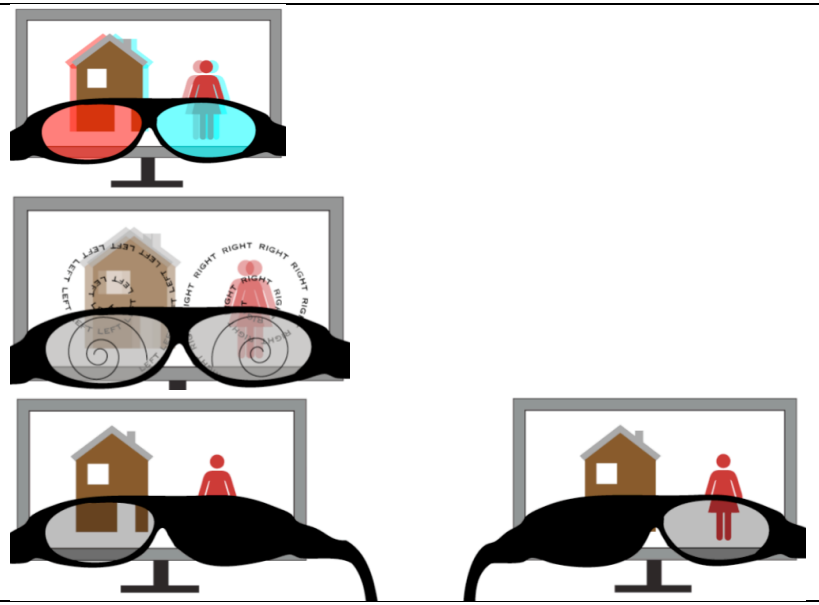
Čo vnímame klasicky?

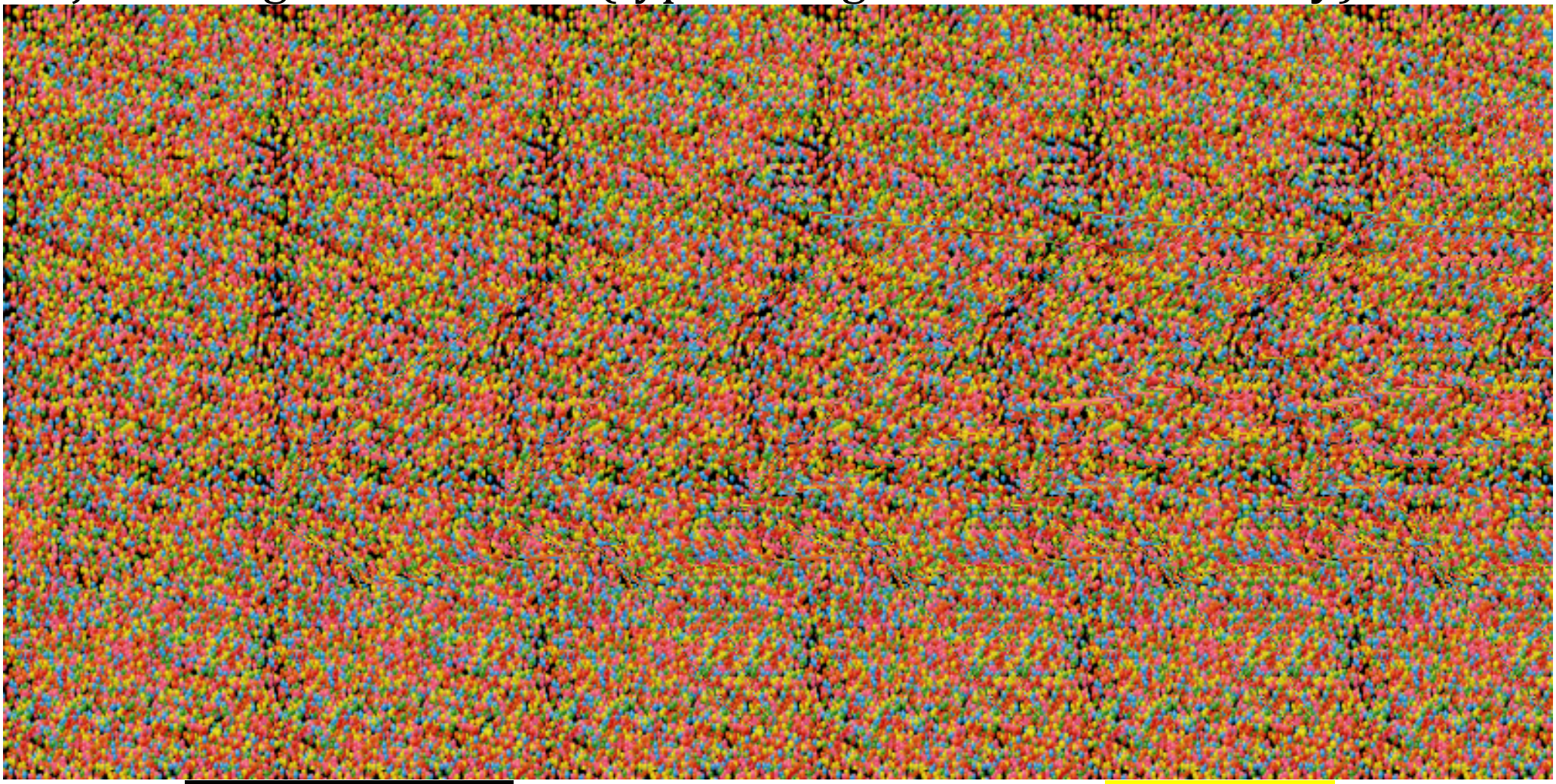
- 3D scénu – reflexne
- Ako to vieme popísať? Ako:
  - 2D signál
  - Stereoskopický – 2x2D signál (pre každé oko zvlášť obraz)
    - Máme normálny papier
      - Obrazy zmixované cez seba -> stereogram!
    - Máme špeciálne okuliare/premietačku

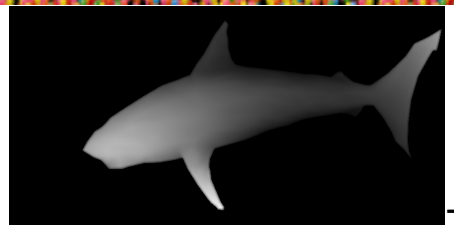
- Pasívne okuliare
  - Červené/zelené sklíčko (anaglyf)
  - Kruhovo polarizované okuliare (každé sklíčko opačným smerom) / špeciálna premietačka
- Aktívne okuliare
  - 3SD – aktívne okuliare (s batériou) a LCD displejmi, ktoré synchronizovane púšťajú obraz raz pre jedno a raz pre druhé oko



    - Prenos zvyčajne SBS (side-by-side) - dva obrázky vedľa seba, je na zariadení, ako ich bude renderovať

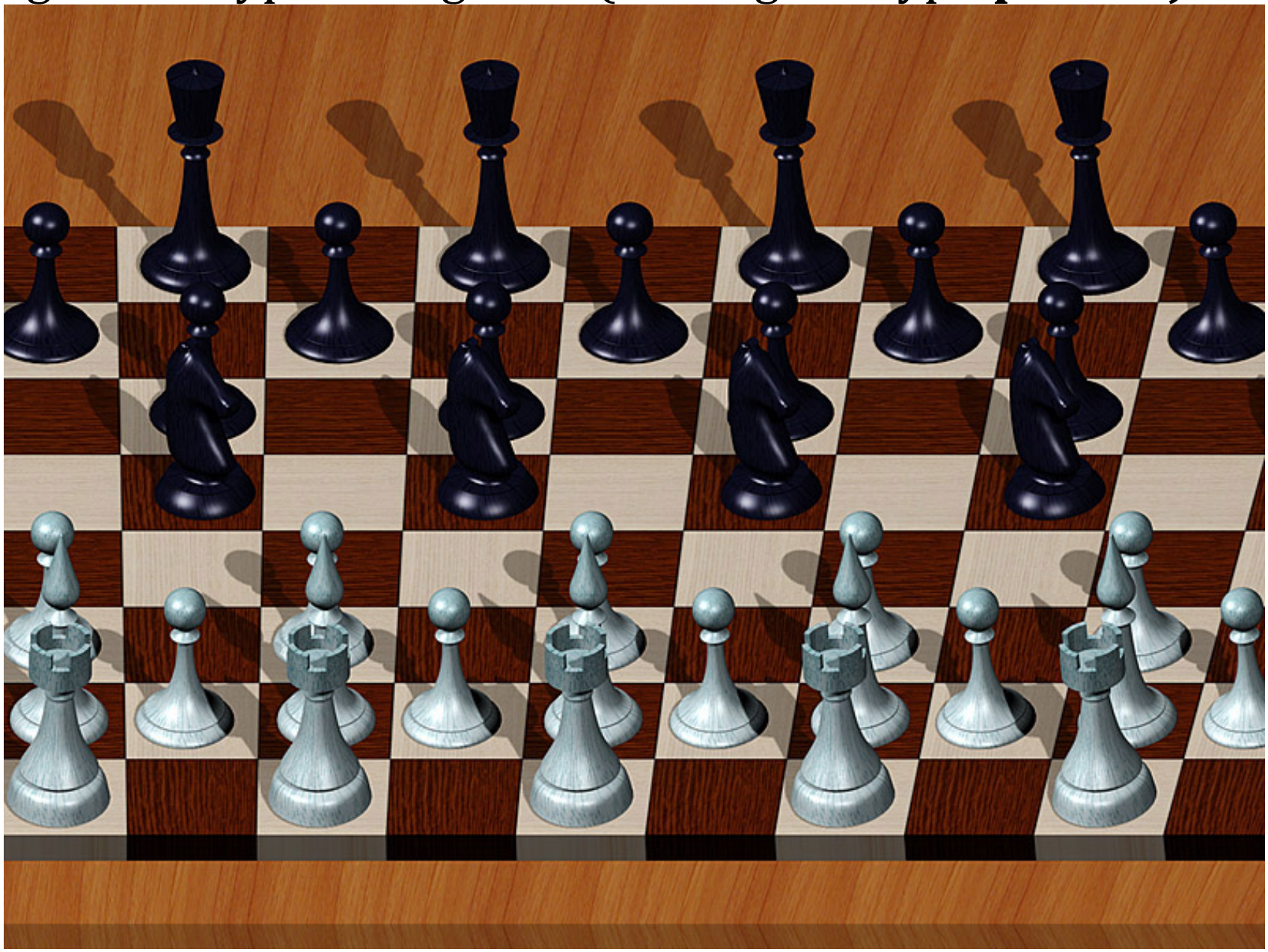
Čo je stereogram? Príklad (typ stereogramu **náhodné body**):





