

NPGR032

CV. 01

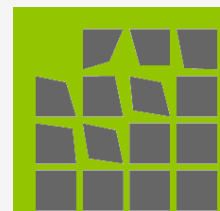
Úvod

MFF

ZS 2016

ÚTIA - ZOI

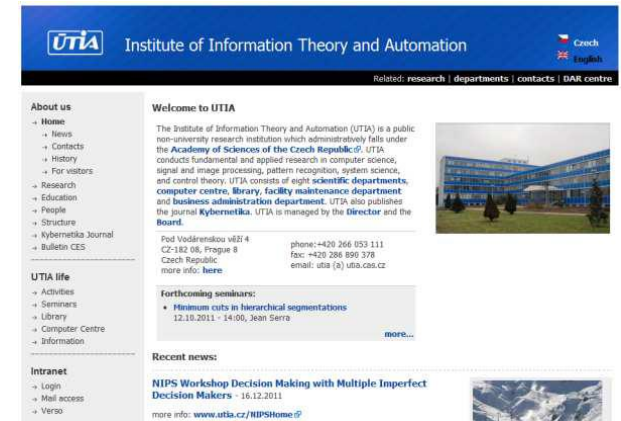
zoi.utia.cas.cz



Kontakty

- Ústav teorie informace a automatizace AV ČR, v.v.i.

- <http://www.utia.cas.cz>



The screenshot shows the homepage of the Institute of Information Theory and Automation (UTIA). The header features the UTIA logo and the text "Institute of Information Theory and Automation" along with the Czech flag and "UTIA logo". A navigation bar includes links for "research", "departments", "contacts", and "DAR centre". The main content area is divided into several sections: "About us" with a list of links (Home, News, Contacts, History, For visitors, Research, Education, People, Structure, Kybernetika Journal, Bulletin CES); "UTIA life" with links (Activities, Seminars, Library, Computer Centre, Information); "Intranet" with links (Login, Mail access, Verso); "Welcome to UTIA" with a detailed description of the institute and its departments; "Forthcoming seminars" listing a seminar on "Maximum cuts in hierarchical segmentations" on 12.10.2011; and "Recent news" featuring a "NIPS Workshop Decision Making with Multiple Imperfect Decision Makers" on 16.12.2011. A photograph of the UTIA building is visible on the right side.

- Zpracování obrazové informace

- <http://zoi.utia.cas.cz>



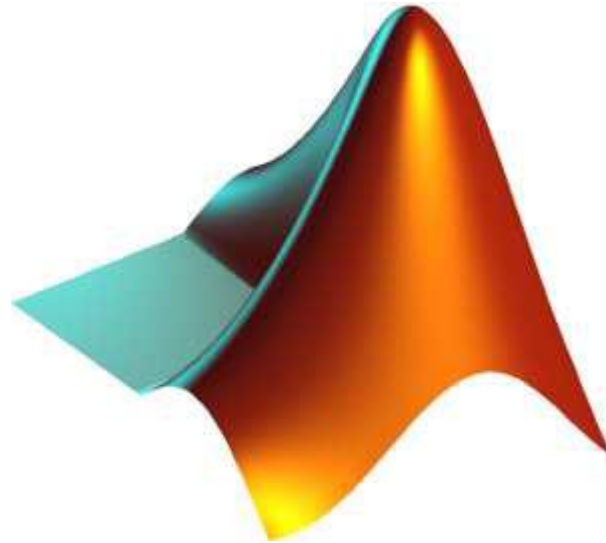
The screenshot shows the website of the Department of Image Processing. The header includes the department name and a logo. The main content area is divided into several sections: "About us" with a list of links (Contacts, News, People, Research, Publications, Seminars, Conference materials, Downloads, For students, Useful links, Login); "Welcome to the Department of Image Processing" with contact information for the Head of the Department (Erich Steiner), Vice head (Eva Svoboda), Secretary (Zena Travníková), and phone number (+420 284 680 730); "Fax: +420 284 680 730"; "address: Institute of Information Theory and Automation of the AS CR, Pod Vítězskou ul. 4, CZ-182 08, Praha 8, Czech Republic"; and a list of research areas including "Recognition of distorted images and patterns by invariant descriptors regardless of their actual position in the scene", "Registration and fusion of several images of the same scene taken at different times, by different sensors and/or from different viewpoints in order to obtain information of higher quality", "Theory of nearest neighbors, convexity, classification, outliers, affine invariants and invariance to translation", and "Restoration of degraded images, namely multichannel blind deconvolution, edge-preserving denoising, local contrast enhancement, and color transformations". A large image showing various image processing results is displayed on the right side.

Kontakty

- Adam Novozámský: novozamsky@utia.cz

Matlab

- Hned po přihlášení zapnout!!
 - (omezený počet licencí na MFF)
- Stáhnout balík souborů na cvičení:
 - <http://zoi.utia.cas.cz/NPGR032/materialy>

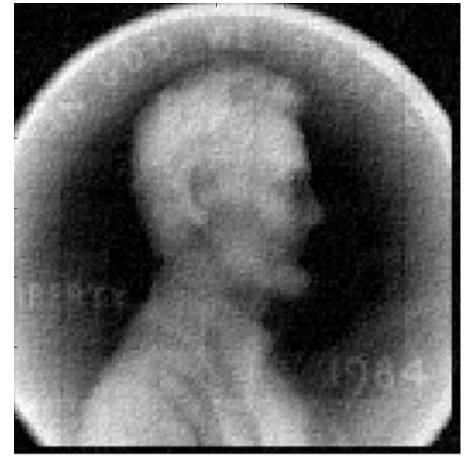
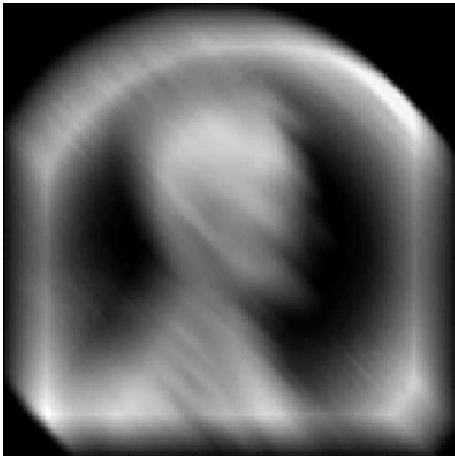


Motivace

- Předzpracování snímku
 - Využití Fourierovi transformace



- Potlačení šumu, dekonvoluce

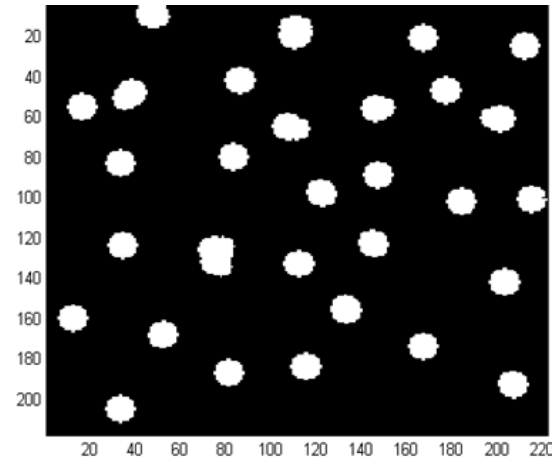
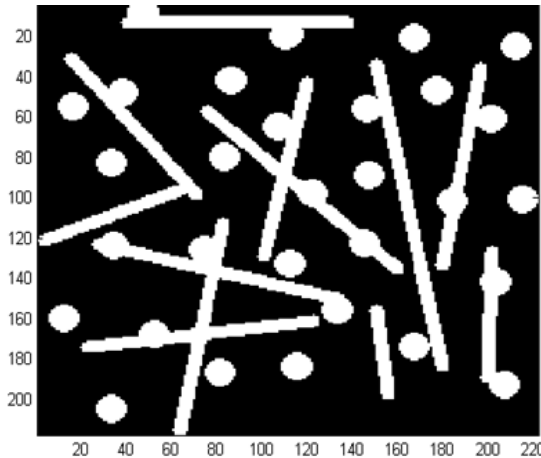


