Human and animal models in BioRobotics

Marcello Calisti
marcello.calisti@santannapisa.it

The BioRobotics Institute
Scuola Superiore Sant'Anna

Pontedera, 01 October 2019
Scuola Superiore Sant'Anna

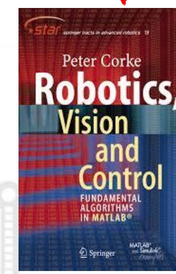
Reference materials and credits

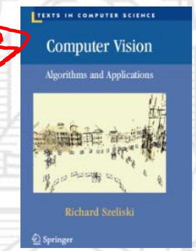
Most of the material presented in these lessons can be find on the brilliant, seminal books on robotics and image analysis reported hereafter:

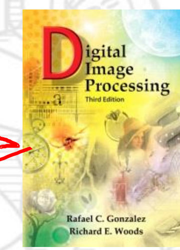
1. P.I. Corke, "Robotics, Vision & Control", Springer 2011, ISBN 978-3-642-20143-1

2. R. Szeliski, "Computer Vision: Algorithms and Applications", Springer-Verlag New York, 2010

3. R.C. Gonzalez & R.E. Woods, "Digital Image Processing (3rd edition)", Prentice-Hall, 2006

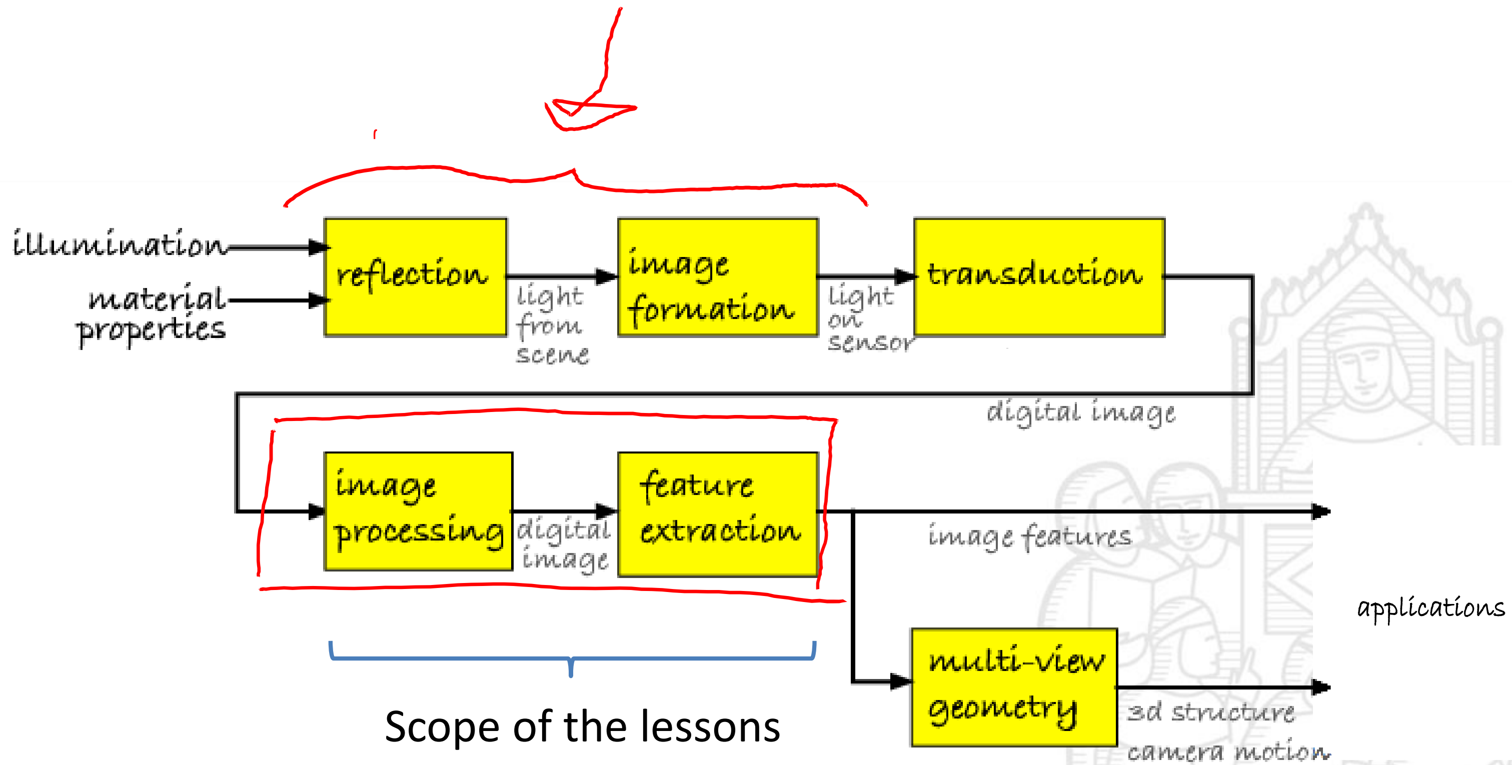






Most of the images of these lessons are downloaded from RVC website http://www.petercorke.com/RVC/index.php and, despite they are free to use, they belong to the author of the book.