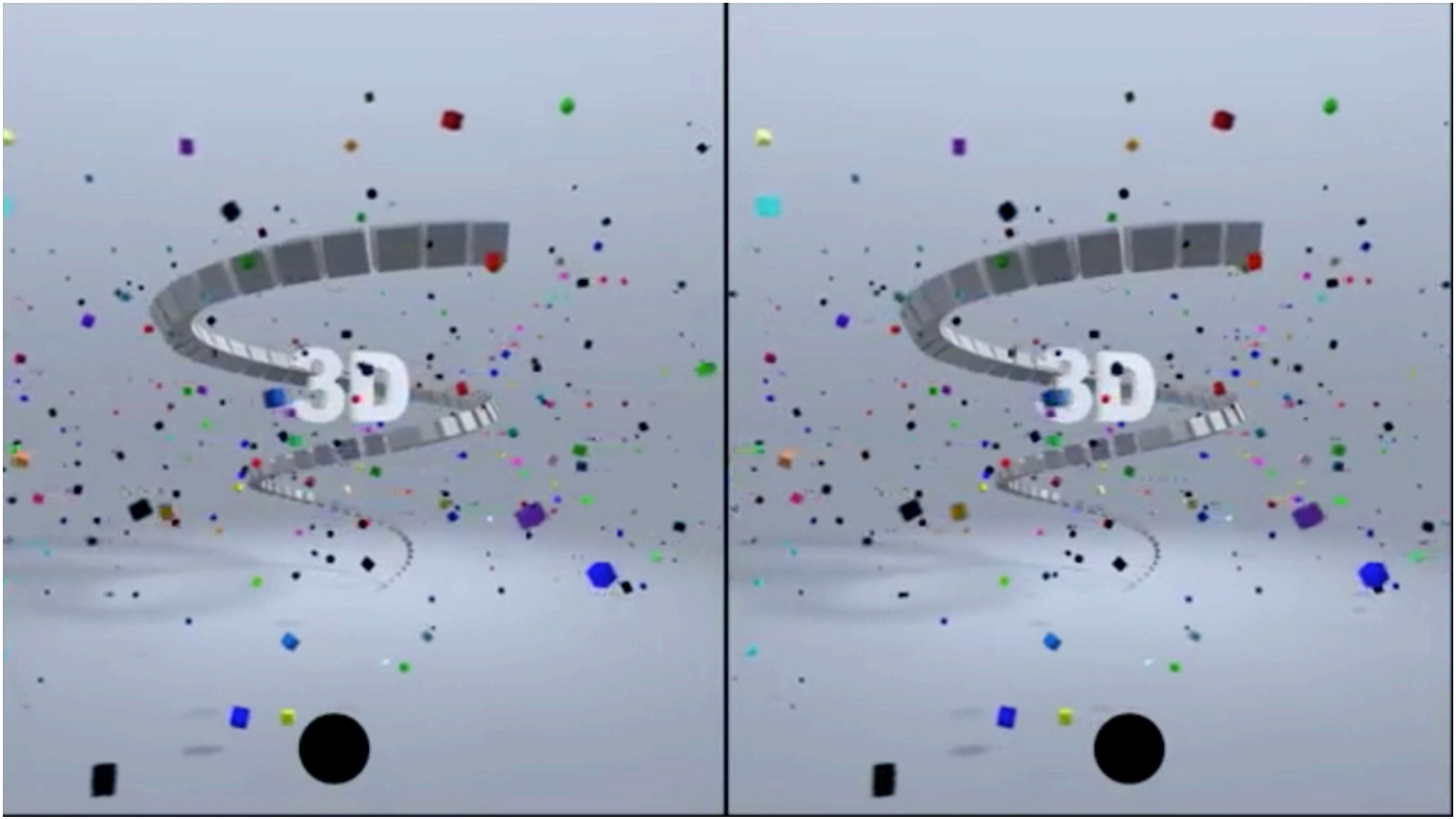
+ príklad že existujú aj podobné videá [10_snowman]

Stereogram2 – typ stereogramu (stereogram typu **pole objektov**)



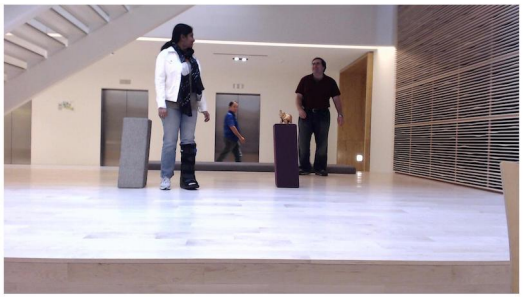
3D obrázky (aj videá) – pre každé oko je zdroj zvlášť (side by side)

[https://www.youtube.com/watch?v=zBa-bCxsZDk]



Anaglyfy a matlab

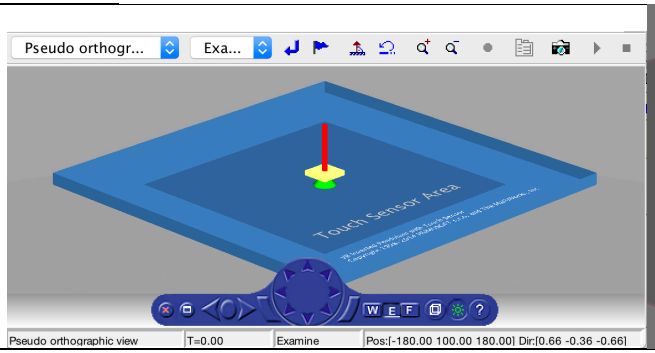
openExample('vision/CreateA3DStereoDisplayExample')

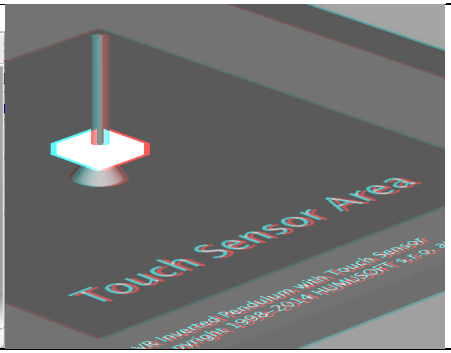


```
load('webcamsSceneReconstruction.mat')
I1 = imread('sceneReconstructionLeft.jpg');
I2 = imread('sceneReconstructionRight.jpg');
 [J1, J2] = rectifyStereoImages(I1, I2, stereoParams);
A = stereoAnaglyph(J1, J2);
figure; imshow(A);
```

VR virtuálna realita:

```
myworld = vrworld('vrpend.wrl')
open(myworld)
view(myworld)

Rendering-> Stereo 3D -> Anaglyph

V prehliadači: view(myworld,'-web')
```



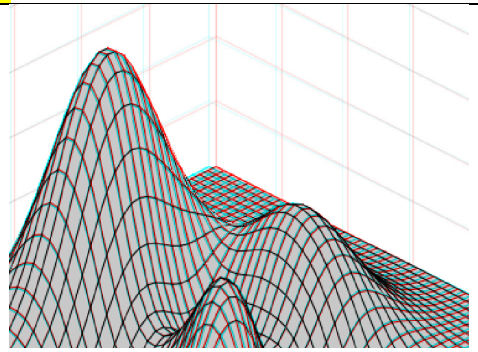


Grafy a anaglyfy

Napr. animované: https://www.mathworks.com/matlabcentral/fileexchange/31133-plot3anianaglyph	Použitie: plot3AniAnaglyphDemos(0) ...

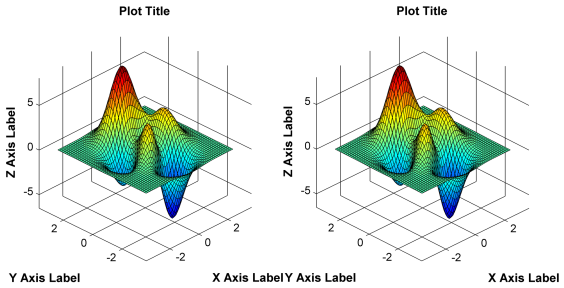
Skonštruovanie vlastného grafu (anaglyf) [2013] 01_graph_ana.m

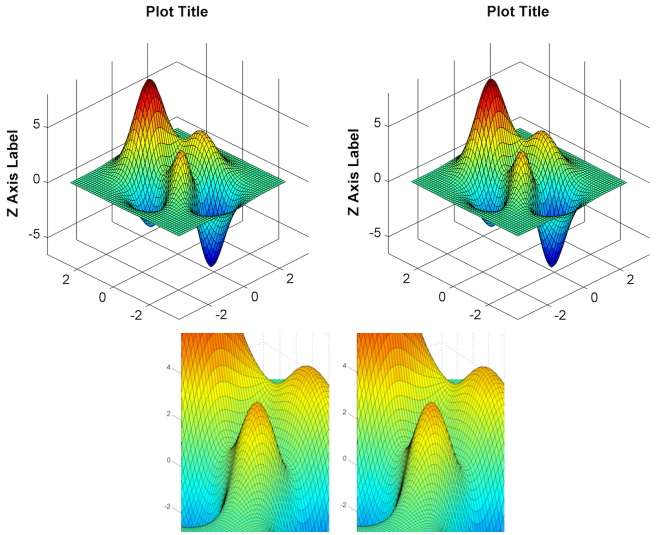
```
[X,Y,Z] = peaks(50) ;       % Create some dummy data.
 parallaxAngle = 0.5 ;       % You can change this to suit yourself.
 cameraAnchor = [  max(X(:))  max(Y(:))  min(Z(:)) ] ;

 leftImage = figure;
 surf(X,Y,Z,'FaceColor',[0.8 0.8 0.8]) % Create the surface figure.
 axis vis3d ;                    % Lock the aspect ration for rotation.
 camtarget(gca,[cameraAnchor])  % Point the camera at the anchor point.
 view((-45 - parallaxAngle),30) ; % Rotate the figre to the correct angle.
 title  ('Plot Title', 'FontWeight','Bold','FontSize',11) % Add labels...
 xlabel ('X Axis Label', 'FontWeight','Bold','FontSize',11)
 ylabel ('Y Axis Label', 'FontWeight','Bold','FontSize',11)
 zlabel ('Z Axis Label', 'FontWeight','Bold','FontSize',11)
 grid on ;                       % Switch on the figure grid, and
 set(gca,'GridLineStyle','-')   % set the grid to solid lines.
  rightImage = figure;
 surf(X,Y,Z,'FaceColor',[0.8 0.8 0.8]) % Create the surface figure.
 axis vis3d ;                    % Lock the aspect ration for rotation.
 camtarget(gca,[cameraAnchor])  % Point the camera at the anchor point.
 view((-45 + parallaxAngle),30) ; % Rotate the figre to the correct angle.
 title  ('Plot Title', 'FontWeight','Bold','FontSize',11) % Add labels...
 xlabel ('X Axis Label', 'FontWeight','Bold','FontSize',11)
 ylabel ('Y Axis Label', 'FontWeight','Bold','FontSize',11)
 zlabel ('Z Axis Label', 'FontWeight','Bold','FontSize',11)
 grid on ;                       % Switch on the figure grid, and
 set(gca,'GridLineStyle','-')   % set the grid to solid lines.
 print(leftImage ,'-dtiffn','-painters', 'Left_Eye.tif')
 print(rightImage,'-dtiffn','-painters','Right_Eye.tif')
  leftEyeImage  = imread('Left_Eye.tif')  ; % Load the left eye image.
 rightEyeImage = imread('Right_Eye.tif') ; % Load the right eye image.
 leftEyeImage(:,:,2:3) = 0 ;               % Removes green and blue from the left eye image.
 rightEyeImage(:,:,1)  = 0 ;               % Removes red from the right eye image.
 anaglyph = leftEyeImage + rightEyeImage ; % Combines the two to produce the finished anaglyph.
 imshow(anaglyph,'border','tight') ;       % Show the anaglyph image with no padding.
 print(gcf,'-dtiffn','-painters','Anaglyph.tif')  % Save the anaglyph image.
```