Motivace

- Informace z obrázku
 - Detekce hran

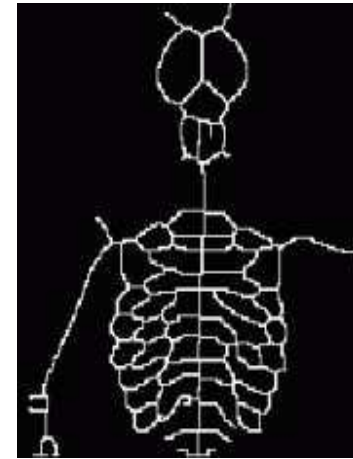


- Morfologie – potlačení objektů

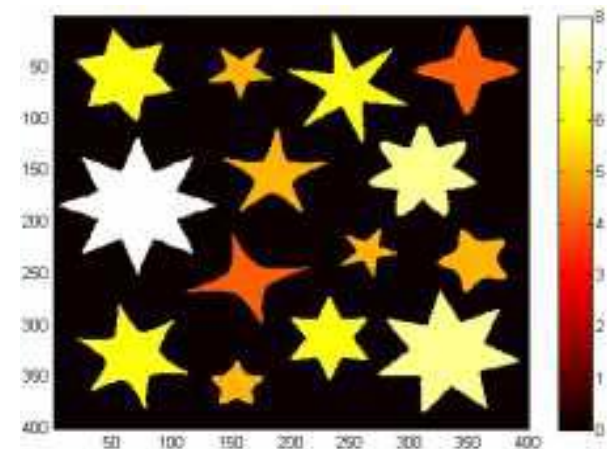
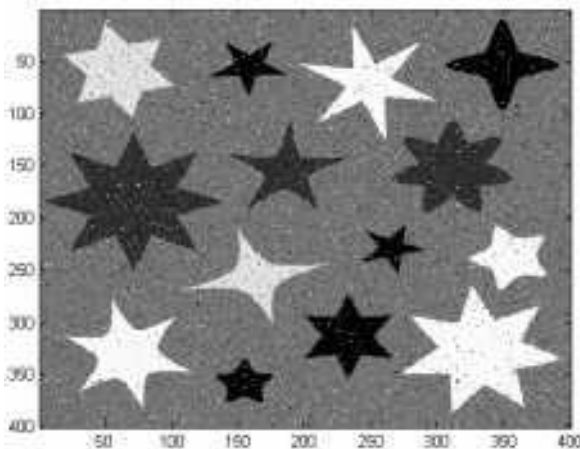


Motivace

- Praktická zkouška
 - 2004 – Kostra kostry

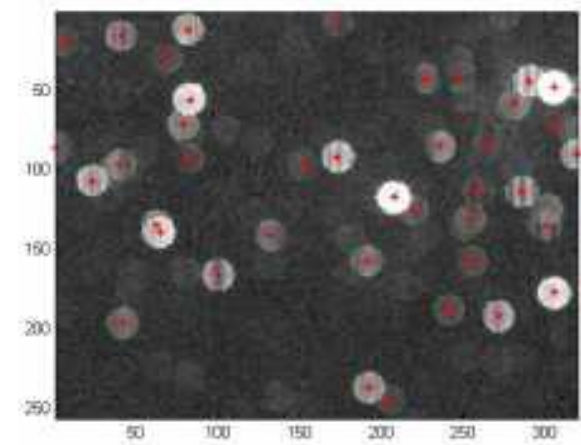
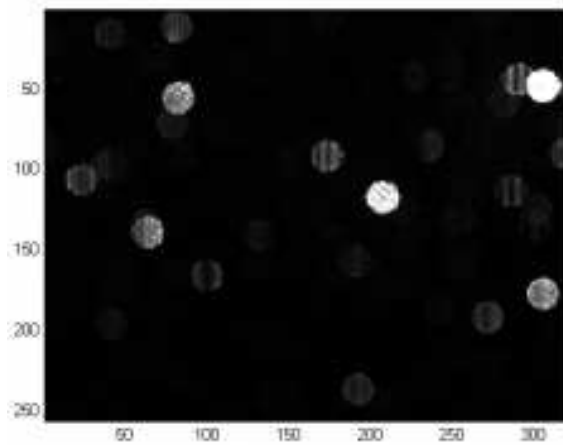


- 2005 – Vánoční atmosféra



Motivace

- Praktická zkouška
 - 2007 – Něco je ve vzduchu



- Zadání písemek:

<http://zoi.utia.cas.cz/zadani-pisemek-z-lonskych-let>

Základy MATLABU

- Úplné základy najdete zde:

http://zoi.utia.cas.cz/files/roz1/matlab_reference_sheet.pdf

Zde je oficiální HELP Matlabu:

<http://www.mathworks.com/help/index.html>

- Nebo nápověda přímo v Matlabu
 - help, doc, F1

Načtení snímku

```
I = imread('lena2.png');
```

Zobrazení matice

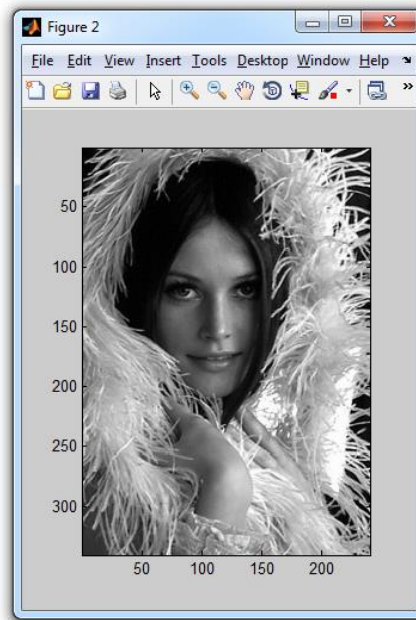
- Vytvořte funkci, která zobrazí matici **Img**

```
function zobr (Img)
```

```
% Img ... matice obrazových dat;
```

```
• • •
```

```
end
```



help: figure, imagesc, axis, colormap, gray

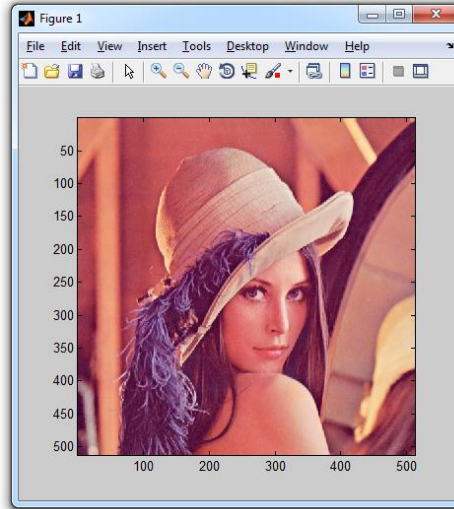
Zobrazení matice

- Vytvořte funkci, která zobrazí matici **Img**

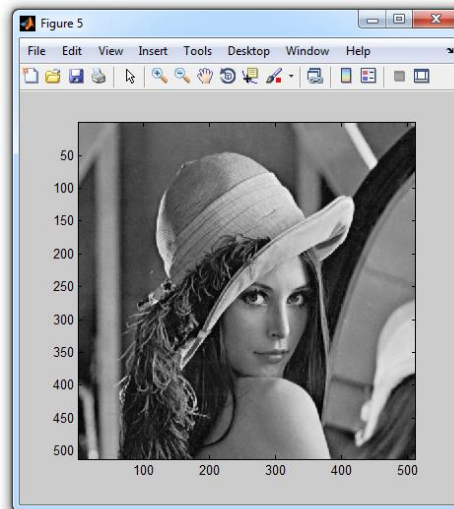
```
function zobr (Img)
% Img ... matice obrazových dat;
figure;
colormap (gray (256) ) ;
image (Img) ;
axis image;
end
```

Práce s vícekanálovým obrázkem

1. Zobrazte barevný obrázek `lena.pgm`



2. Zobrazte 2 kanál (green) obrázku: `lena.pgm`



Práce s vícekanálovým obrázkem

1. Zobrazte barevný obrázek `lena.png` (3-D matice)

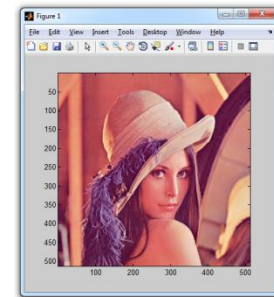
- pokud jsou data typu `uint8`:

- rozmezi 0 – 255

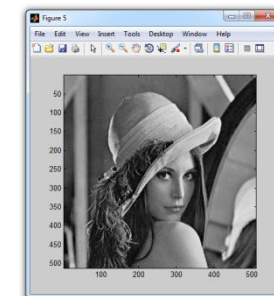
```
I = uint8(imread('lena.png'));  
zobr(I);
```

- pokud jsou data typu `double`, tak funkce `image` potřebuje mít v tomto typu data v rozmezí $<0, 1>$, proto dělíme hodnoty 0-255 v takovém případě 255

```
I = double(imread('lena.png'));  
zobr(I/255);
```