Overall computer vision process and scope of the lessons

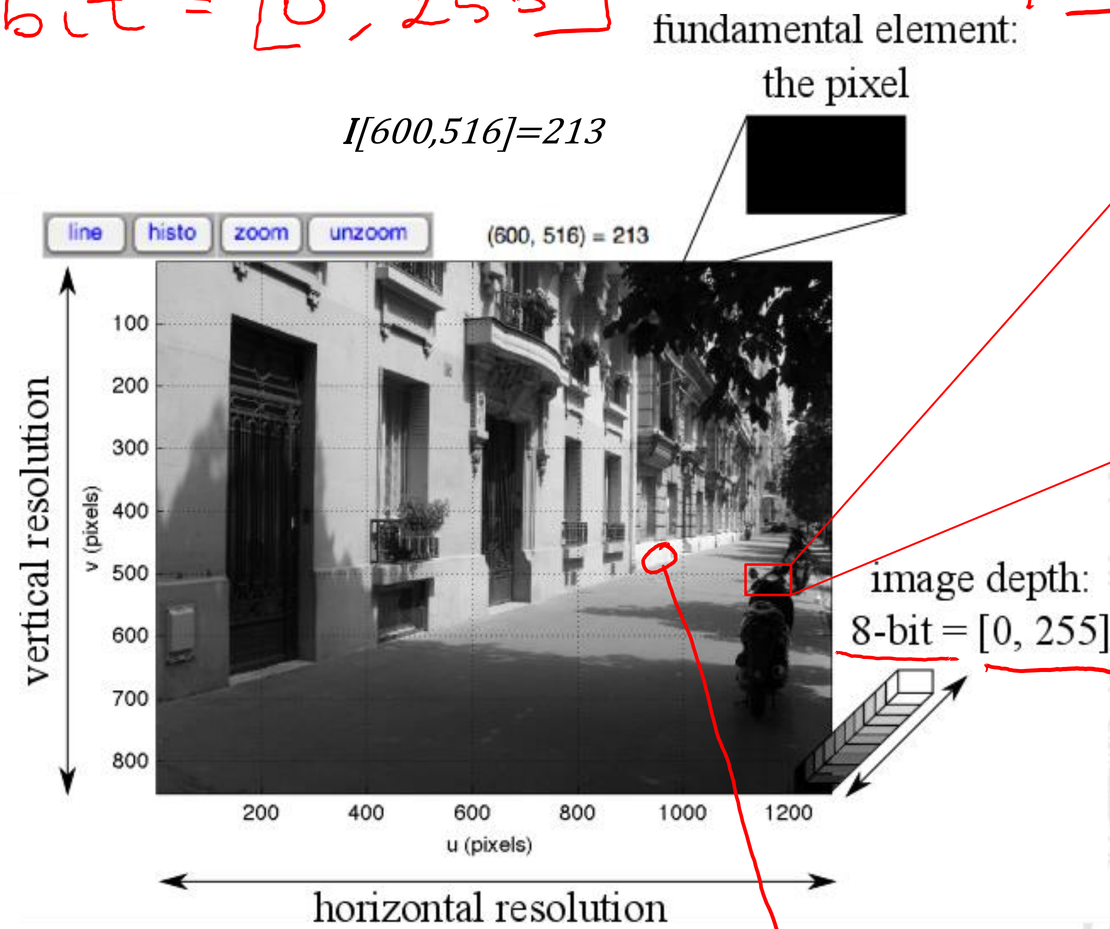


What's a digital image?

320 x 240, LR

1080 x 920

8-bit = [0, 255]