Skonštruovanie vlastného 3D SBS grafu 02_graph_sbs.m

```
outImgName='_my3D_HiDpiColor.png';
 [X,Y,Z] = peaks(50) ;       % Create some dummy data.
 parallaxAngle = 0.5 ;       % You can change this to suit yourself.
 cameraAnchor = [  max(X(:))  max(Y(:))  min(Z(:)) ] ;
 leftImage = subplot(1,2,1)
 surf(X,Y,Z) % Create the surface figure.
 axis vis3d ;                % Lock the aspect ration for rotation.
 camtarget(gca,[cameraAnchor])  % Point the camera at the anchor point.
 view((-45 - parallaxAngle),30) ; % Rotate the figre to the correct angle.
 title  ('Plot Title', 'FontWeight','Bold','FontSize',11) % Add labels...
 xlabel ('X Axis Label', 'FontWeight','Bold','FontSize',11)
 ylabel ('Y Axis Label', 'FontWeight','Bold','FontSize',11)
 zlabel ('Z Axis Label', 'FontWeight','Bold','FontSize',11)
 grid on ;                   % Switch on the figure grid, and
 set(gca,'GridLineStyle','-')   % set the grid to solid lines.
rightImage = subplot(1,2,2)
 surf(X,Y,Z) % Create the surface figure.
 axis vis3d ;                % Lock the aspect ration for rotation.
 camtarget(gca,[cameraAnchor])  % Point the camera at the anchor point.
 view((-45 + parallaxAngle),30) ; % Rotate the figre to the correct angle.
 title  ('Plot Title', 'FontWeight','Bold','FontSize',11) % Add labels...
 xlabel ('X Axis Label', 'FontWeight','Bold','FontSize',11)
 ylabel ('Y Axis Label', 'FontWeight','Bold','FontSize',11)
 zlabel ('Z Axis Label', 'FontWeight','Bold','FontSize',11)
 grid on ;                   % Switch on the figure grid, and
 set(gca,'GridLineStyle','-')   % set the grid to solid lines.

print(gcf,outImgName,'-dpng','-r600');
```





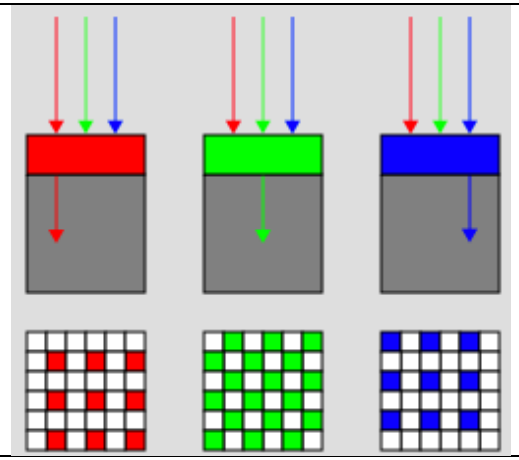
Druhy obrazov

- Farebnosť / hĺbka informácie
  - Farebné -> v akom „farebnom priestore“
    - 256 farieb
    - 256*256*256 farebné (24 bitov)
  - Šedotónové
    - 8 bitov (bežne), 12 bitov (medicínske obrazy)
  - Čierno/biele (1 bitová farebná hĺbka)
  - Z/bez ditheringu (čo to je? Pamätáte na ADSS1)
- Z hľadiska tvorby obrazu sa rozlišuje (prienik s počítačovou grafikou)
  - zosnímané (cez nejaký snímač)
  - vytvorené / upravené (v počítači)
    - Raster/vektor
    - 2D/3D
      - 2D grafika
        - vrstvy ich priehľadnosť, alfa kanál
      - 3D grafika
        - scény, osvetlenie, priehľadnosť materiály, ...
    - 3D obraz – rezy mozgu, tkanív, ...

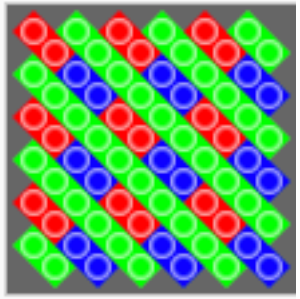
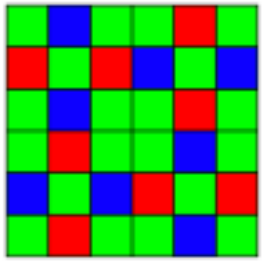
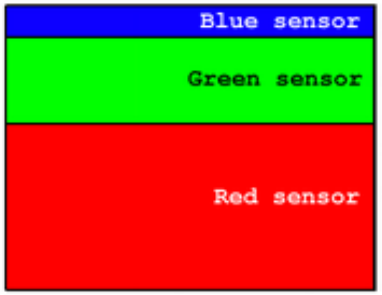
Snímač obrazu – fotoaparát

- Obrazové senzory
  - v súčasnosti najma CMOS (Complementary Metal Oxid Seminductor)
    - začína medzi fotografmi trend „nostalgického návratu k CCD“ – lebo to sníma „ináč“

  - pred senzorom sú farebné filtre, CFA (color filter array) najpoužívanejšie Bayerove: chýbajúce farebné vzorky sú interpolované (algoritmy sú individuálne)
  - každý bod má informáciu iba o jednej farbe, zelených je 50%, červenýcha modrých po 25%

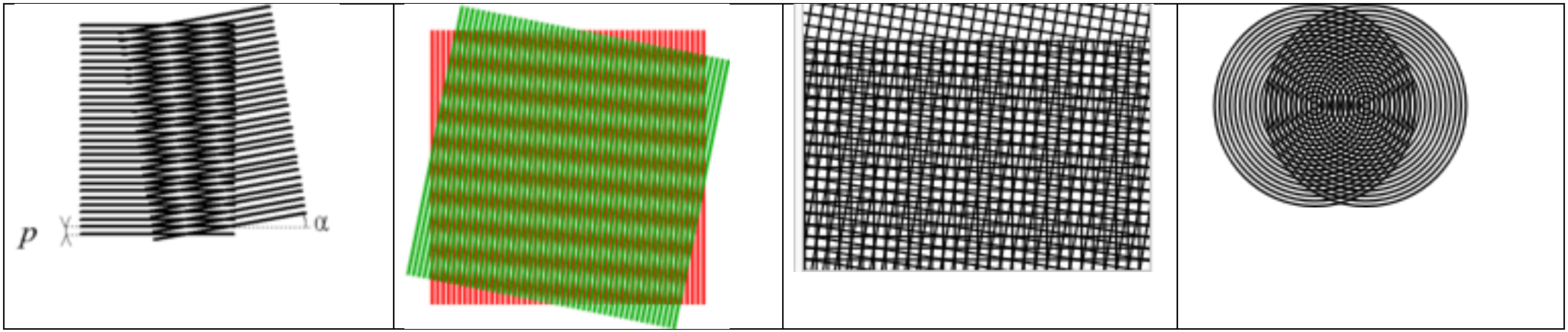


  - Pred Bayerovými filttrami bývajú ešte OLPF (Opticel Low-pass filter) – ktoré robia rozostrenie každého detailu, ktorý je jemnejší ako rozlíšenie senzora (obmedzuje moaré)
  - Iné druhy filtrov (pozn. **Foveon** sníma v každom bode všetky 3 farby) ↓

Kodak RGBW	FUjifilme EXR	Fujifilm X-trans	**Foveon**
			

Čo je moiré/moaré?

- Rušivý efekt, ktorý vzniká keď pravidelný obrazec bodov snímača (alebo bodov displeja) interferuje s nejakým pravidelným vzorom na obrázku
- Rôzne druhy moaré
    - Napr. paralelné čiary, keď sa otočia o malý uhol a preložia s neotočenými, podobne mriežky, kružnice, ...