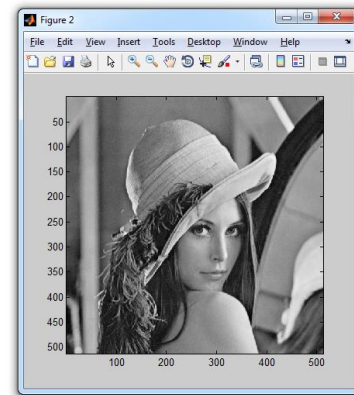
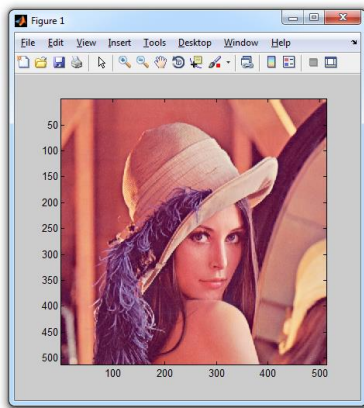
2. Zobrazte 2 kanál (green) obrázku: `lena.png`



RGB to GRAY

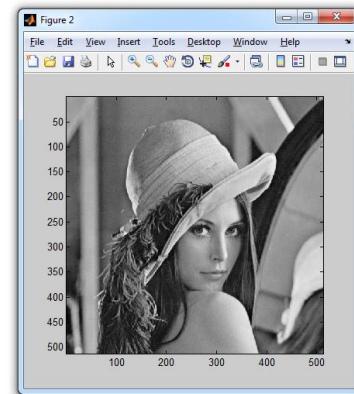
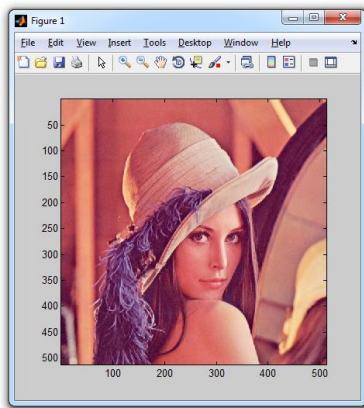
- Napište funkci `rgb2grayscale (Img)`, která převede RGB obrázek `Img` na šedotónový
- Vyzkoušejte funkci na obrázku `lena.pgm`

```
function [Y] = rgb2grayscale (Img)  
% Img ... matice obrazových dat;  
...  
end
```



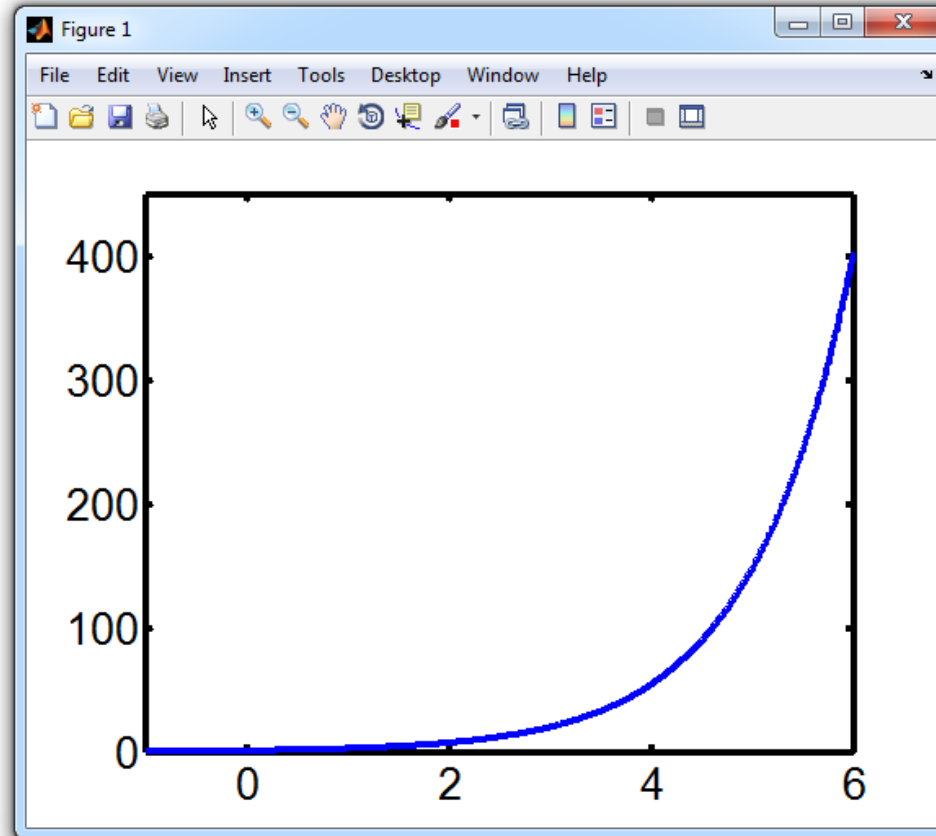
RGB to GRAY

```
function [Y]=rgb2grayscale(I)  
R = I(:, :, 1);  
G = I(:, :, 2);  
B = I(:, :, 3);  
Y = 0.2989*R + 0.5870*G + 0.1140*B;  
end
```