255 B.

Digital images are **mosaics** made of **pixels**

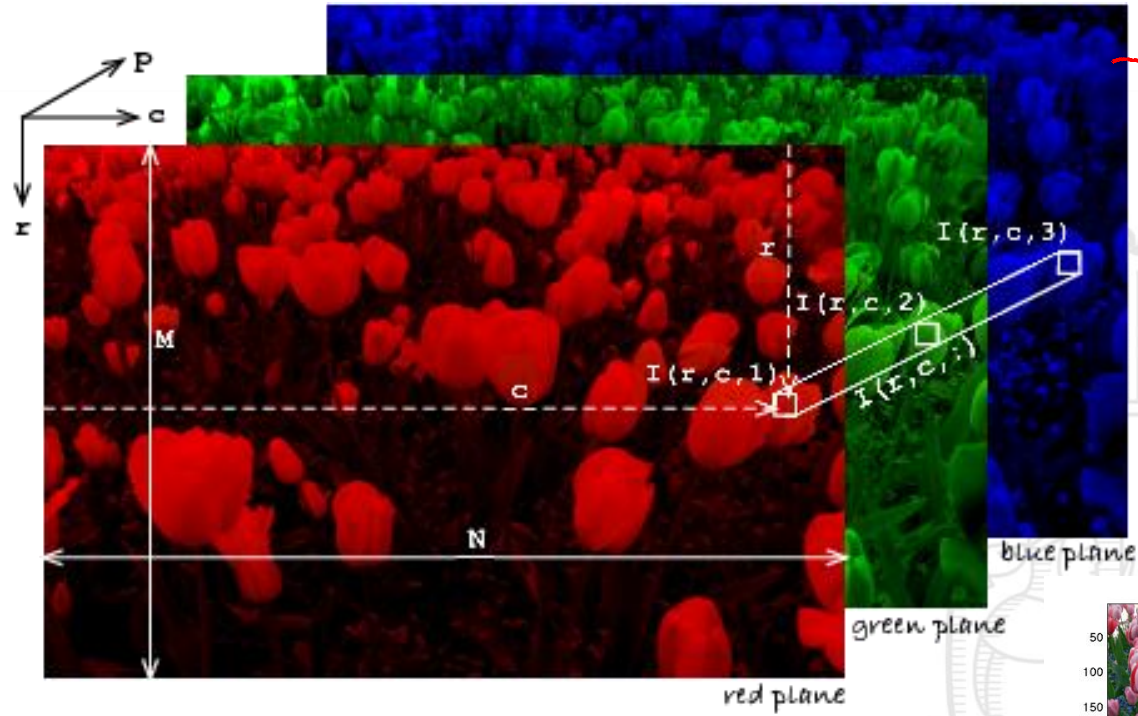
Image resolution is the number of pieces (pixel) used to build the mosaic (image)

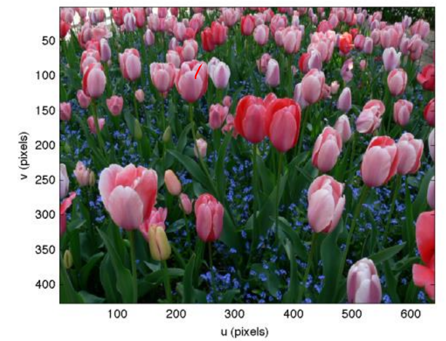
Image depth is the number of colours (levels) of mosaic pieces