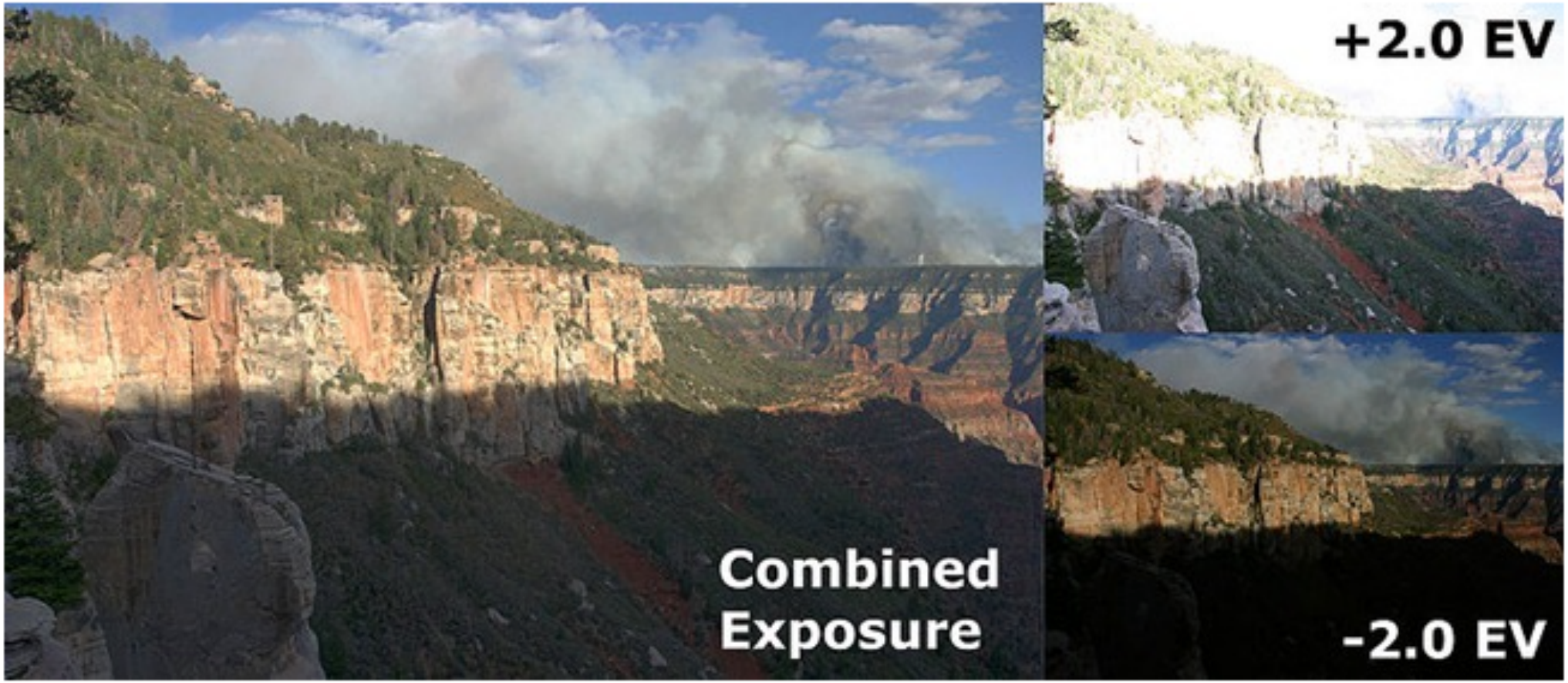
- Bayerove filtre ho tvoria
- Foveon ho netvorí

RAW format

- Formát údajov priamo po nasnímaní z jednotlivých buniek
- Napr Canon má vlastné formáty CRW (http://xyrion.org/ciff/) a CR2 (http://lclevy.free.fr/cr2/)
- Jednotlivé R, G, B zložky z Bayerových filtrov sú prenášané ako 12/14 bitové ...
- Pri „vyvolávaní z RAW“ sa údaje prevedú do štandardnej RGB formy a napr. JPEG formátu

Čo je HDR (High Dynamic range)?

- Viaceré fotky fotené pri rôznych expozíciách sú zlúčené do jednej – plnšie detaily aj v tmavšej aj vo svetlejšej časti
- Napr. Iphone používa v HDR móde 3 fotky, ktoré skladá do jednej. Príklad HDR:



- HDR sa dá realizovať aj manuálne, napr. pomocou GIMPu
  - https://www.youtube.com/watch?v=tHzlmK-S8vo
  - https://www.gimp.org/tutorials/Blending_Exposures/

Málo dramatická obloha?

Kombinovanie so sebou samým napr. Pomocou “blending” v GIMPe

Originál: Večerný pohľad z dlhých dielov na rakúsko	Zlúčenie tmavšej verzie fotky (použitá obloha) a svetlejšej (použitá spodná časť)
	

Čo je to -2.0 EV, +2.0 EV?

Potrebujeme pojem expozícia:

- Čo je expozícia: celkové množstvo svetla, ktoré dopadne na fotografické médium
  - svetlo sa na film/čip dostáva cez objektív (optická sústava, ktorá prispôsobuje tok svetla)
- Čo na ňu vplýva?
  - **Clona (apertúra)** – mechanické zariadenie, ktoré obmedzuje množstvo svetla prechádzajúce cez objektív. Najbežnejšia je „irisová clona“ (viď obr.). Celkové množstvo dopadajúceho svetla nezávisí len od veľkosti otvoru v clone, ale aj od ohniskovej vzdialenosti objektívu. Na clone sa udáva tzv. clonové číslo (nazývané aj „f-číslo) „f-číslo“=f/D. Kde f=ohnisková vzdialenosť, D=priemer otvoru clony. Oboje spravidla v mm. Napr. ak f=100mm a D=25mm, potom 100/25=4, t.j. f-číslo = „f4“ resp. „f/4“ (niekedy sa f číslo udáva aj v tvare „f/N“)

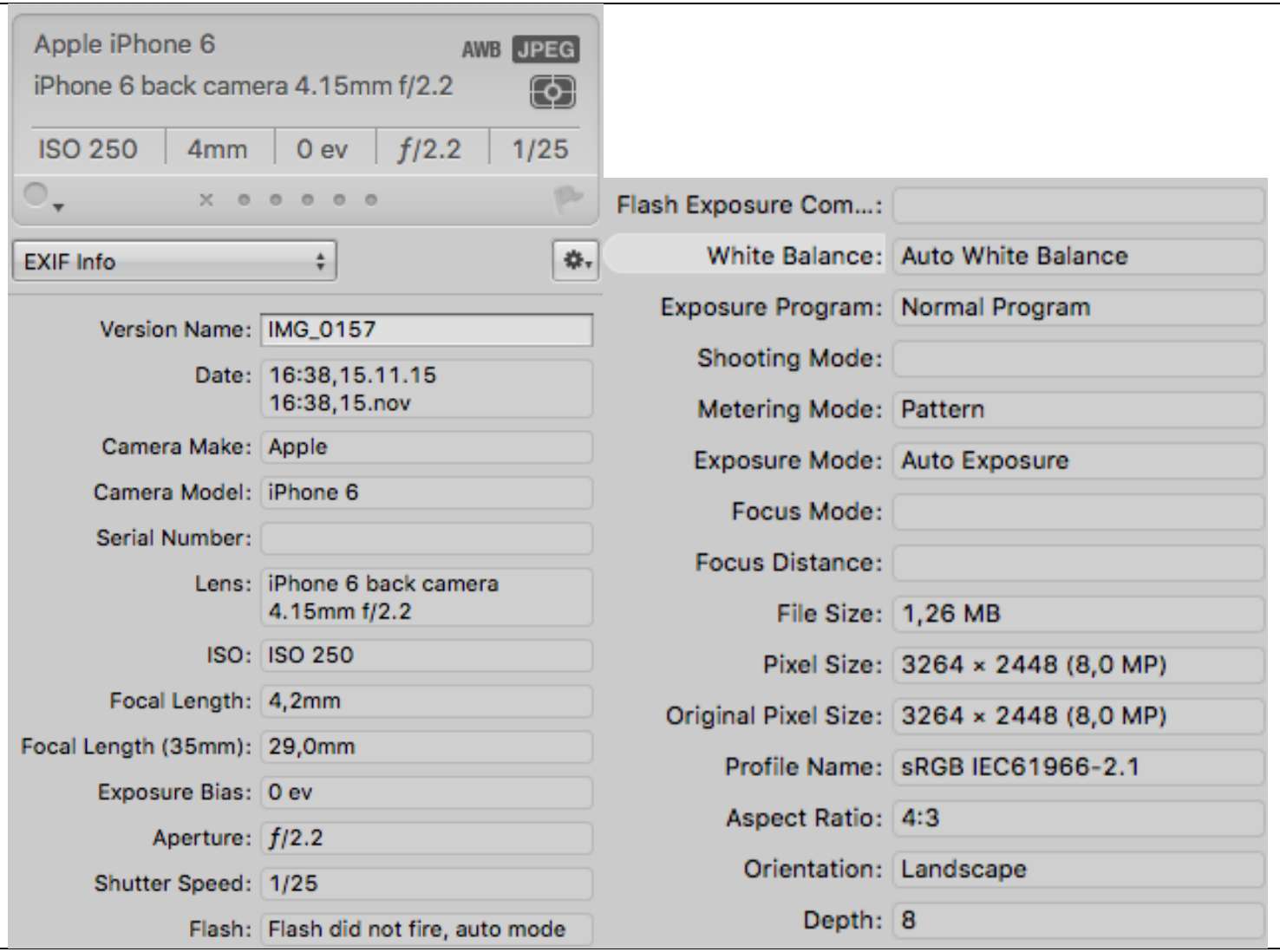
    - clona ľudského oka (zrenica) má clonové číslo f/8.3 (zrenica zúžená na 2mm) až f/2.1 (zrenica otvorená na 8mm) . Z toho akú ma ľudské oko ohniskovú vzdialenosť?
  - **expozičná doba** – čas, po ktorý sa dostáva na médium svetlo, regulujeme pomocou závierky (mechanickej, elektronickej), dôležitý pojem je „rýchlosť závierky“. Napr. 1/2000s.
  - **ISO citlivosť** – citlivosť média (čím väčšia citlivosť tým stačí menej fotónov aby dopadlo, pri elektronických prístrojoch sa realizuje zosilnením)

EV=exposition value = miera množstva svetla na scéne. 0EV=(ISO 100,f/1.0, exp. doba1s) = množstvo svetla ako je v noci v okolí miest

Čo je hĺbka ostrosti? Viete s ňou pracovať? Fotili ste niekedy Makro? Pozerali do stereolupy? Čo je to bokeh? Čo sú polarizačné filter? Čo sú prechodové filter?