Práce s maticemi

- Zobrazte průběh exponenciály od -1 do 6



`help: exp, plot`

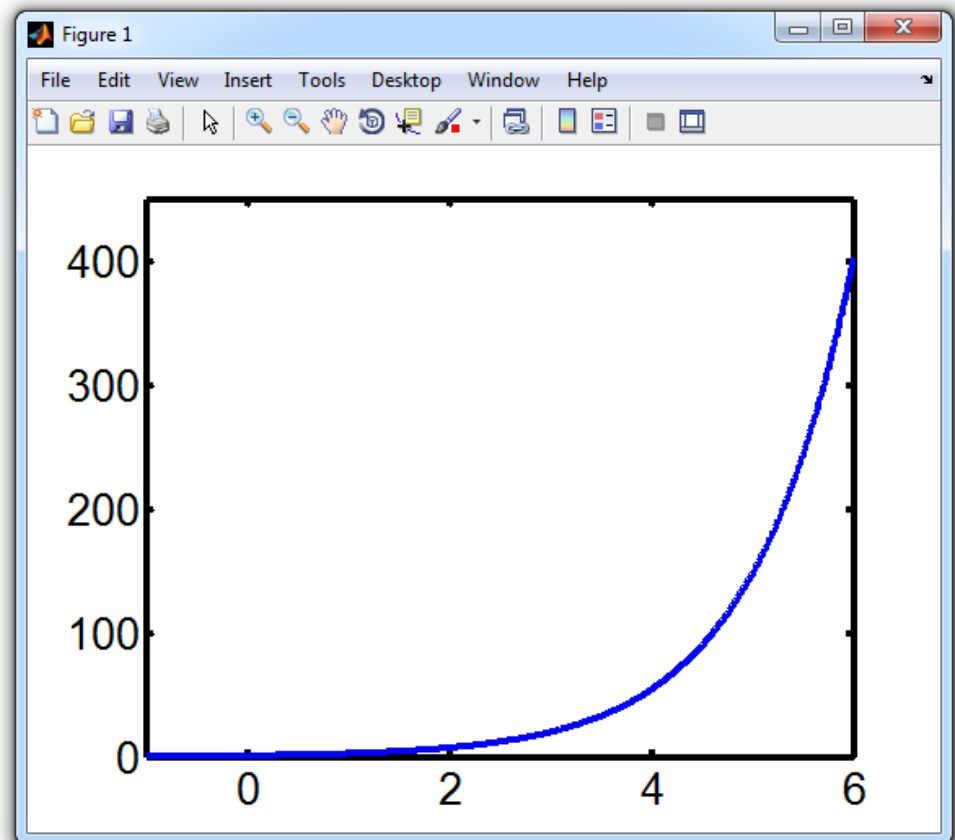
Práce s maticemi

- Zobrazte průběh exponenciály od -1 do 6

```
x = -1 : 0.01 : 6;
```

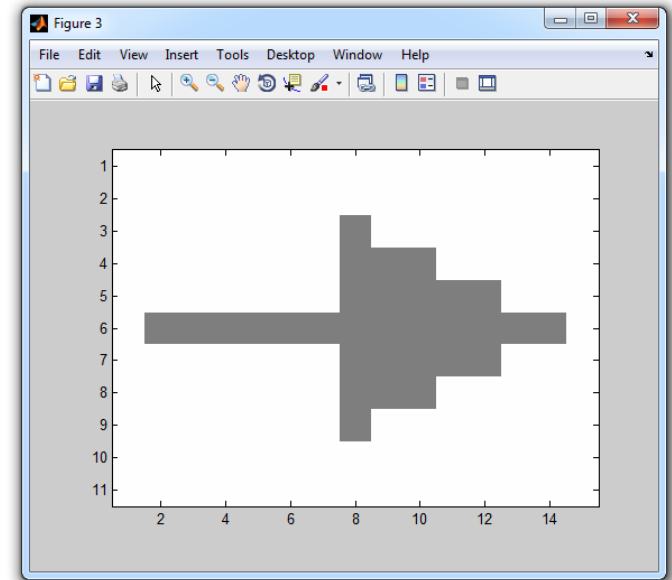
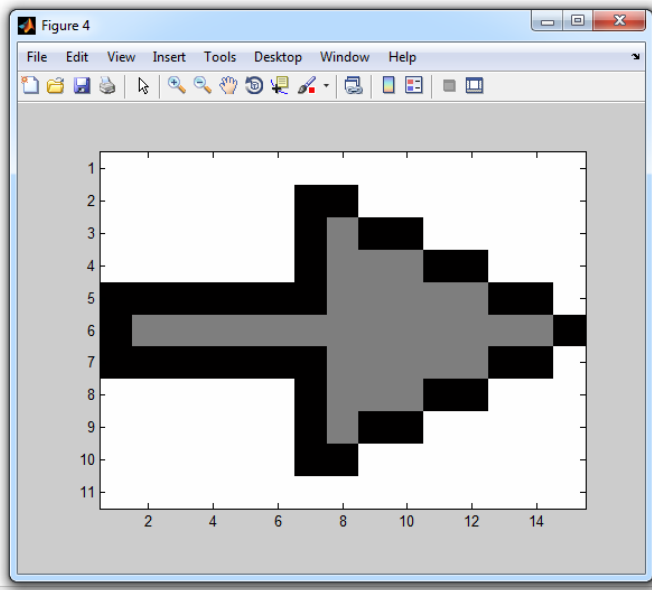
```
y = exp(x);
```

```
plot(x, y);
```



Práce s maticemi

- Vymažte šipce (sipka.pgm) černou konturu



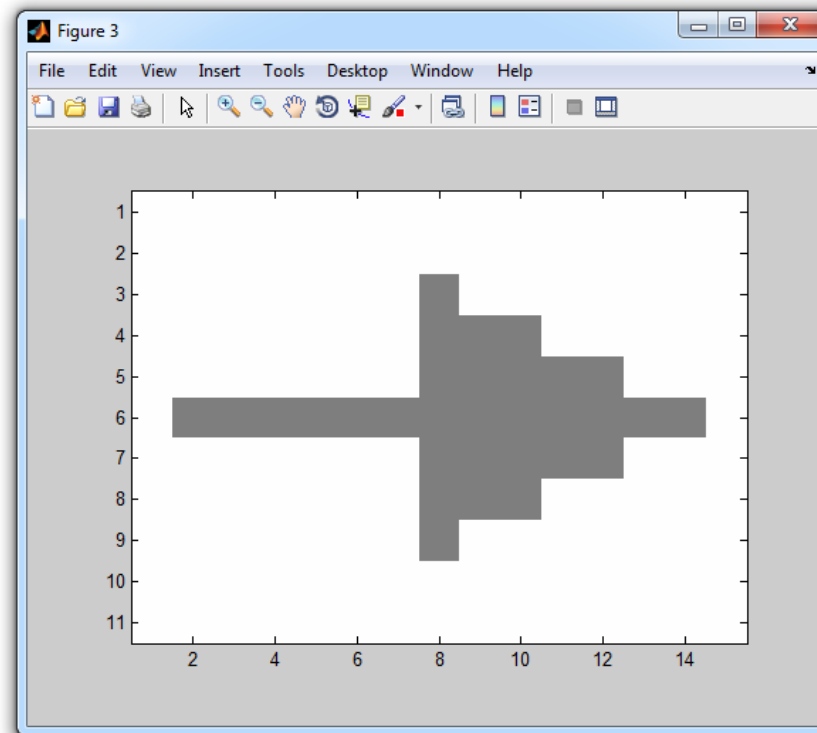
Práce s maticemi

- Vymažte šipce (sipka.pgm) černou konturu

```
S=double(imread('sipka.pgm'));
```

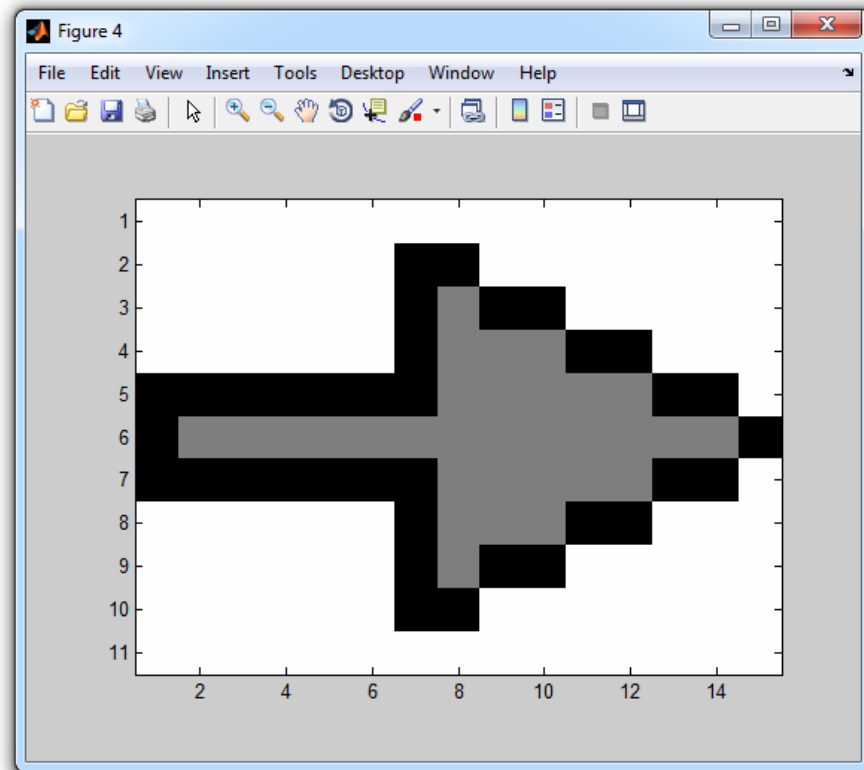
```
S(S==0)=255;
```

```
zobraz(S);
```



Práce s maticemi

- Určete délku kontury šipky (počet pixelů)

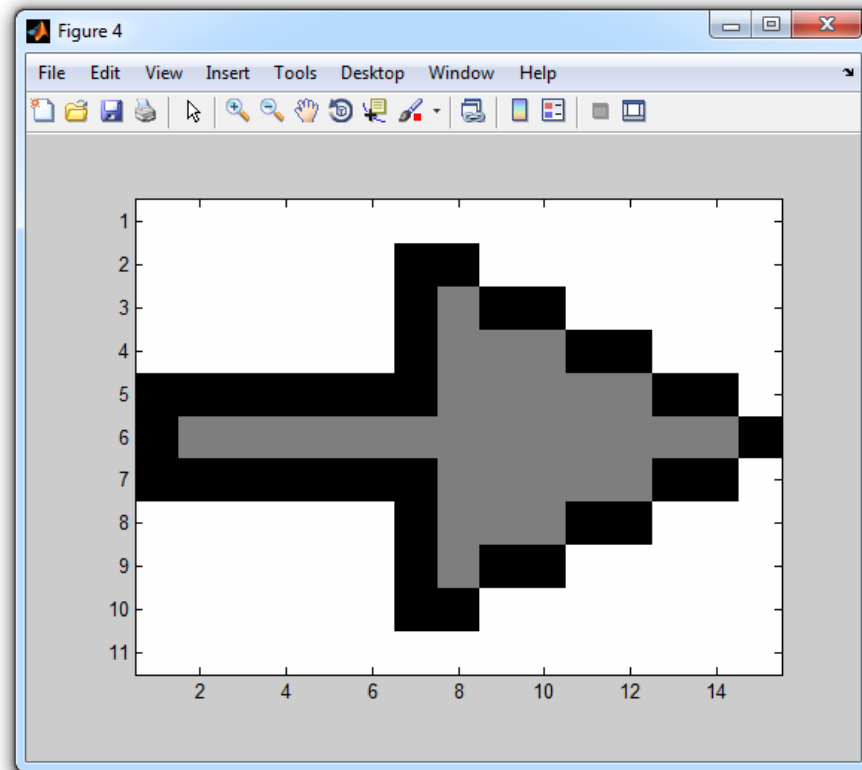


Práce s maticemi

- Určete délku kontury šipky (počet pixelů)