Sant'Anna
Scuola Universitaria Superiore Pisa

Colour images

Colour images have three channels: the most common triplet is the R-G-B





There are other very useful common space:
HSV, XYZ, CIE, YUY, …

Sant'Anna
Scuola Universitaria Superiore Pisa

Colour images

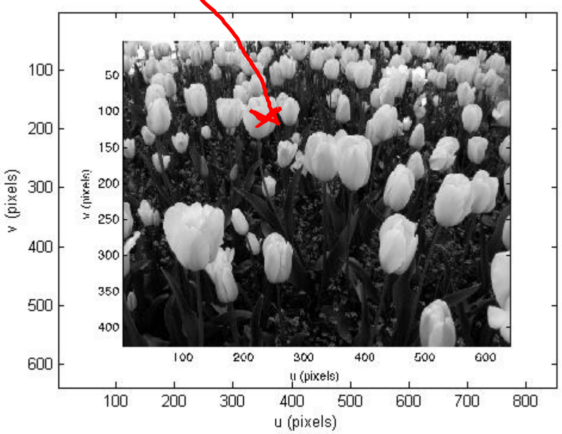
640x854x3 uint8

0-255

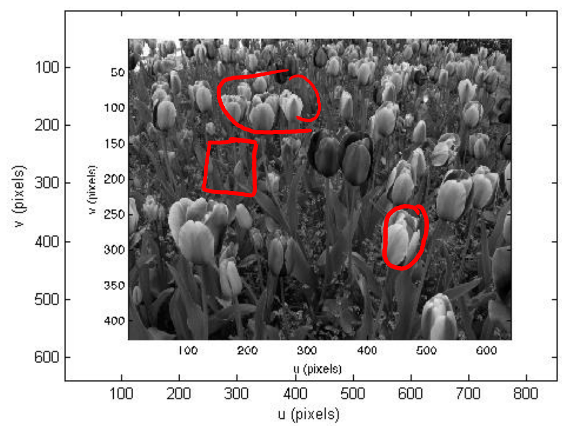
R G B

255, 255, 255

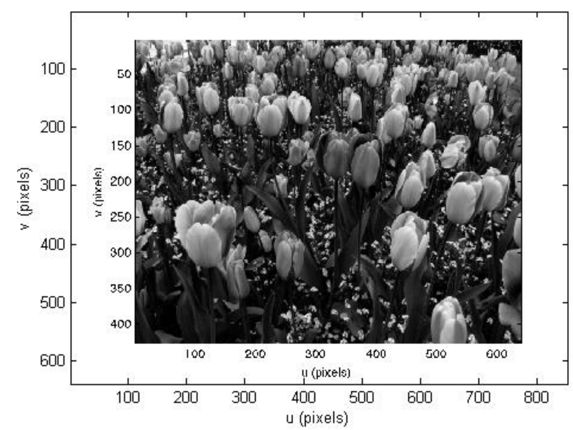
Red channel



Green channel



Blue channel

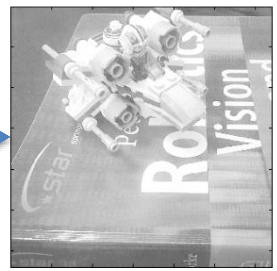


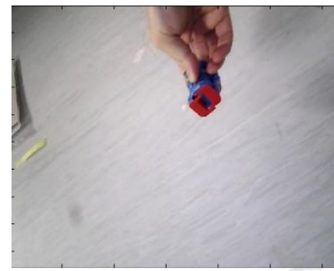
640x854x1 uint8

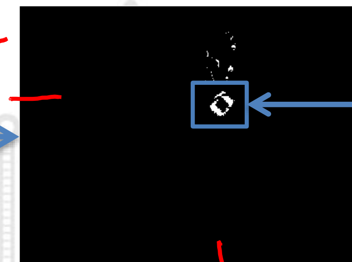
Image processing

Transform one or more input images into an output image.

To enhance the image

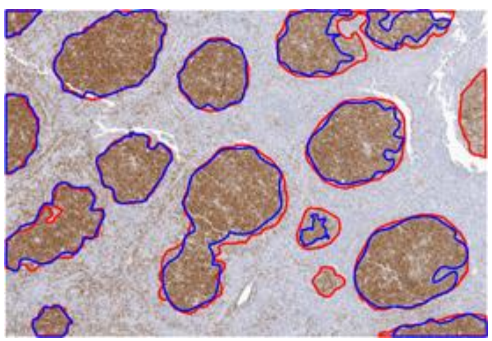






Human interpretation

Features extraction



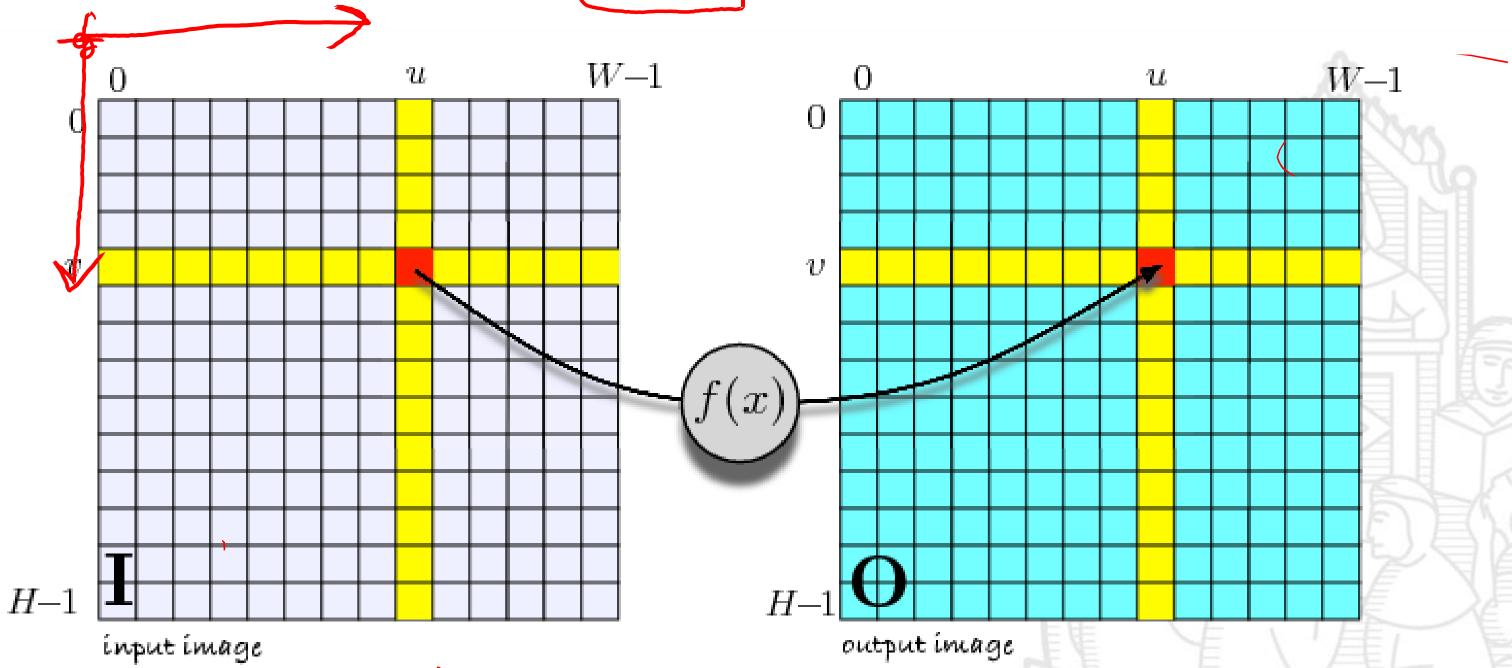
Computer aided diagnosis
Metin Gurcan, Ph.D – Ohio State

