

# Základy počítačovej grafiky a spracovanie obrazu

Ing. Viktor Kocur  
victor.kocur@fmph.uniba.sk

DAI FMFI UK

27.11.2017

# Obsah

- 1 Rekapitulácia
- 2 Škálovanie obrázkov
  - Nearest-neighbor transformácia
  - Bilineárna transformácia
- 3 Histogram
  - Histogram
  - Ekvalizácia histogramu
- 4 Šum
  - Aditívny gausovský šum
  - Impulzný šum
- 5 Vyhladzovanie
  - Princíp
  - Konvolučné vyhladzovanie
  - Mediánová filtrácia

# Rekapitulácia z minula

## Matlab - základy

- Operácie - maticové vs. po zložkách
- Indexácia - jedným indexom, n-ticou indexov, logickou maticou
- Funkcie - konštrukcia funkcií pomocou m-file
- Control flow - if, switch, for, while

## Základy práce s obrázkami

- imread
- imshow vs. image a imagesc
- RGB vs. grayscale
- Datové typy!

# Obrázky na prácu

## Obrázky a skripty

Stiahnite si zip z <http://sccg.sk/~kocur/>.

# Úloha

## Scale

Vytvorte funkciu `myimresize(I,s)`, ktorá vráti obrázok zväčšený pomerom `s` pomocou metódy `nearest point interpolation`. Ak spravíte algoritmus s for cyklom, tak sa pokúste z neho spraviť iba s vektorovými operáciami.

## Riešenie - bez forcyklu

```
function I = myimresize(I, s)
    oldrows = size(I,1);
    oldcols = size(I,2);
    r = round(linspace(1,oldrows,round(s*oldrows)));
    c = round(linspace(1,oldcols,round(s*oldcols)));
    I = I(r,c);
end
```

## Príklad - myimresize2.m

## Bilineárna interpolácia

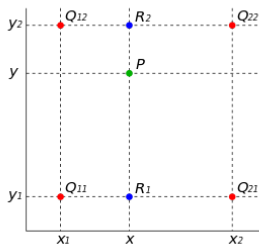
$$f(x, y) = a_{00} + a_{10}(x - x_1) + a_{01}(y - y_1) + a_{11}(x - x_1)(y - y_1)$$

$$a_{00} = f(x_1, y_1)$$

$$a_{10} = f(x_2, y_1) - f(x_1, y_1)$$

$$a_{01} = f(x_1, y_2) - f(x_1, y_1)$$

$$a_{11} = f(x_2, y_2) + f(x_1, y_1) - (f(x_2, y_1) + f(x_1, y_2))$$



# Histogram obrázku

Pozn.:

Od teraz budeme pracovať s šedotónovými obrázkami (.pgm)

## Nakreslenie histogramu

```
I = imread('zatisie.pgm');  
imshow(I);  
figure;  
histogram(I);  
J = imread('krajinka.pgm');  
figure;  
imshow(J);  
figure;  
histogram(J);  
figure;  
histogram(J, 'BinLimits', [0 255]);
```

# Histogram obrázku - úloha

## Úloha

Na histograme obrázku zátišia sú tri peaky. Upravte obrázok, tak aby pixely približne patriace len jednému z peakov boli úplne biele.



## Ekvalizácia pomocou histeq

```
J = imread('krajinka.pgm');  
JEQ = histeq(J,256);  
subplot(2,2,1);  
imshow(J);  
subplot(2,2,2);  
imshow(JEQ);  
subplot(2,2,3);  
histogram(J,'BinLimits',[0 255]);  
subplot(2,2,4);  
histogram(JEQ,'BinLimits',[0 255]);
```

# Popis

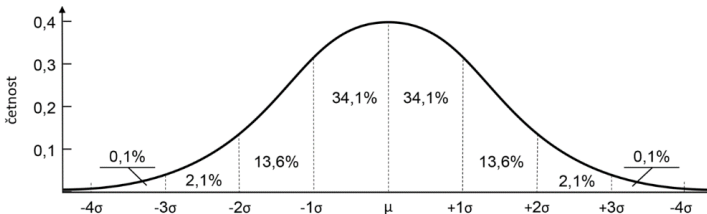
## Aditivita

$$I = I_{real} + S$$

## Gaussovský charakter šumu

$$P(S_{i,j} = x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$\sigma$  - stredná odchylka,  $\mu$  - stredná hodnota



## Generovanie gausovského šumu v matlabe

`randn(size(I))` - funkcia generuje maticu náhodných čísel s gaussovskou distribúciou  $\mu = 0$  a  $\sigma = 1$  s rovnakou veľkosťou ako matica `I`.

## Úloha

Vytvorte funkciu `zasum(I,s)`, vráti obrázok `I` zašumený gaussovským šumom s  $\sigma = s$ . Zašumte si ľubovoľný obrázok s rôznymi strednými odchýlkami.

## Hint

Najprv treba obrázok pretypovať na `double` pomocou `im2double` a potom zašumený naspäť pomocou `im2uint8`!

## Riešenie

```
function O = zasum(I,s)
    O = im2uint8(im2double(I) + s/256*randn(size(I)));
end
```

# Impulzný šum

## Impulzný šum - korenie a soľ

Impulzný šum zmení s určitou pravdepodobnosťou jeden pixel na úplne čierny (korenie), alebo úplne biely (soľ).