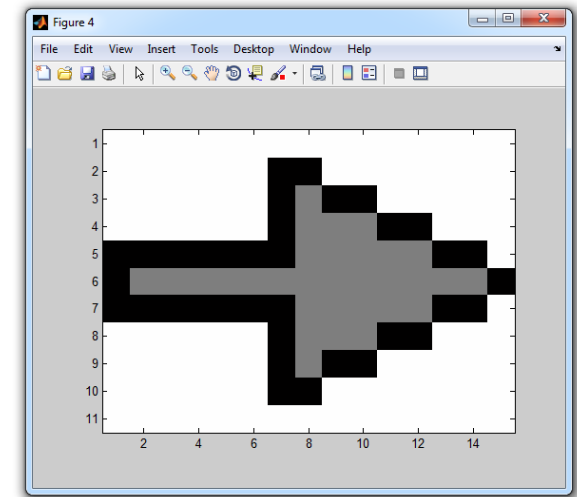
```
S = double(imread('sipka.pgm'));  
sum(sum(S == 0));
```

```
ans = 36
```



Práce s maticemi

- najděte binární i grayscale těžiště šipky
 - binární:
 - všechny barvy mají váhu 1 kromě bílé (hodnota 255), kterou beru jako pozadí a má váhu 0
 - grayscale:
 - hodnota intenzity je jeho váha
 - tedy černá (hodnota 0) má váhu 0
 - bílá (hodnota 255) má váhu 255



- *help: find, mean*

Práce s maticemi

- **binární:** `[x, y] = [8.0896, 6]`

```
[y, x] = find(I < 255);
```

```
mean([x, y]);
```

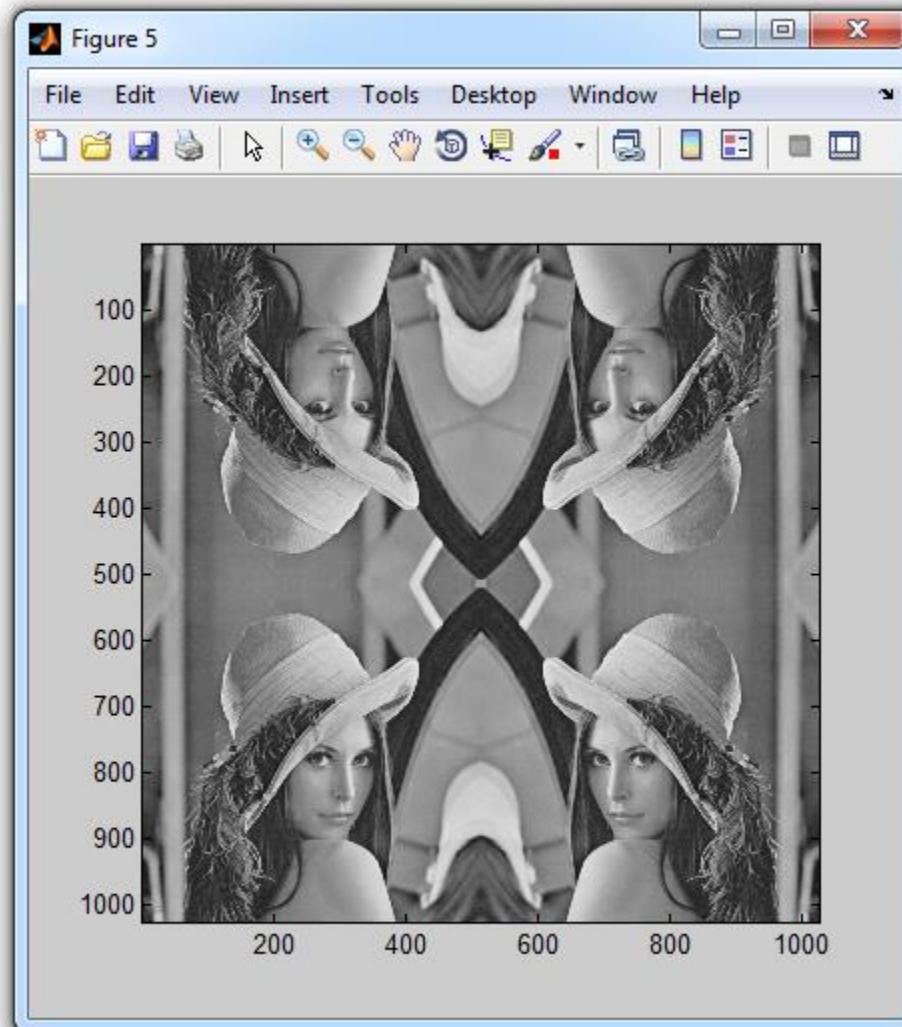
- **grayscale:** `[x, y] = [8.0613, 6]`

```
x = sum(I,1) * [1:size(I,2)]' / sum(I(:));
```

```
y = [1:size(I,1)] * sum(I,2) / sum(I(:));
```


Práce s maticemi

- Vytvořte tento obraz bez použití `flipud` a `fliplr`



Práce s maticemi

```
L=double(imread('lena.png'));
```

```
L = rgb2grayscale(L);
```

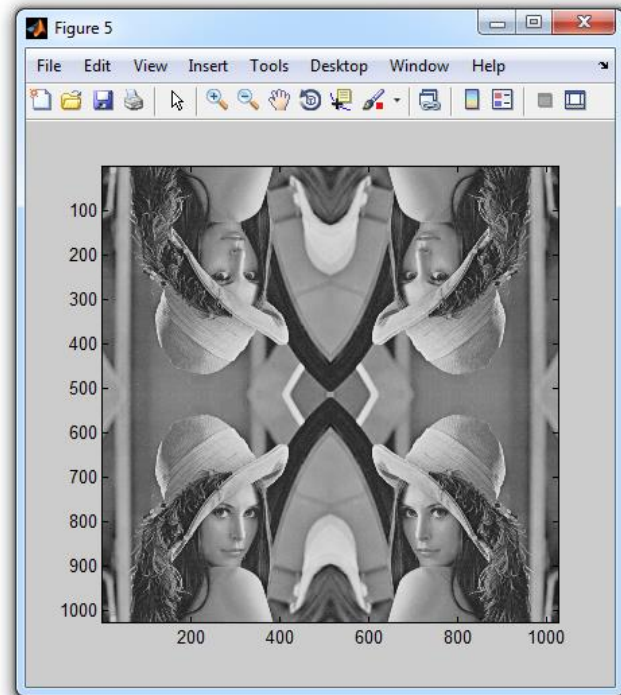
```
L1=(L(end:-1:1,end:-1:1));
```

```
L2=(L(end:-1:1,:));
```

```
L3=L;
```

```
L4=(L(:,end:-1:1));
```

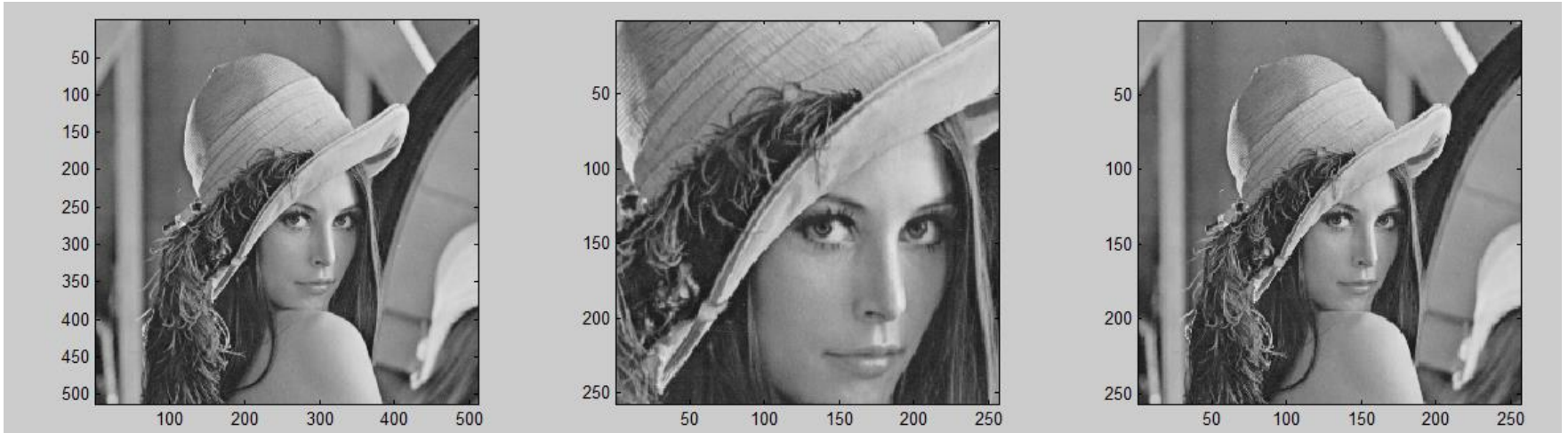
```
zobr([L2 L1;L3 L4]);
```



Práce s maticemi

- Napište funkci:

```
function [croppI, subI]=croppSub (I)
% croppI ...výřez středu poloviční velikosti I
% subI ...I zmenšený na půlku (bez interpolace)
...
end
```



help: size, round

Práce s maticemi

- Napište funkci:

```
function [croppI, subI]=croppSub(I)
% croppI...výřez středu poloviční velikosti I
% subI ...I zmenšený na půlku
S = round(size(I)/4);
croppI=I(S(1):3*S(1),S(2):3*S(2));
subI=I(1:2:end,1:2:end);
end
```