Objects recogniton
HERB robot butler – Carnegie Mellon

Monadic operations (pixel operators)

O[u,v] = f(I[u,v])

$$\boldsymbol{O}[u,v] = f(\boldsymbol{I}[u,v]), \qquad \forall(u,v) \in \boldsymbol{I}$$

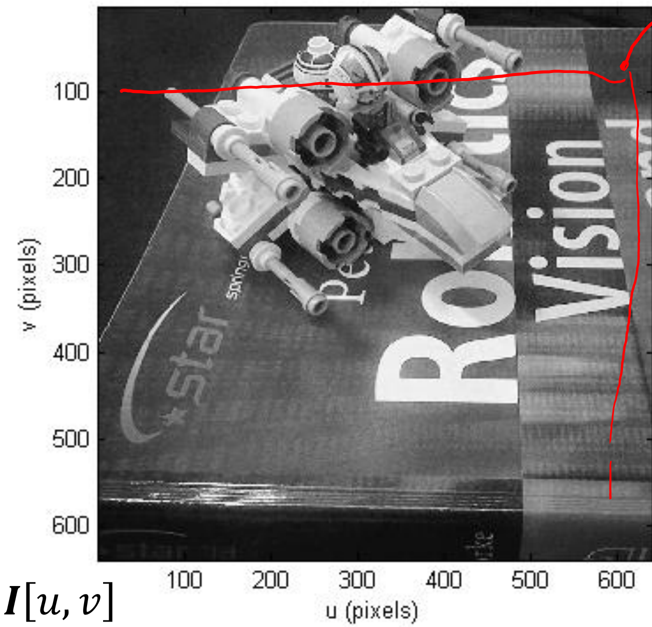


Img

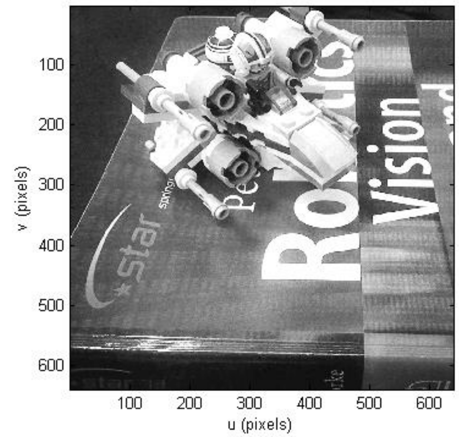
Lightening and darkening

8-bit → 2^8 → 255

I[600,100] = 120

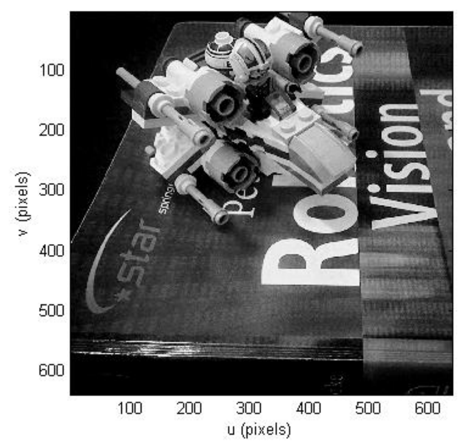


$\boldsymbol{I}[u, v]$



$\boldsymbol{O}[u, v] = \boldsymbol{I}[u, v] + 50$

205 → 255

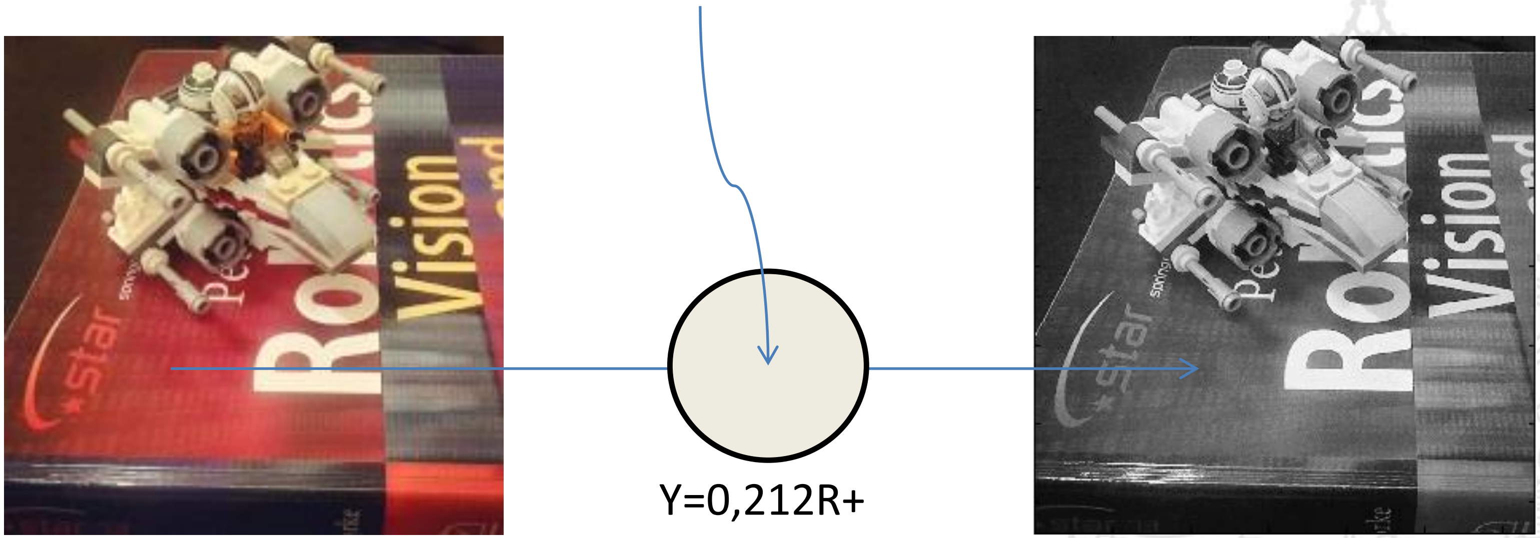


$\boldsymbol{O}[u, v] = \boldsymbol{I}[u, v] - 50$

Monadic operations change the distribution of grey levels on images

Simple monadic operation (more channel):

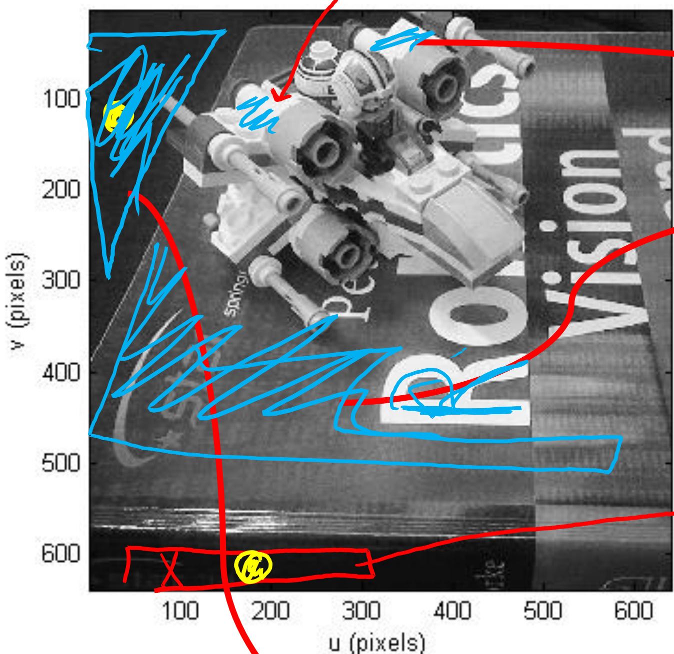
Gray-scale conversion with International Telecommunication Unit (ITU) recommendation 709

