix)
- PICT formát, používá Macintosh, bez komprese nebo run-length encoding
- TGA (Truevision/Targa), v oblasti počítačové grafiky (hry a pod.).
- Další formáty >>> nové, staré, speciální programy, satelitní snímky, otisky prstů

VELIKOST OBRAZU

- DPI >>> Dot Per Inch >>> body na palec; 1 palec -> 2,54 cm
- rozměr v cm = počet bodů * 2,54 / DPI
- počet bodů = rozměr v cm * DPI / 2,54
- př.: rozměr = 9 cm, 300 DPI, velikost v bodech = ?
  velikost v bodech = 9 * 300 / 2,54 · 1063 pixelů

GEOMETRICKÉ TRANSFORMACE

- Transformace >>> zvětšení, rotace, odstranění geometrického zkreslení obrazu



- Geometrická transformace >> vektorová funkce >> zobrazí bod x, y do bodu x', y'

$$x' = T_X(x, y) \quad y' = T_Y(x, y)$$

- T_x, T_y známé nebo je hledáme na základě znalosti původního a transformovaného obrazu >>> známé (vlícové) body

- Geometrická transformace 1) transformace souřadnic bodů
  2) aproximace jasové funkce

- Transformace souřadnic bodů

$$x' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} a_{rk} x^r y^k, \quad y' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} b_{rk} x^r y^k$$

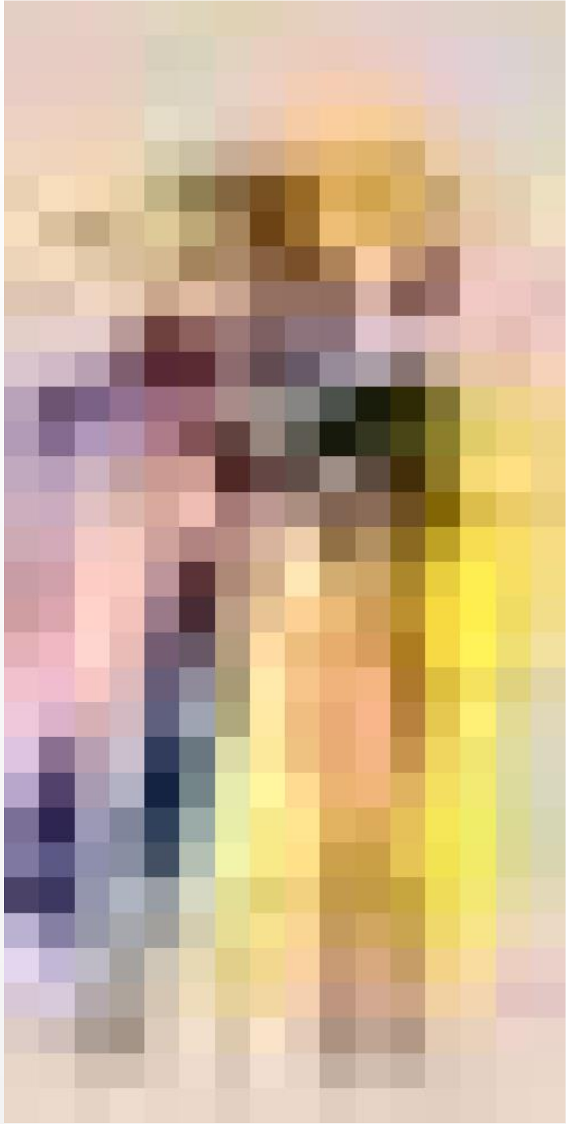
výpočet a_{rk}, b_{rk} metodou nejmenších čtverců

GEOMETRICKÉ TRANSFORMACE

GEOMETRICKÉ TRANSFORMACE

OR

GEOMETRICKÉ TRANSFORMACE



GEOMETRICKÉ TRANSFORMACE

- Nahrazení bilineární transformací >>> pro malé změny m = 2 až 3

$$x' = a_0 + a_1 x + a_2 y + a_3 xy \qquad y' = b_0 + b_1 x + b_2 y + b_3 xy$$

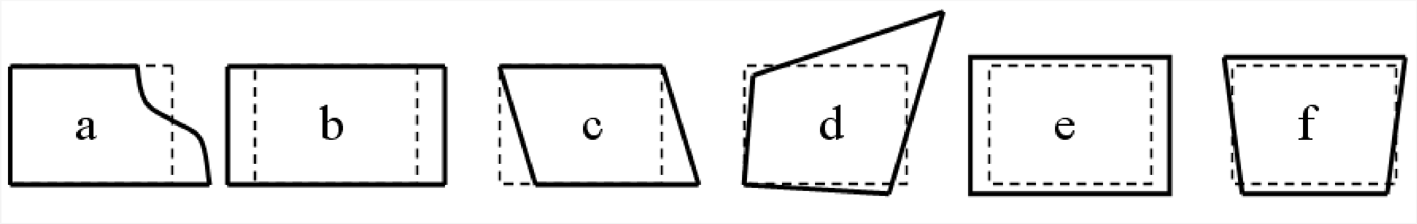
- Afinní transformace >>> zvláštní případ bilineární transformace

$$x' = a_0 + a_1 x + a_2 y \qquad y' = b_0 + b_1 x + b_2 y$$

- Převedení do homogenních souřadnic

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_0 \\ b_1 & b_2 & b_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Složité geometrické zkreslení >>> rozdělení obrazu na menší části