Práce s maticemi

- Napište funkci:

```
function [Y]=jaskontrast(I, jas, kontrast)
```

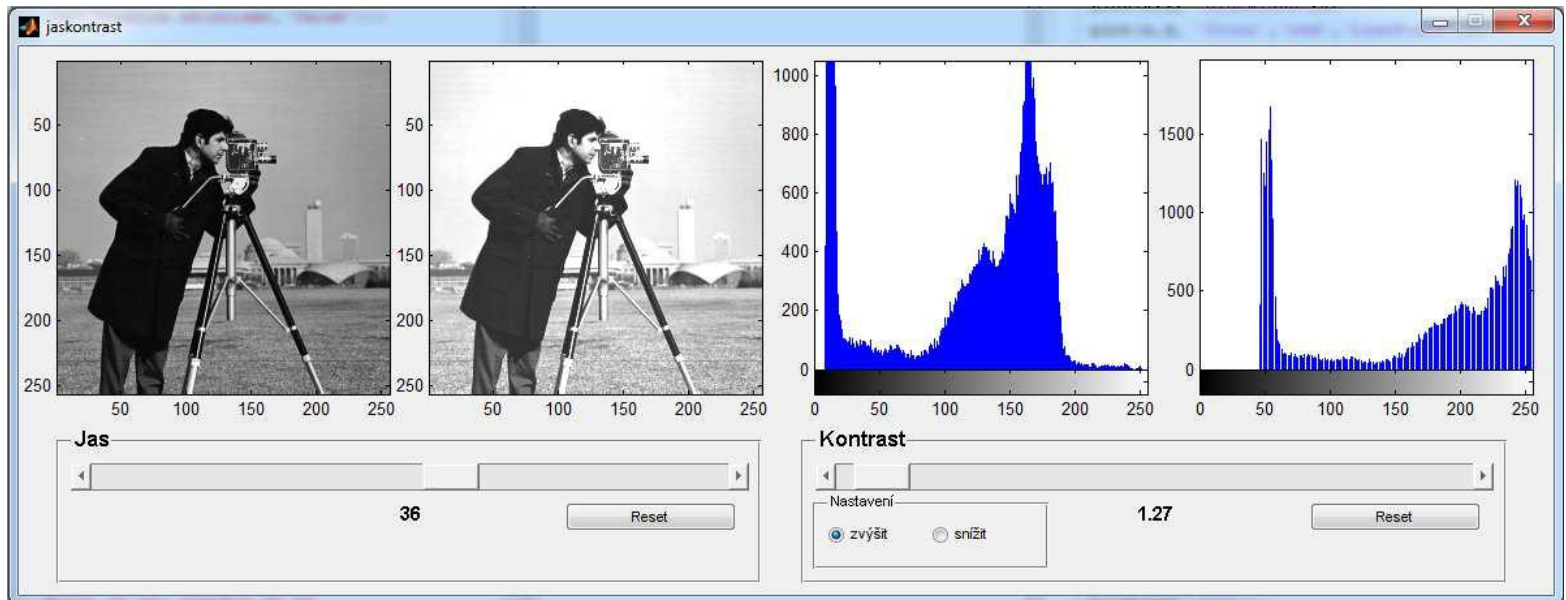
```
% I...vstupní obrázek
```

```
% jas...hodnota zvýšení/snížení jasu
```

```
% kontrast...hodnota zvýšení/snížení kontrastu
```

```
...
```

```
end
```



help: hist, imhist

Práce s maticemi

- Napište funkci:

```
function [Y]=jaskontrast(I, jas, kontrast)
% I...vstupní obrázek
% jas...hodnota zvýšení/snížení jasu
% kontrast...hodnota zvýšení/snížení kontrastu
Y=I * kontrast+jas;
end
```

Práce s obrázkem

- načtete obrázek **lena.png**
- udělejte z něho jednokanálový binární obrázek (např. `Img < 128`)
- uložte ho jako `*.bmp` obrázek



`help: imwrite`

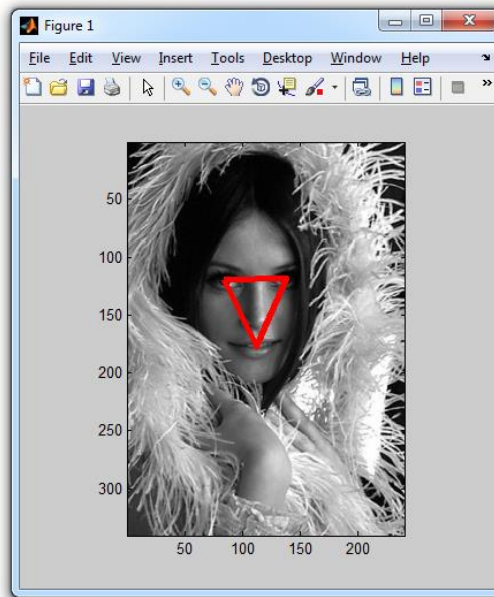
Práce s obrázkem

- načtete obrázek **lena.png**
- udělejte z něho jednokanálový binární obrázek (např. $\text{Img} < 128$)
- uložte ho jako *.bmp obrázek

```
Img = double(imread('lena.png'));  
Img = rgb2grayscale(Img);  
Img = (Img>128)*255;  
imwrite(Img, 'lean_bin.bmp', 'bmp');
```


Práce s obrázkem

- Pomocí výběru bodů v obrázku **lena2.png** nakreslete červený trojúhelník, jehož body budou: levé oko, pravé oko, střed pusy
- síla čáry bude 4px



help: ginput, line, imsave

Práce s obrázkem

- Pomocí výběru bodů v obrázku **lena2.png** nakreslete červený trojúhelník, jehož body budou: levé oko, pravé oko, střed pusy
- síla čáry bude 4px

```
Img = imread('lena2.png');  
zobr(Img);  
[x, y] = ginput(3);  
x = [x; x(1)];  
y = [y; y(1)];  
line(x, y, 'Color', 'r', 'LineWidth', 4);
```

Hromadné zpracování souborů

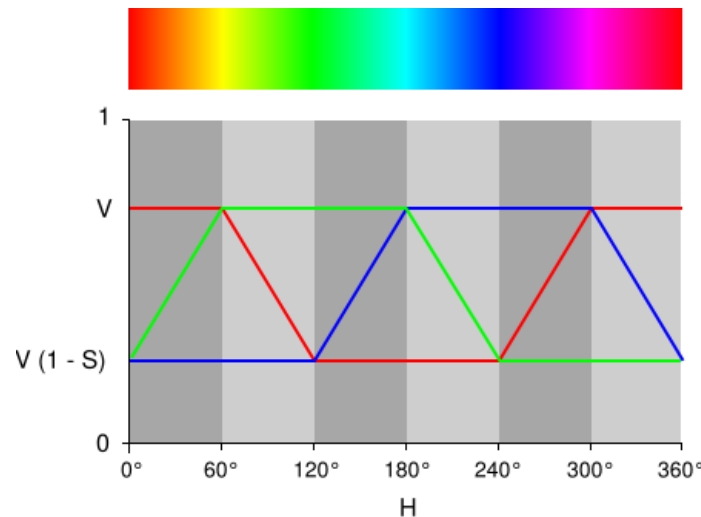
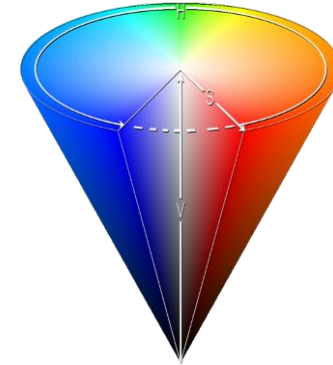
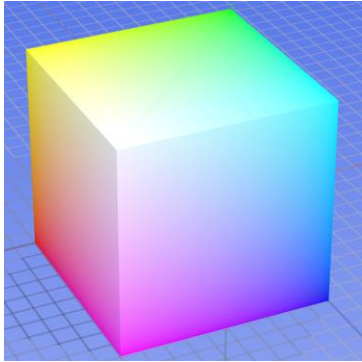
- napište skript, kterému předáte jméno složky a on vám přebarví ve všech *.png obrázcích všechny zelené pixely na bílo (255,255,255) a poté obrázky uloží do podsložky ('\\nG') jako „***name_of_file-noG.png***“
- Vyzkoušejte na složce 'flags'