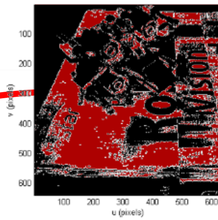


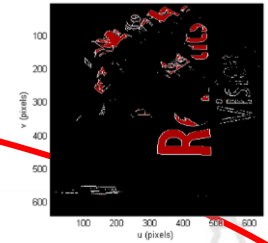
Y=0,212R+
0,7152G+0
,0722B

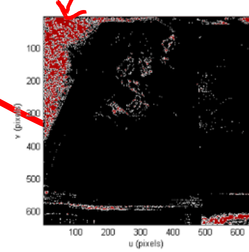
Histogram

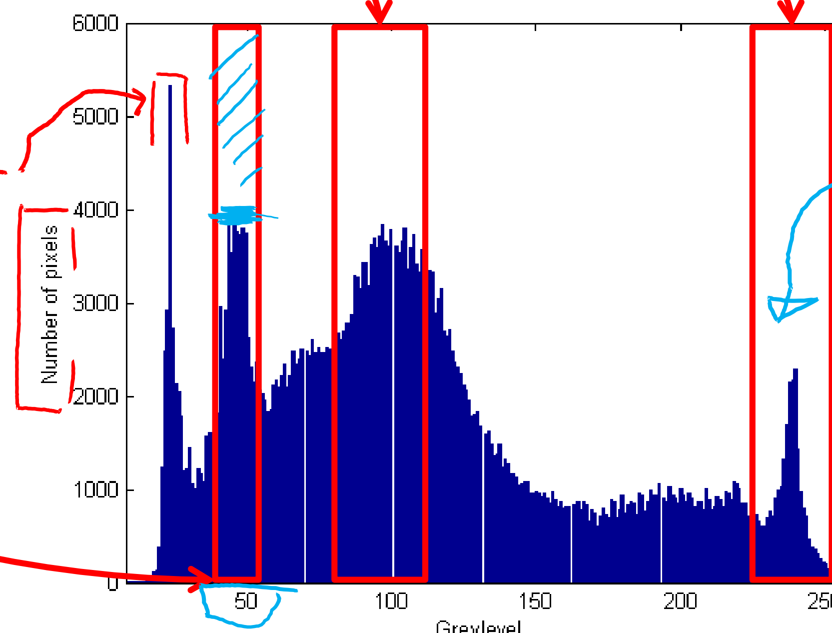
Is a graph representing the grey level occurrences of an image.