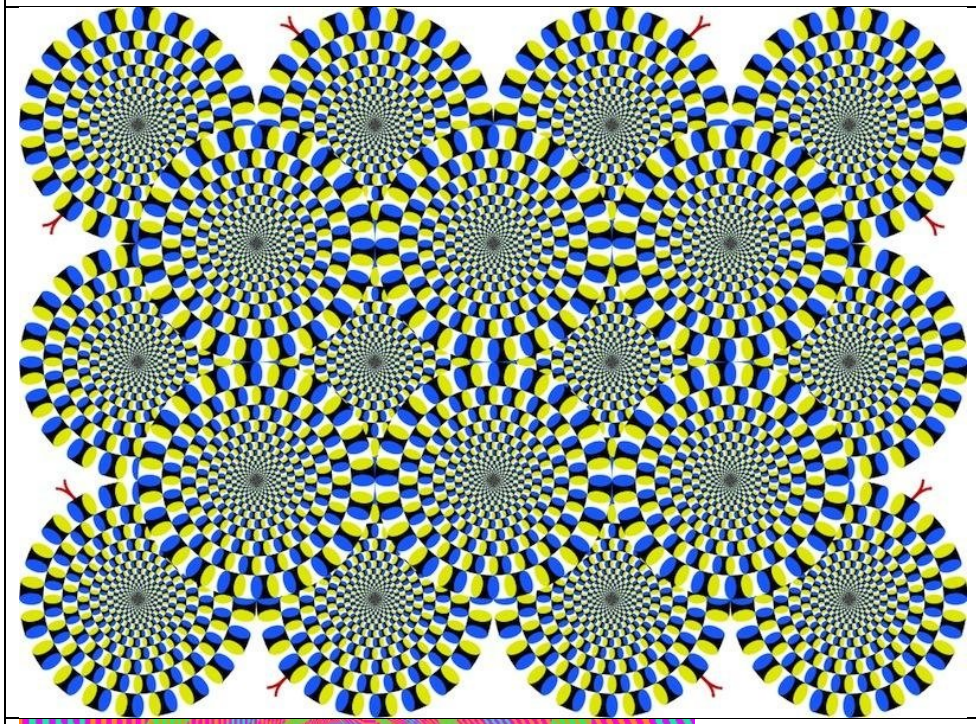
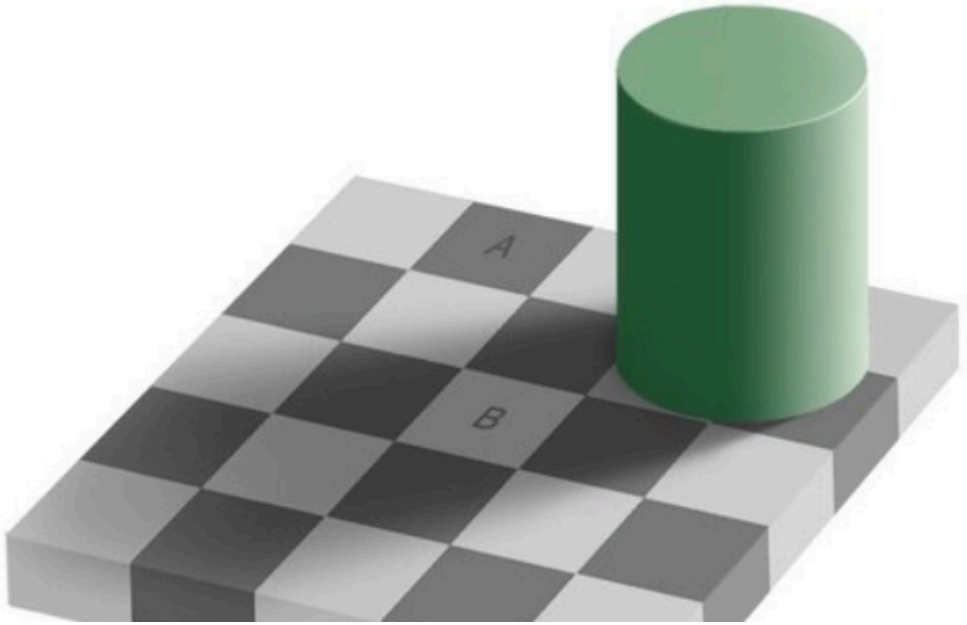
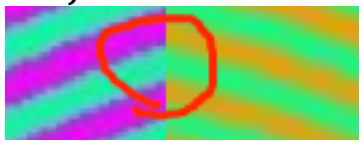
Čo je to EXIF?

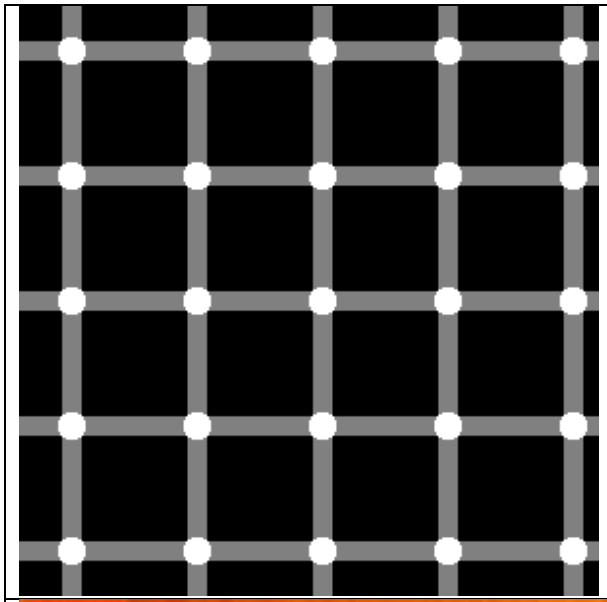
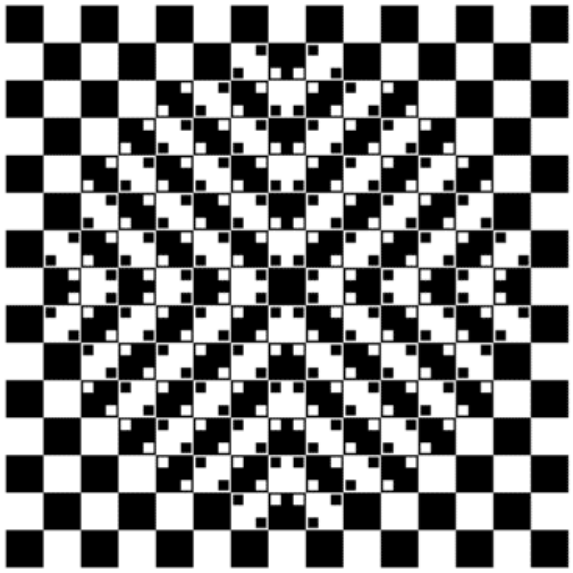
- Špecifikácia pre formát metaúdajov ktoré sa vkladajú do súborov, ktoré vytvárajú digitálne fotoaparáty
- Podporuje ho JPEG, TIFF, ale nepodporuje ho JPEG2000 a PNG

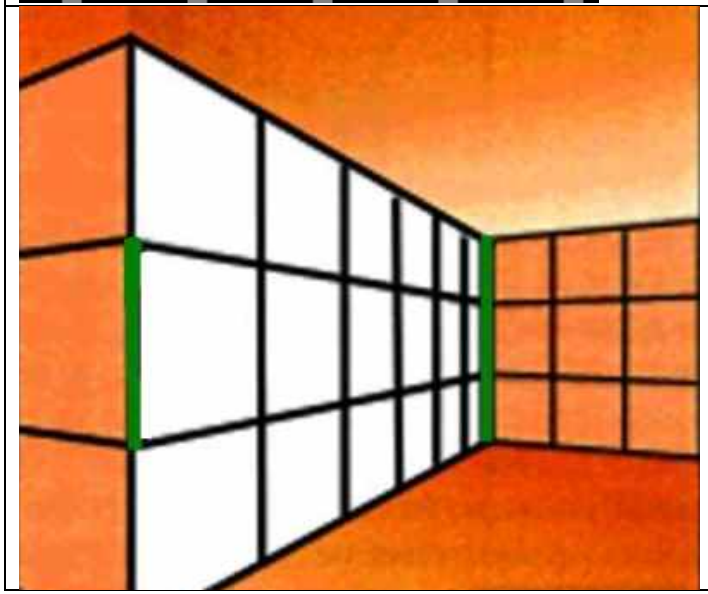
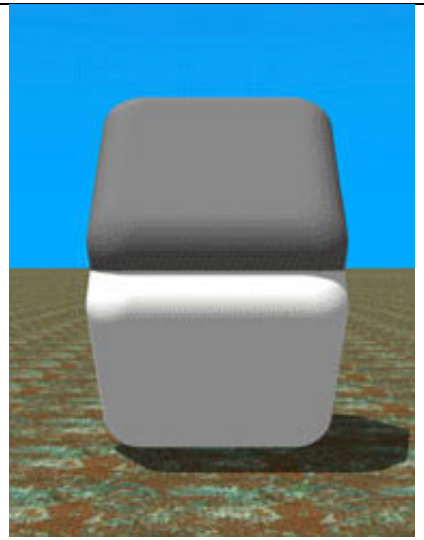
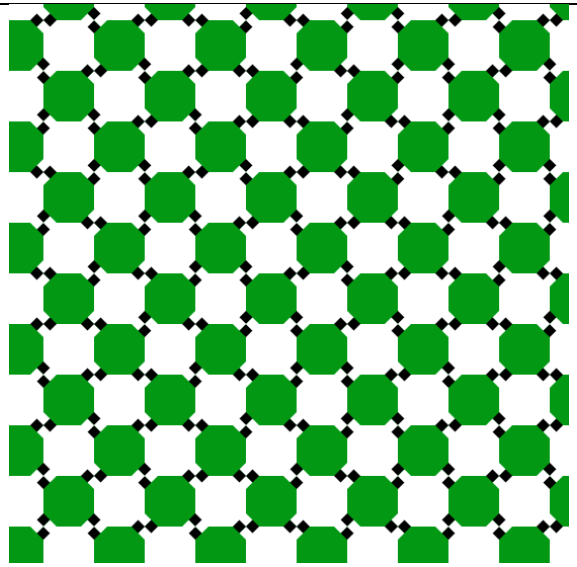


Optické ilúzie

Doverujete vlastnému vizuálnemu systému?

- Obraz stojí, alebo sa hýbe?	Čo je tmavšie A, alebo B? (A, B)
	
	Majú zelená a modrá špirála tú istú farbu? 