- `help: dir, fullfile, exist, mkdir, length, fprintf, imwrite, rgb2hsv`

Hromadné zpracování souborů

- Náповěda z teorie barevných prostorů:
 - je dobré obrázek převést z RGB do HSV:



- **Hue H (barva)** ($0^\circ - 360^\circ$)
- **Saturation S (syťost)** ($0\% - 100\%$)
- **Value V (jas)** ($0\% - 100\%$)

Hromadné zpracování souborů

```
function [Y] = noGreen(Img)
Img_hsv = rgb2hsv(Img);

Img_bin = Img_hsv(:,:,1) > 0.1944 & Img_hsv(:,:,1) < 0.4722;

R = I(:,:,1);
G = I(:,:,2);
B = I(:,:,3);

R(Img_bin) = 255;
G(Img_bin) = 255;
B(Img_bin) = 255;

Y(:,:,1) = R;
Y(:,:,2) = G;
Y(:,:,3) = B;
end
```

Hromadné zpracování souborů

```
function del_green(folder)
    train_images = dir(fullfile(folder, '*.png'));

    my_path = [folder '\\ 'nG'];
    if (exist(my_path, 'dir') ~= 2)
        mkdir(my_path);
    end

    for i = 1:length(train_images)
        i
        current_filename = train_images(i).name;
        fprintf('Processing %s\\%s.....\n', folder, current_filename);
        Img = double(imread([folder '\\ ' current_filename]));
        Img_nG = noGreen(Img);
        imwrite(Img_nG./255, [folder '\\ 'nG\\' current_filename], 'png');
    end
end
```

Práce s videem

- ve filmové praxi se už několik let používá tzv. Green-Screen technika:
 - nasnímá se objekt (např. reportér) před zeleným plátnem a poté se všechny zelené pixely nahradí obrázkem v pozadí



... takhle to vypadá, když si rosnička oblékne zelené šaty ...

zdroj:

<http://www.mirror.co.uk/news/world-news/weather-forecaster-wardrobe-malfunction-presenter-825367>

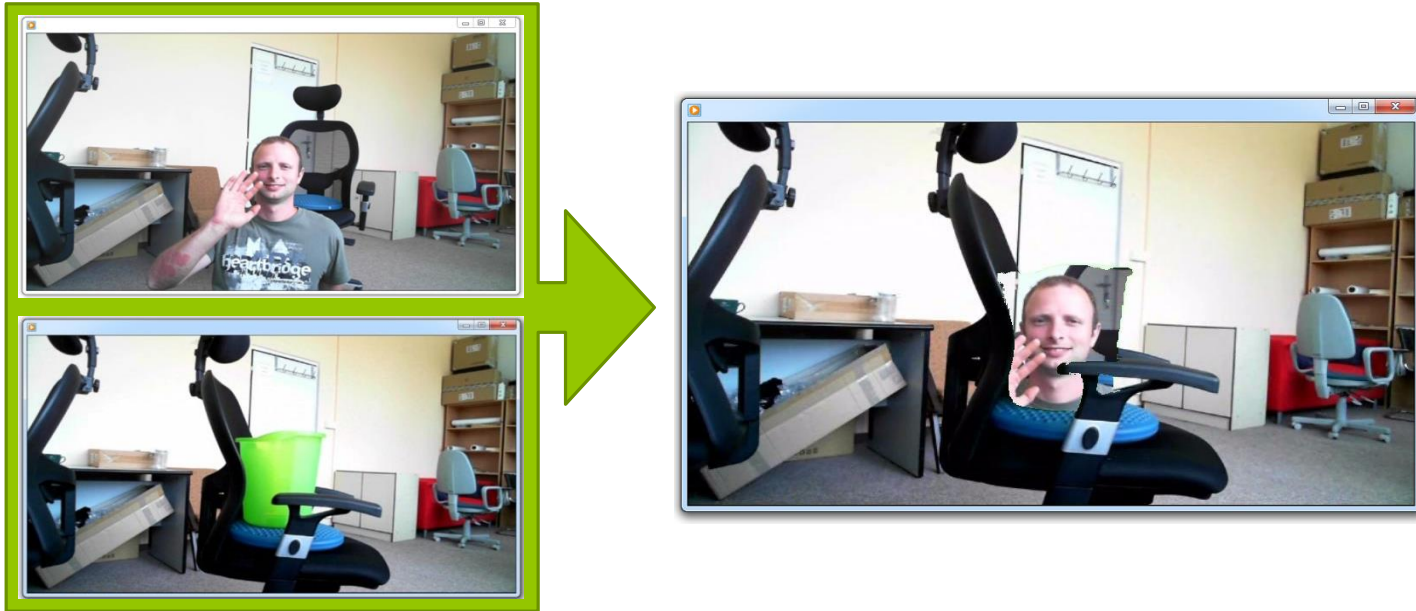
Práce s videem

• napište skript, kterému předáte dvě videa:

- green-screen video (**vicka.mp4**, **kos.wmv**)
- background video (**ulice.mp4**, **hlava.wmv**)

a on vám uloží video, které bude kombinací těchto dvou

• je potřeba mít nainstalované kodeky pro mp4 !!!



• *help: VideoReader, VideoWriter*

Práce s videem

```
function [Img] = switch_geen_px(Img, Img_back)
    R = Img(:,:,1);
    G = Img(:,:,2);
    B = Img(:,:,3);
    R_back = Img_back(:,:,1);
    G_back = Img_back(:,:,2);
    B_back = Img_back(:,:,3);

    Img_hsv = rgb2hsv(Img);
    myBin = Img_hsv(:,:,1) > 0.1944 & Img_hsv(:,:,1) < 0.4722 ...
        & Img_hsv(:,:,2) > 0.20 & Img_hsv(:,:,3) > 0.15;

    R(myBin) = R_back(myBin);
    G(myBin) = G_back(myBin);
    B(myBin) = B_back(myBin);

    Img(:,:,1) = R;
    Img(:,:,2) = G;
    Img(:,:,3) = B;

end
```

Práce s videem

```
function [] = greenScreen(name_in1, name_in2, name_out)
    xyloObj = VideoReader(name_in1);
    xyloObj_back = VideoReader(name_in2);
    vidObj = VideoWriter(name_out);
    open(vidObj);

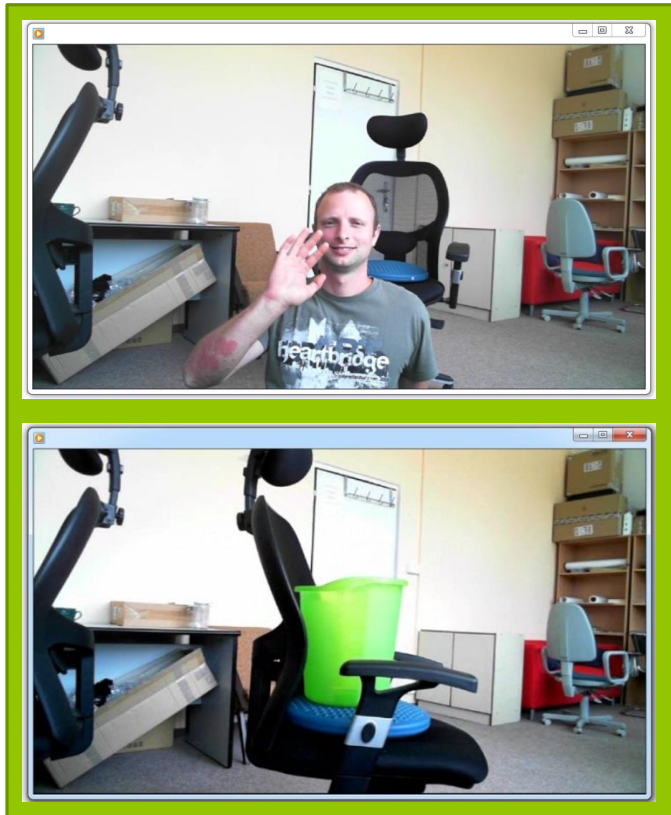
    nFrames = xyloObj.NumberOfFrames;
    nFrames_back = xyloObj_back.NumberOfFrames;

    if(nFrames < nFrames_back)
        nFrames_all = nFrames;
    else
        nFrames_all = nFrames_back;
    end

    for k = 1 : nFrames_all
        clc;
        fprintf('Done: %2.2f %% \n', 100*k/nFrames_all);
        Img = read(xyloObj, k);
        Img_back = read(xyloObj_back, k);
        Img = switch_green_px(Img, Img_back);
        writeVideo(vidObj, Img);
    end
    close(vidObj);
    fprintf('\n');
end
```