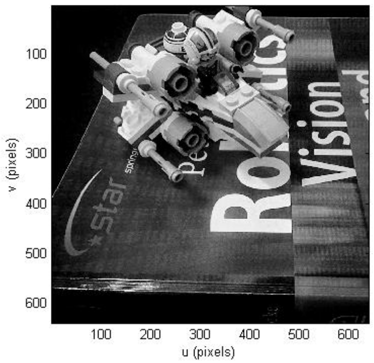




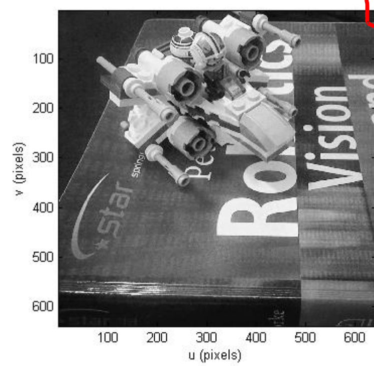




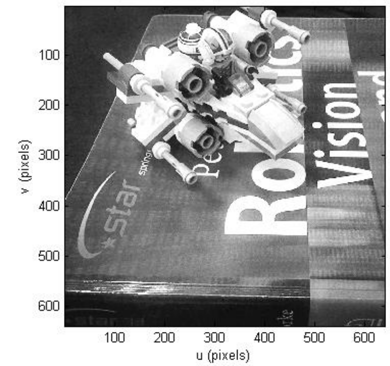
Histograms and monadic operations



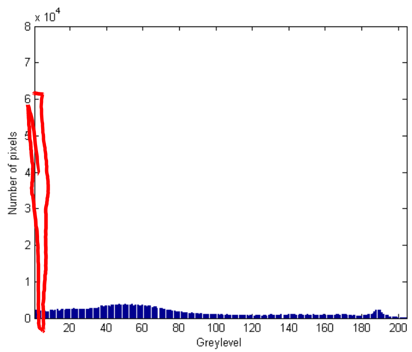
$O[u,v] = I[u,v] - 50$

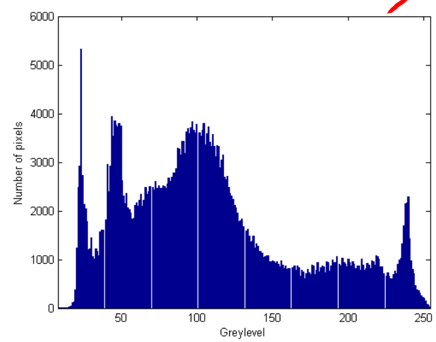


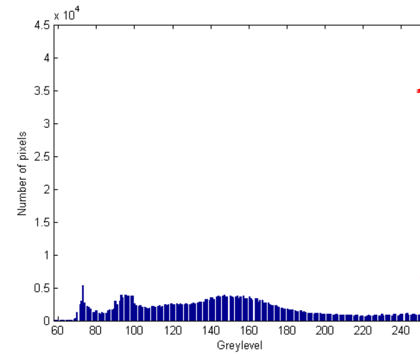
$I[u,v]$



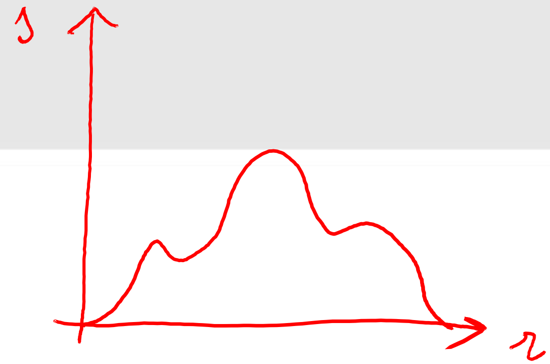
$O[u,v] = I[u,v] + 50$







Common operations



$$s(r) = c \cdot r^{\gamma}$$