- a-rozdílná vzdálenost objektů od zrcadla senzoru, b-rovinné zkreslení (skener), c-vesmírný skener-otáčení zeměkoule, e-kamera se vzdaluje od objektů, f-perspektivní zobrazení – model dírkové komory

GEOMETRICKÉ TRANSFORMACE

- Jakobián J – informace o změně systému při geometrické transformaci

$$J = \left| \frac{\partial(x',y')}{\partial(x,y)} \right| = \begin{vmatrix} \frac{\partial x'}{\partial x} & \frac{\partial x'}{\partial y} \\ \frac{\partial y'}{\partial x} & \frac{\partial y'}{\partial y} \end{vmatrix}$$

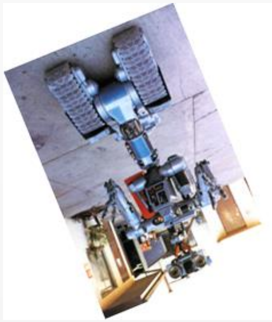
singulární transformace J = 0
obraz invariantní vůči transformaci J = 1

$$J = a_1 b_2 - a_2 b_1 + (a_1 b_3 - a_3 b_1)x + (a_3 b_2 - a_2 b_3)y$$

pro bilineární transformaci

$$J = a_1 b_2 - a_2 b_1$$

pro afinní transformaci



- **Rotace** o úhel Φ proti originálnímu obrazu

$$x' = x.\cos\Phi + y.\sin\Phi \quad y' = -x.\sin\Phi + y.\cos\Phi \quad J = 1$$

- **Změna měřítka**

$$x' = ax \quad y' = bx \quad J = ab$$

- **Zkosení** o úhel Φ

$$x' = x + y.\tan\Phi \quad y' = y \quad J = 1$$

GEOMETRICKÉ TRANSFORMACE

- Výpočet souřadnic v původním obraze

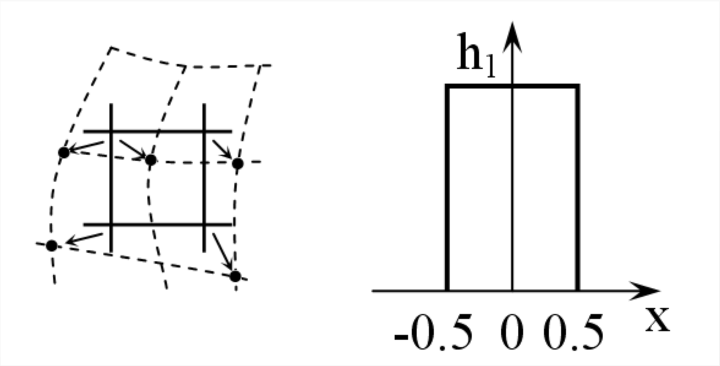
$$(x,y)=T^{-1}(x',y')$$

- Výsledný (interpolovaný) jas

$$f_n(x,y)=\sum_{l=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}g_s(l\Delta x,k\Delta y).h_n(x-l\Delta y,y-k\Delta y)$$

hn... interpolační jádro

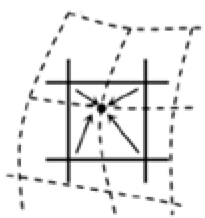
- 1) Metoda nejbližšího souseda

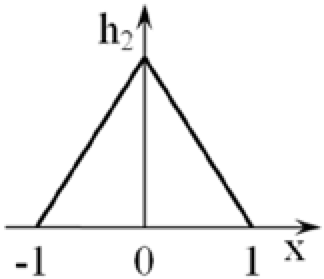


$$f_1(x,y)=g_s(round(x),round(y))$$

GEOMETRICKÉ TRANSFORMACE

2) Lineární interpolace

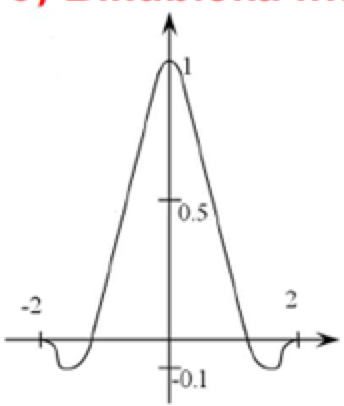




$$f_2(x,y) = (1-a).(1-b).g_s(l,k) + a.(1-b).g_s(l+1,k) + b.(1-a).g_s(l,k+1) + a.b.g_s(l+1,k+1)$$

$$l = round(x) \quad k = round(y) \quad a = x - l \quad b = y - k$$

3) Bikubická interpolace - bikubický polynom, okolí 16 bodů



$$h_3 = \begin{cases} 1 - 2|x|^2 + |x|^3 & pro\ 0 \le |x| < 1 \\ 4 - 8|x| + 5|x|^2 - |x|^3 & pro\ 1 \le |x| < 2 \\ 0 & jinde \end{cases}$$