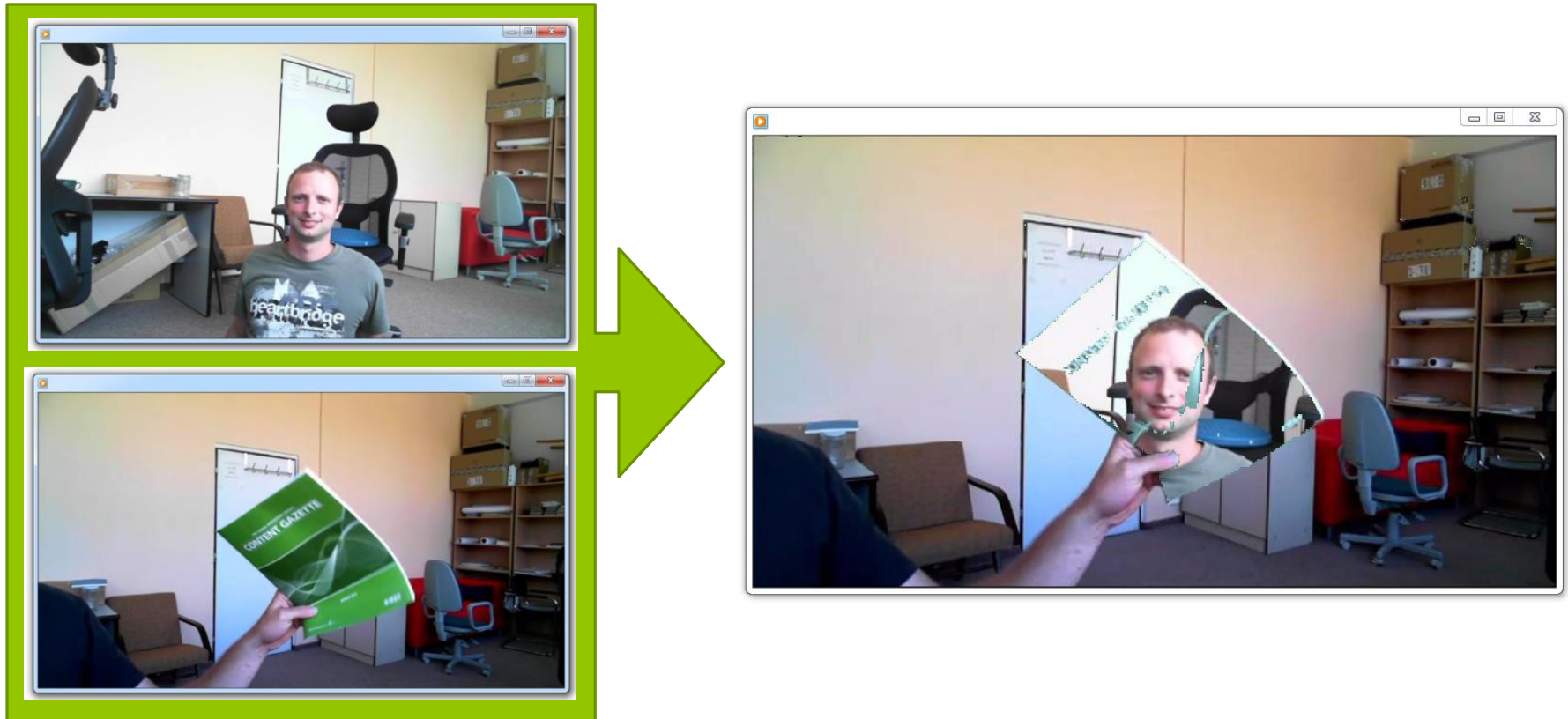
Práce s videem

```
greenScreen ('video\kos.wmv', ...  
            'video\hlava.wmv', 'video\output');
```



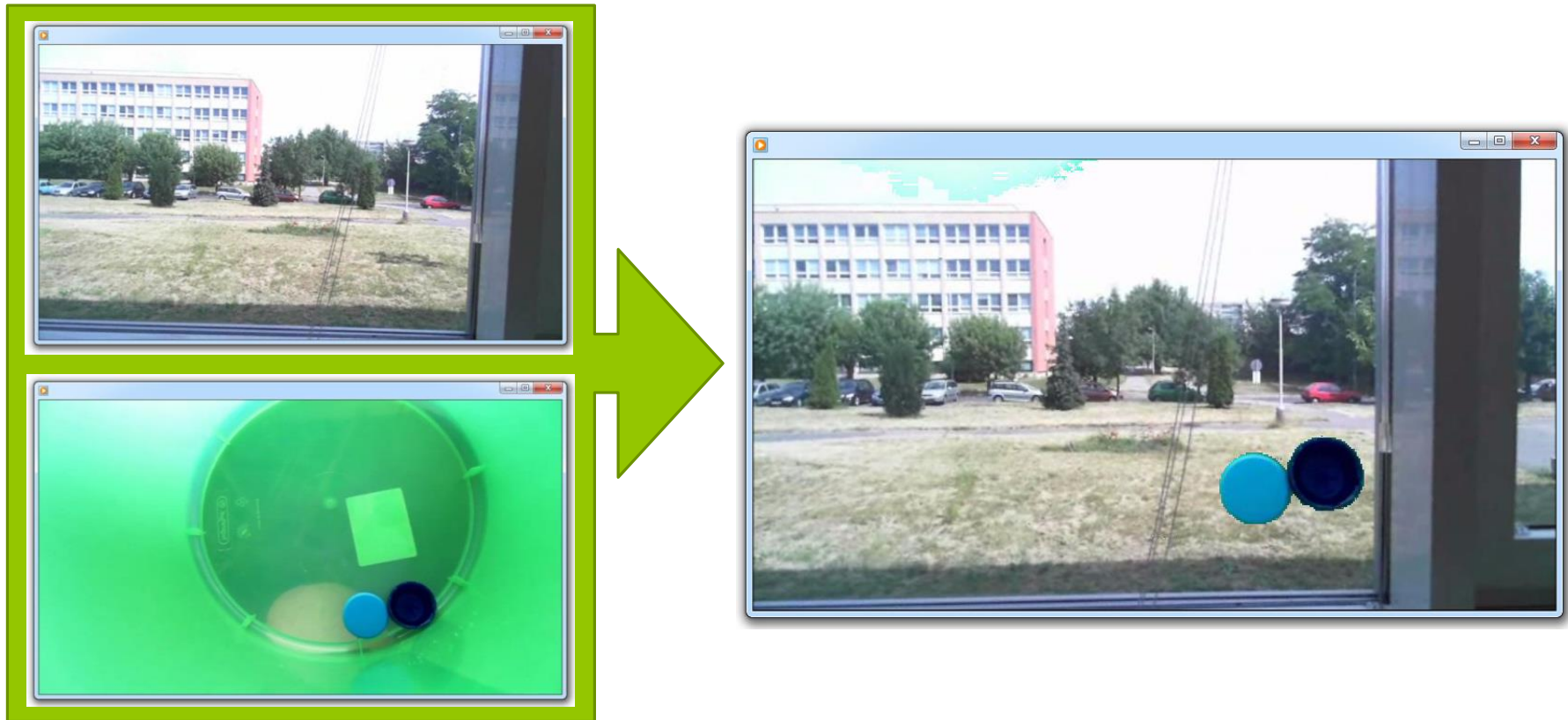
Práce s videem

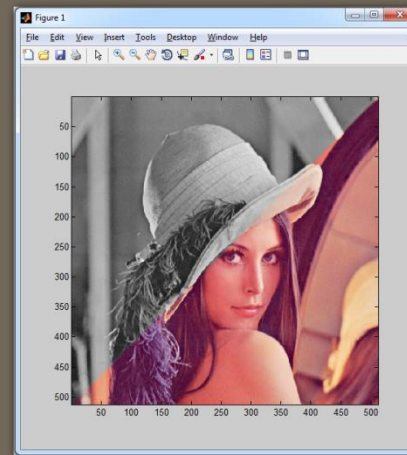
```
greenScreen ('video\sesit.mp4', ...  
            'video\hlava.wmv', 'video\output2');
```



Práce s videem

```
greenScreen ('video\vicka.mp4', ...  
            'video\ulice.wmv', 'video\output3');
```





Děkuji za
pozornost