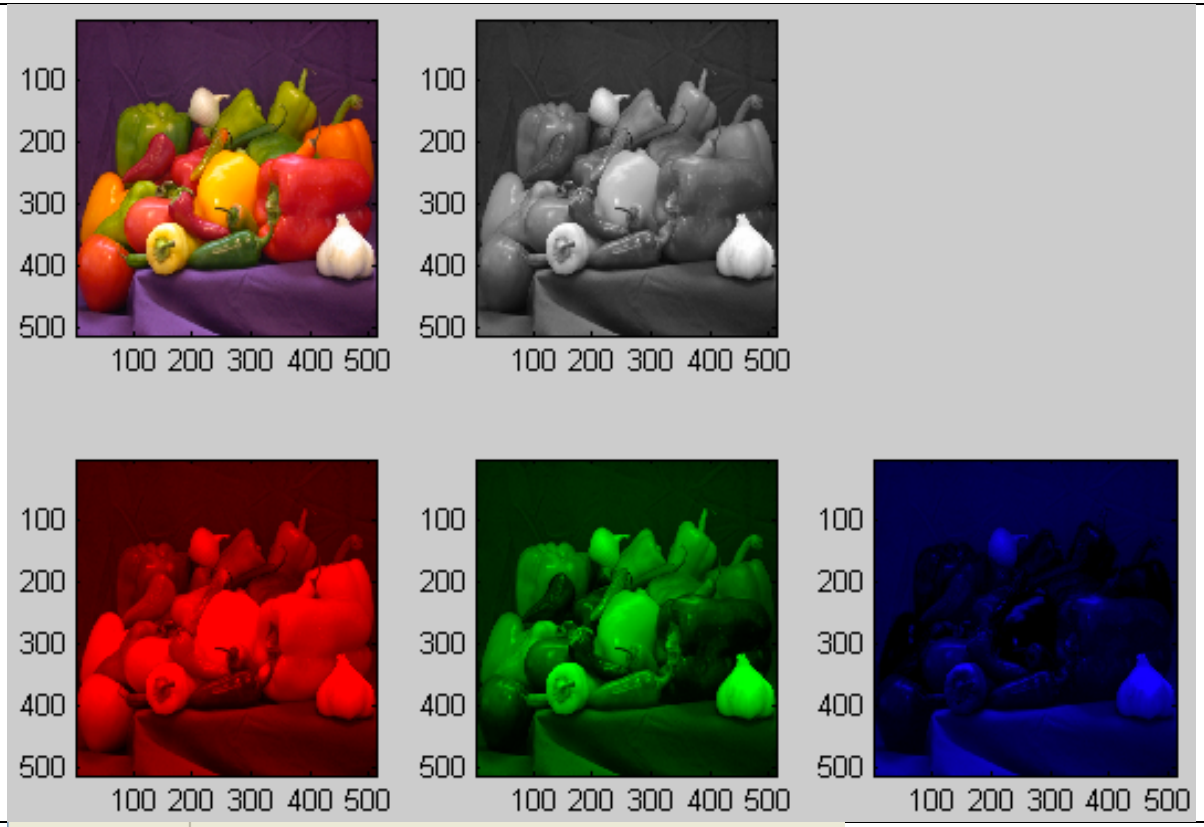



Pozrite si [03_obrazove_iluzie_video]

MATLAB a základná práca s obrazom

```
clear all;
close all;
obr= imread('wpeppers.jpg');
subplot(2,3,1), image(obr);
colormap(gray(256));
obrg=rgb2gray(obr);
subplot(2,3,2), image(obrg);
N=512;
robr=uint8(zeros(N,N,3));
gobr=uint8(zeros(N,N,3));
bobr=uint8(zeros(N,N,3));
robr(:,:,1)=obr(:,:,1);
gobr(:,:,2)=obr(:,:,2);
bobr(:,:,3)=obr(:,:,3);
subplot(2,3,4), image(robr);
subplot(2,3,5), image(gobr);
subplot(2,3,6), image(bobr);
```

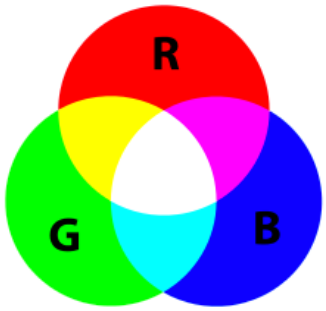



```
imageinfo('wpeppers.jpg')
```

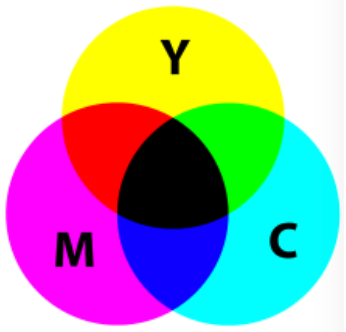
Attribute	Value
Filename	L:\2015_2016_03_04_MM2_02_PREDNASKY_PREDNASKA_06\wpeppers.jpg
FileModDate	31-May-2008 19:31:52
FileSize	153100
Format	jpg
FormatVersion	''
Width	512
Height	512
BitDepth	24
ColorType	truecolor
FormatSignature	''
NumberOfSamples	3
CodingMethod	Huffman
CodingProcess	Sequential
Comment	{}

Skladanie farieb ...

- Spôsob skladania farieb závisí od prezentačného média (a sprievodných fyzikálnych vlastností)
- Ak používame monitor/displej (tvorí svetelné lúče s danou farbou) – aditívne skladanie, lúče sa sčítavajú, farba je čoraz svetlejšia
  - preto displeje majú zvyčajne farby RGB (red, green, blue)= primárne farby
  - žltá = červená+ zelená, magenta=červená + modrá, cyan= ... (sekundárne farby)



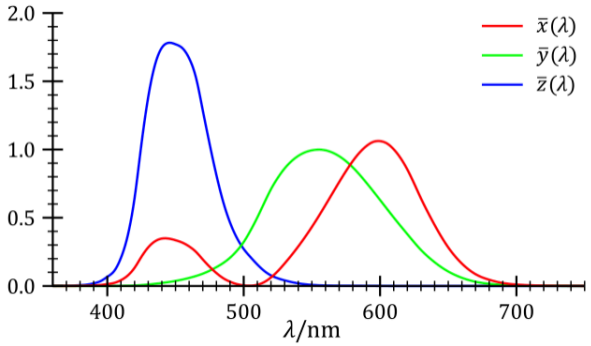
- Ak používame papier, miešaním sa farby odčítavajú (odráža sa čoraz menej a menej) – subtraktívne skladanie
  - preto tlačiarne zvyčajne používajú farby CMYK (cyan, magenta, yellow, black)
  - žltá*magenta=(červená OR zelená) AND (červená OR modrá) = červená



Farebné modely – kolorimetria [http://sccg.sk/~cernekova/Pocitacove_videnie.pdf]

Kolorimetria sa zaoberá numerickým opisom ľudského vnímania farieb
Farebný model opisuje základné farby a schémy miešania týchto farieb do výslednej farby.
Gamut = dosiahnuteľná obasť farieb vo farebom priestore daným zariadením
CIE 1931 (Commission International de L'Eclairage)
Koncept farby= 2 časti: luminancia (jas) a chromaticita (chromaticity)

Model CIE XYZ



Krivky odpovedajúce "štandardnému pozorovateľovi"

$$X = \int_{380}^{780} M(\lambda)\bar{x}(\lambda)d\lambda$$

$$Y = \int_{380}^{780} M(\lambda)\bar{y}(\lambda)d\lambda$$

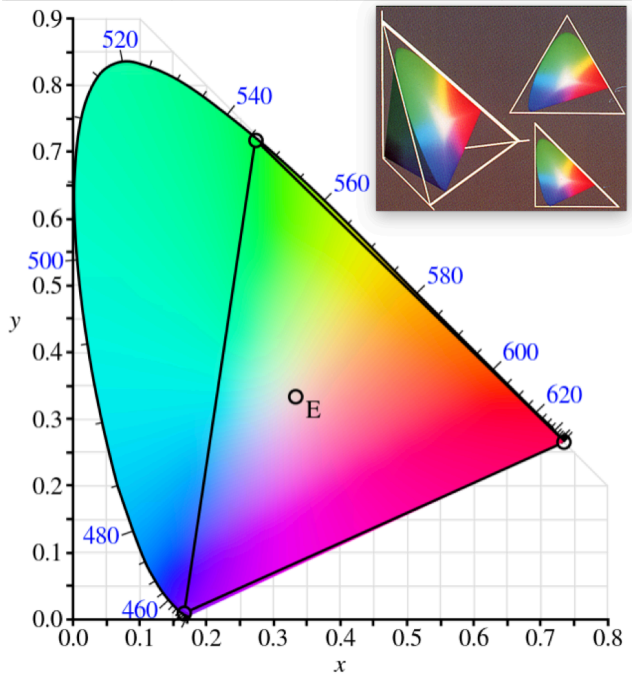
$$Z = \int_{380}^{780} M(\lambda)\bar{z}(\lambda)d\lambda$$

Projekciou do roviny x+y+z=1 (rovnaký jas, farbabude potom definovaná dvomi súradnicami x,y) dostaneme:

$$x = \frac{X}{X+Y+Z}$$

$$y = \frac{Y}{X+Y+Z}$$

$$z = \frac{Z}{X+Y+Z} = 1 - x - y$$



Platí:

- E= neutrálny bod / izoenergetický bod (aj D65)
- Farby vnútri trojuholníka sa dajú získať miešaním farieb vo vrcholoch.
- Trojuholník na obrázku je priestor CIE RGB
- Používa sa aj priestor CIE xyY

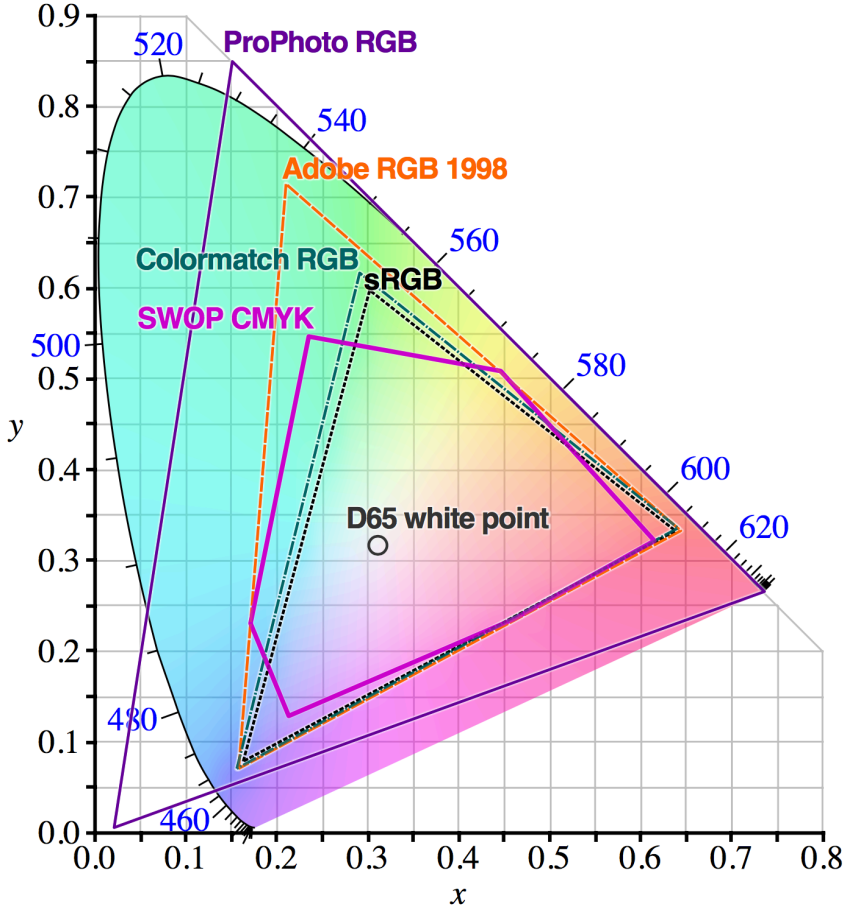
Napr. prepočet CIE xyY -> XYZ:

- $X = xY/y$
- $Z=(1-x-y)Y/y$

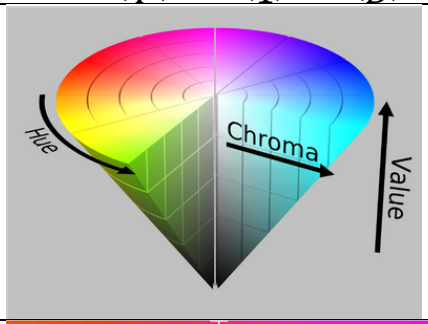
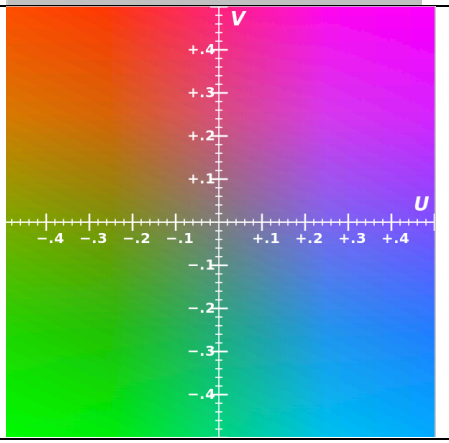
CIE RGB

CIE RGB-> CIE XYZ	CIE XYZ-> CIE RGB
$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{b_{21}} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49 & 0.31 & 0.20 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00 & 0.01 & 0.99 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$	$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 0.41847 & -0.15866 & -0.082835 \\ -0.091169 & 0.25243 & 0.015708 \\ 0.00092090 & -0.0025498 & 0.17860 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$

Rôzne iné farebné modely a ich gamut:



Ďalšie farebné modely

CMYK Cyan Magenta Yellow blacK	kódovanie pre subtraktívny farebný systém, ktorého základnými farbami sú doplnkové farby k červenej, zelenej a morej - azúrová, purpurová a žltá.	CMYK ..(0,1) RGB (0..255) $R = 255(1-C)(1-K)$ $G = 255(1-M)(1-K)$ $B = 255(1-Y)(1-K)$ červená: (0,1,1,0)->(255,0,0)
CMY	Ako CMYK, ale bez čiernej	$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}$
HSV (Hue Saturation Value)	hue udáva odtieň, saturation sýtosť a value jas. Čiernu dostaneme, ak je jas nastavený na 0, bielu ak je jas na maxime (nezávisí od sýtosti).	
YUV (YCbCr) (YPbPr)	kódovanie, ktoré slúži na zachovanie čiernobielej zložky televízneho vysielania. Y je jas (svietivosť, luminance) a predstavuje čiernobielu zložku, U a V sú farebné zložky (farebnosť, chrominance).	

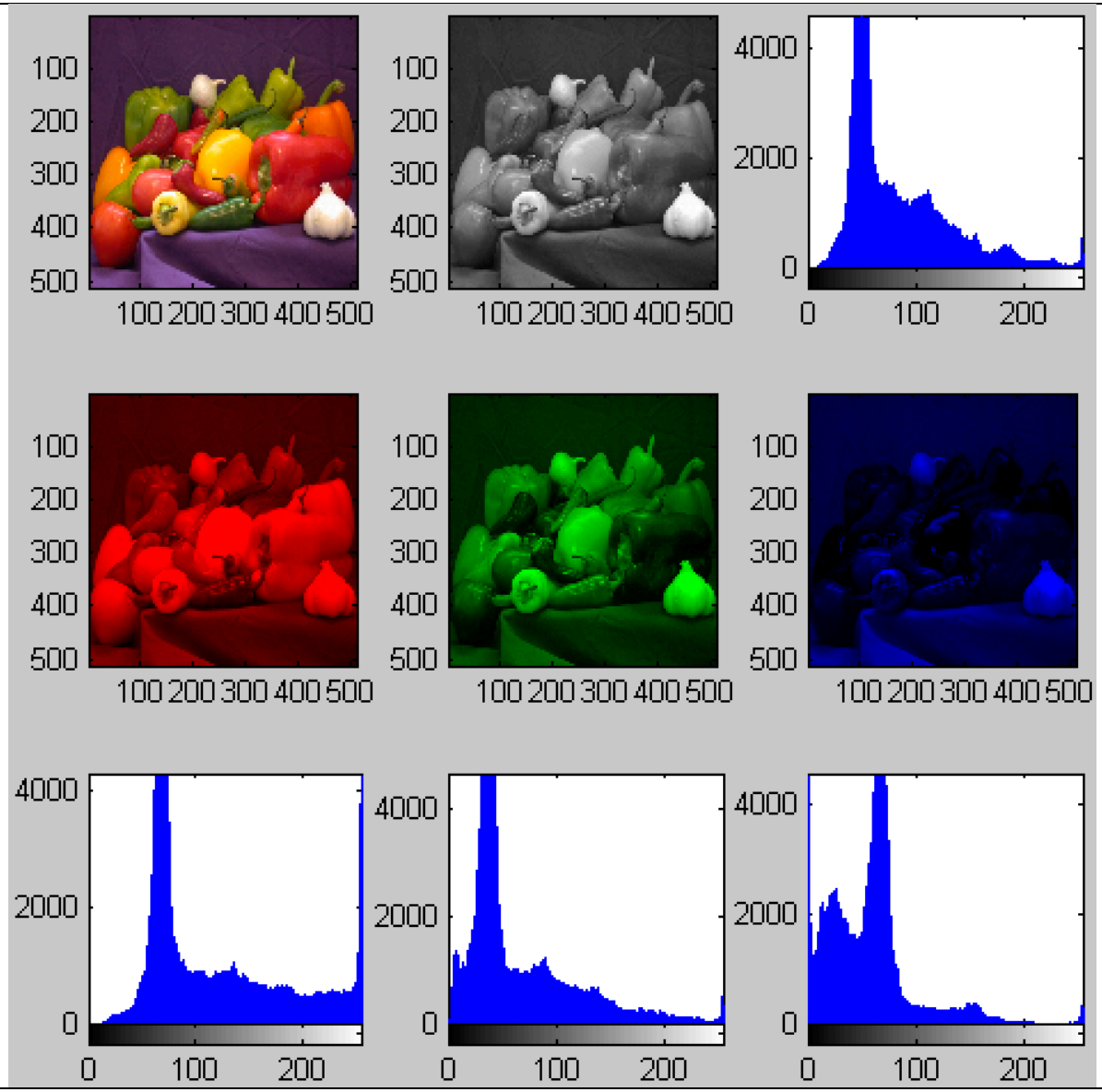
Redukcia farieb

```
clear all;
close all;
I = imread('wpeppers.jpg');
imshow(I);

[X_no_dither,map]= rgb2ind(I,8,'nodither');
figure, imshow(X_no_dither,map);

[X_dither,map]=rgb2ind(I,8,'dither');
figure, imshow(X_dither,map);
```

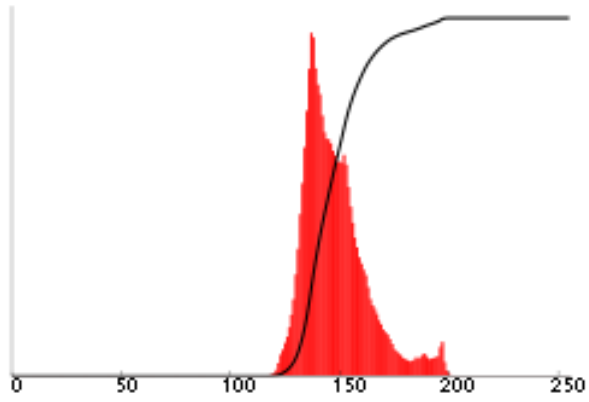
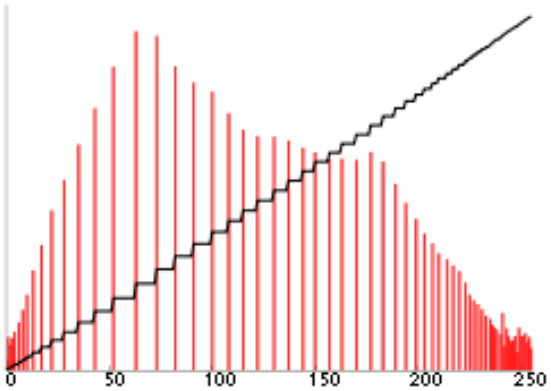
Farebné Histogramy

Histogram je funkcia, ktorá pre každú úroveň jasu udáva počet pixelov v obraze, ktoré majú túto úroveň.

```
obr= imread('wpeppers.jpg');
imageinfo('wpeppers.jpg')
subplot(3,3,1), image(obr);
colormap(gray(256));
obrg=rgb2gray(obr);
subplot(3,3,2), image(obrg);
subplot(3,3,3), imhist(obrg);
N=512;
robr=uint8(zeros(N,N,3));
gobr=uint8(zeros(N,N,3));
bobr=uint8(zeros(N,N,3));
robr(:,:,1)=obr(:,:,1);
gobr(:,:,2)=obr(:,:,2);
bobr(:,:,3)=obr(:,:,3);
subplot(3,3,4),image(robr);
subplot(3,3,5),image(gobr);
subplot(3,3,6),image(bobr);
subplot(3,3,7),imhist(obr(:,:,1));
subplot(3,3,8),imhist(obr(:,:,2));
subplot(3,3,9),imhist(obr(:,:,3));
```



Histogram a jeho ekvalizácia

		Pôvodný obrázok a jeho histogram Interval jasu: $< p_0, p_k >$ Histogram: H(p) Histogram odpovedá funkcii hustoty pravdepodobnosti (červená), zobrazená je aj distribučná funkcia (čierna).
		Ekvalizovaný obrázok a jeho histogram Interval jasu: $< q_0, q_k >$ Histogram: G(q)