## Generovanie impulzného šumu v matlabe

Použijeme funkciu `rand(size(I))`, ktorá vytvorí maticu, ktorej prvky sú náhodné s rovnomernou distribúciou na intervale  $< 0, 1 >$ .

# Impulzný šum - úloha

## Impulzný šum - korenie a soľ

Vytvorte funkciu `okoren(I,p1,p2)`, ktorá vráti obrázok `I` zašumený impulzným šumom s pravdepodobnosťou `p1` pre soľ a `p2` pre korenie. Otestujte si na nejakom obrázku rôzne pravdepodobnosti zašumenia.

## Riešenie

```
function I = okoren(I,p1,p2)
    R = rand(size(I));
    I(R < p1) = 0;
    I(R > 1-p2) = 255;
end
```

# Princíp

## Prečo chceme vyhladzovať?

Pri pozorovaní jemne zašumeného obrázku okom je stále jednoduché pozorovať obrázok a porozmieť jeho obsahu. V počítačovom videní však často aj slabý šum môže narušiť funkčnosť niektorých algoritmov počítačového videnia. Preto je veľmi často vhodné zmierniť mieru šumu. To sa dá dosiahnuť rôznymi vyhladzovacími metódami.

## Princíp

Obrázok vyhladzujeme tak, že pre každý pixel sa pozeráme na jeho okolie a z neho usúdime novú hodnotu daného pixelu.

# Konvolúcia a korelácia

## 2D Konvolúcia

$$J = I * M \iff J(r, c) = \sum_{u=-h}^h \sum_{v=-h}^h I(r - c, c - v)M(u, v)$$

## 2D Korelácia

$$J = I \circ M \iff J(r, c) = \sum_{u=-h}^h \sum_{v=-h}^h I(r + c, c + v)M(u, v)$$

## Konvolúcia vs. Korelácia

Pre symetrické masky je konvolúcia totožná s koreláciou. Obecné má však konvolúcia lepšie niektoré vlastnosti. Preto používame hlavne termín konvolúcia.

# Konvolúcia a korelácia

1	1	2	5	6	3	6	7	3
2	3	4	6	7	5	1	8	4
8	7	6	5	7	6	3	3	4
2	3	5	6	7	8	2	7	3
4	5	3	2	1	6	8	7	2
1	4	5	3	2	6	7	8	1
2	3	4	5	6	8	9	2	1

Input image

\*  $\frac{1}{9}$ 

1	1	1
1	1	1
1	1	1

Mask



Convolution operation

1	1	1	5	6	3	6	7	3
2	3	4	6	7	5	1	8	4
8	7	6	5	7	6	3	3	4
2	3	5	6	7	8	2	7	3
4	5	3	2	1	6	8	7	2
1	4	5	3	2	6	7	8	1
2	3	4	5	6	8	9	2	1

1	2	3	4	4	4	4	4	3
3	4	5	6	6	5	5	5	4
3	5	5	6	7	6	5	4	4
4	5	5	5	6	6	6	5	3
3	4	4	4	5	6	7	5	3
3	4	4	4	5	6	7	5	3
2	3	3	3	4	5	5	4	2

Output Image



# Konvolučné filtre v matlabe

## Funkcie

- `conv2(A,B)` - vráti 2D konvolúciu matíc A a B
- `imfilter(A,h)` - vráti obrázok A filtrovaný filtrom (maticiou) h  
!!! Lepšie optimalizovaný je `imfilter`

## Čo sa deje na okrajoch?

Keď sme na okrajoch obrázka, tak filter "siaha" mimo obrázka. Preto je potrebné definovať aké hodnoty máme doplniť.

Použijeme `imfilter(A,h,'option')`, kde options sú:

- `X` - pixely mimo obrázka akoby majú hodnotu `X` - defaultne 0
- `'symmetric'` - za krajom sú pixely zrkadlovo obrátené z obrázka
- `'replicate'` - opakujeme pixely z okraja
- `'circular'` - filter presiahne na opačnú stranu obrázka

# Masky

## fspecial

- `fspecial('average',hsize)`
- `fspecial('gaussian',hsize,sigma)`
- `fspecial('laplacian',alpha)`

## Úloha

Vytvorte si priemerovacie gaussovské filtre s rôznymi parametrami a aplikujte ich na obrázok, ktorý ste zašumeli aditívnym gaussovským šumom tiež s rôznymi parametrami. Otestujte tieto filtre aj na obrázky s impulzným šumom.

# Odstraňovanie impulzného šumu

## fspecial

V poslednej úlohe sme si všimli, že konvolučná filtrácia nám nepomohla odstrániť impulzný šum. Preto potrebujeme iný nástroj.

## Mediánová filtrácia

Mediánová filtrácia zoberie okolie pixelu a vypočíta medián hodnôt, ktoré sa v ňom nachádzajú. Tento medián potom bude hodnota pixelu vo vyfiltrovanom obrázku.

# Mediánová filtrácia

## Mediánová filtrácia v matlabe

- `medfilt2(A)` - vráti obrázok po mediánovej filtrácii v okne  $3 \times 3$
- `nlfilter(A,[m n],fun)` - obecnější filter, aplikuje vždy `fun` na okno  $m \times n$

## Úloha

Zašumte obrázok impulzným šumom s rôznymi pravdepodobnosťami a následne ho vyfiltrujte mediánovým filtrom. Skúste filtrovať aj obrázky zašumené aditívnym šumom.