Další interpolace – fraktály…

GEOMETRICKÉ TRANSFORMACE

- **Bikubická interpolace** - algoritmus

1. id = i‘.(w / w‘)
   jd = j‘.(h / h‘)
   w (šířka),h (výška) originálního obrazu
   w‘ (šířka),h‘ (výška) nového obrazu

2. ic = c.č (id) id=1.8 -> ic = 1
   jc = c.č (jd)

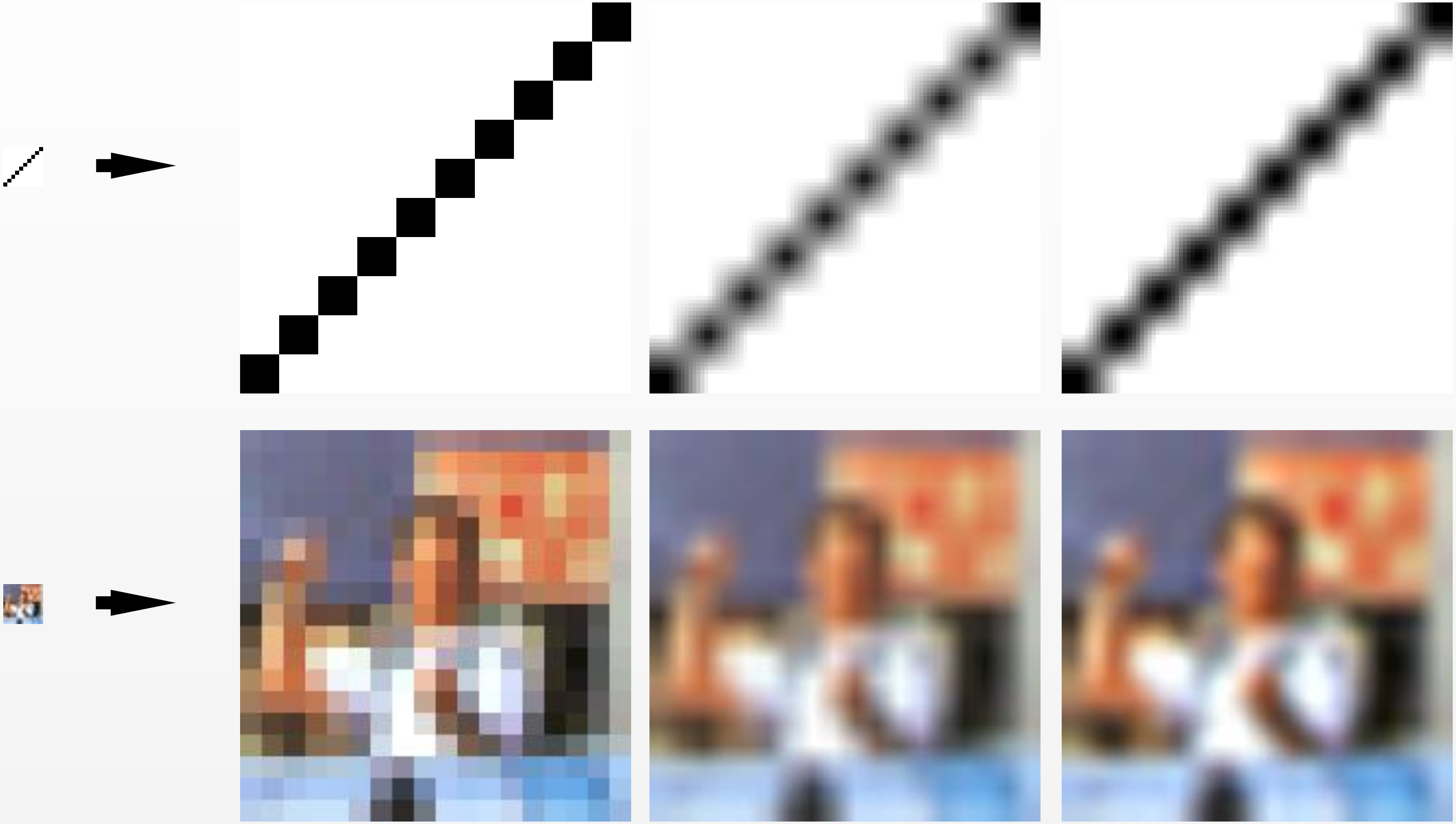
3. dx = id – ic
   dy = jd – jc

4. f (i‘, j‘) = f (ic + m, j + n).R (m - dx).R(dy - n)

5. R (x) = (1/6).[P(x + 2)3 - 4 P(x + 1)3 + 6 P(x)3 - 4 P(x -1)3]

6. P(x) = x, pro x > 0
   P(x) = 0, pro x ≤ 0

GRAFICKÉ FORMÁTY ULOŽENÍ OBRAZU



Nejbližší soused | Lineární interpolace | Bikubická interpolace