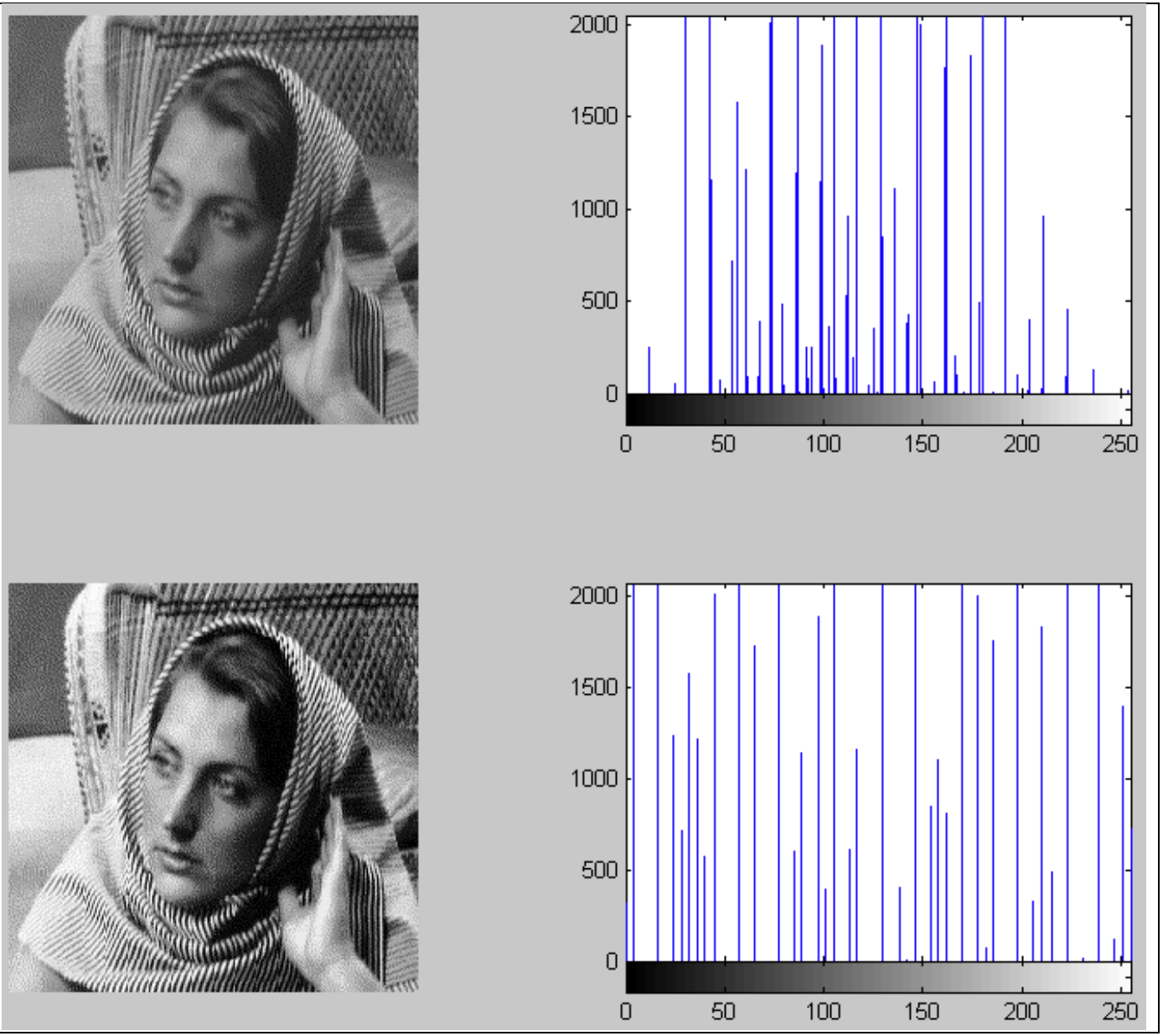
Cieľom je nájsť jasovú transformáciu T hodnôt p na q: $q = T(p)$ aby výsledý histogram $G(q)$ bol rovnomerný pre celý výstupný rozsah jasov $< q_0, q_k >$. Ak počet bodov obrazu je N^2 potom platí:

$$q = T(p) = \frac{q_k - q_0}{N^2} \sum_{i=p_0}^{p} H(i) + q_0$$

(vidíme, že prepočítavame „cez distribučnú funkciu“)

Ekvalizácia v matlabe

```
clear all;
close all;
X=load('woman.mat');
A=uint8(X.X);
subplot(2,2,1),imshow(A);
subplot(2,2,2),imhist(A);
B=histeq(A)
subplot(2,2,3),imshow(B);
subplot(2,2,4),imhist(B);
```



Normalizácia histogramu (zmena rozsahu jasu)

$$q = T(p) = (p - p_0)\frac{q_k - q_0}{p_k - p_0} + q_0$$

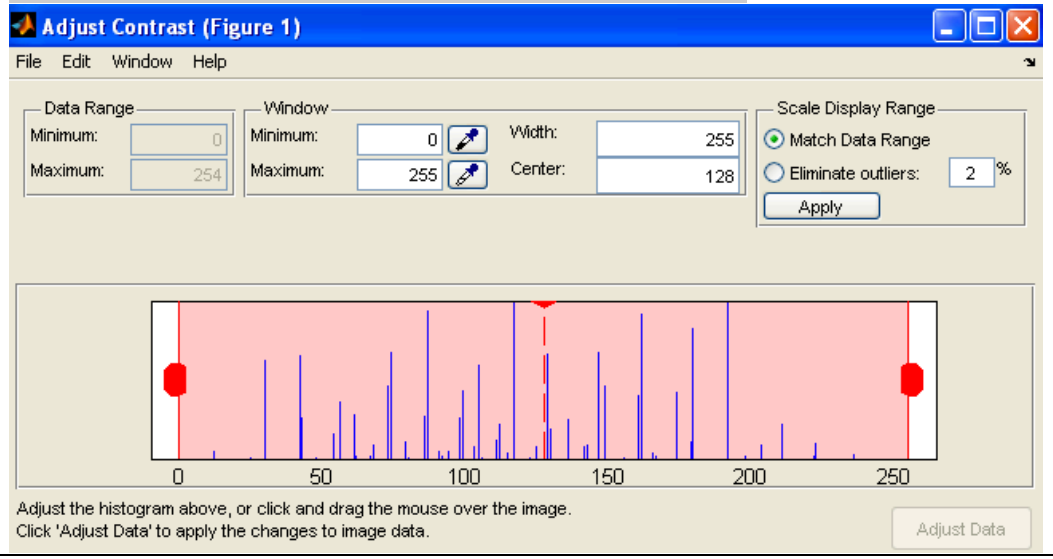
(iba uniformne rozťahujeme/zťahujeme, neprepočitavame cez distribučný funckiu)

Bodové jasové transformácie

- Sčítanie a odčítanie obrazov: $c(x, y) = a(x, y) \pm b(x, y)$,
- Násobenie a delenie obrazov
- Násobenie a delenie obrazu konštantou
- Logaritmický operátor: $g(x, y) = \frac{255}{\log(1+R)} \log(1 + f(x, y))$, R je max. jas vstupného obrazu. Používa sa na úpravu podexponovaného obrazu
- Exponenciálny operátor: používa sa na úpravu preexponovaného obrazu
- Konvolúcia: $g(x, y) = \sum_m \sum_n h(x - m, y - n) f(m, n)$,kde $(m, n) \in O$
  - h(x,y)- konvolučná maska, konvolučné jadro
  - podľa toho aká maska sa zvolí, dosiahneme napr.
    - vyhladzovanie – potláčanie šumu, napr. priemerovací filter $h = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$
    - ostrenie – detekcia hrán a čiar, napr. Laplaceov operátor $h = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$
- Nelineárne metódy

Matlab - šedotónový obraz - kontrast a jeho zmena

```
X=load('woman.mat');
A=uint8(X.X);
colormap(gray(256));
h=imshow(A);
colorbar;
hp = impixelinfo;
imcontrast(h);
```

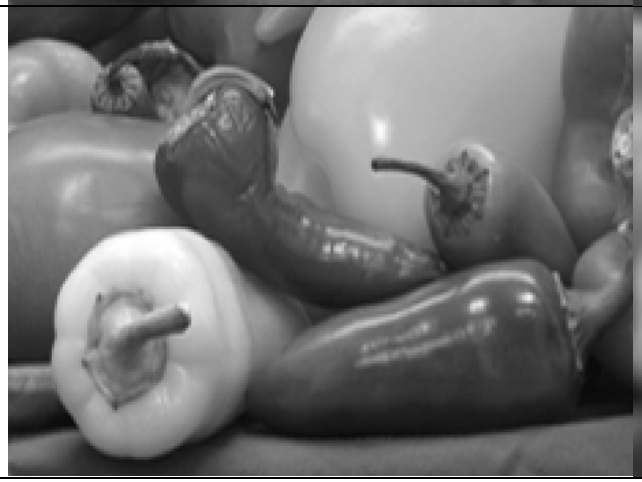


Matlab konvolúcia

```
I = imread('wpeppers.jpg');
h = fspecial('unsharp');
I2 = imfilter(I,h);
imshow(I)
figure, imshow(I2)
```

```
h =
  -0.1667  -0.6667  -0.1667
  -0.6667   4.3333  -0.6667
  -0.1667  -0.6667  -0.1667
```

```
I = imread('wpeppers.jpg');
h = ones(10,10)/100;
I2 = imfilter(I,h);
imshow(I)
figure, imshow(I2)
```



(vľavo origialy, vpravo prefiltrované)

Ostrenie a “unsharp”

- Prečo sme mali pri ostrení : “h = fspecial('unsharp');” ?
- Aj kedysi sa pri „ostrení“ fotiek z negatívov ostrilo pomocou rozmazaného (unsharp) obrazu
- Postup:
  - od obrazu odčítame jeho rozmazanú verziu -> dostaneme rozdiely, ktoré vlastne chceme v pôvodnom obraze zvýrazniť
  - rozdiely pričítame k pôvodnému obrazu (a tým rozdiely zvýrazníme)
  - Ako to spraviť pomocou konvolúcie?
  - Originál = $O = O * \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
  - rozmazaný obraz $B = O * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} 1/9$
  - Rozdielový obraz = $O - B$
  - Zostrený obraz=

$$O + (O - B) = 2O - B = 2O * \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} - O * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} 1/9 =$$

$$O * \left(2 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} - \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} 1/9 \right) = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 17 & -1 \\ -1 & -1 & -1 \end{pmatrix} 1/9$$