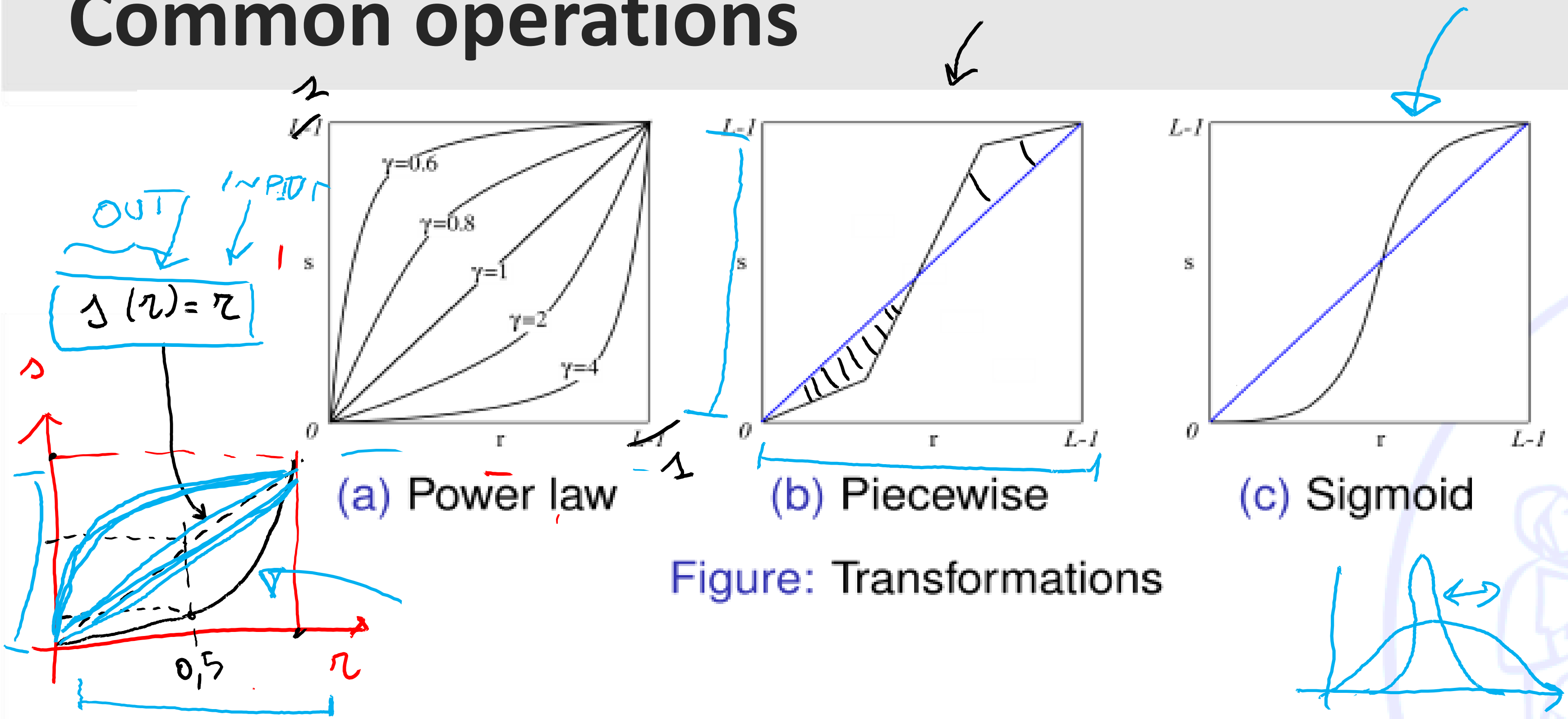
$$s(r) = \begin{cases} c_1 \cdot r & 0 \leq r < r_{min} \\ c_2 \cdot r & r_{min} \leq r < r_{max} \\ c_3 \cdot r & r_{max} \leq r < L-1 \end{cases}$$

$$s(r) = \frac{c_1}{1 + e^{-r}}$$

- power law
- piecewise
- sigmoid

Each function requires parameters definition

Common operations

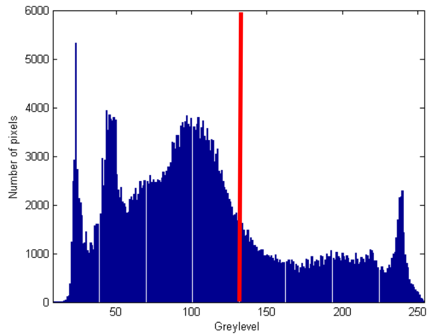


(a) Power law (b) Piecewise (c) Sigmoid

Figure: Transformations

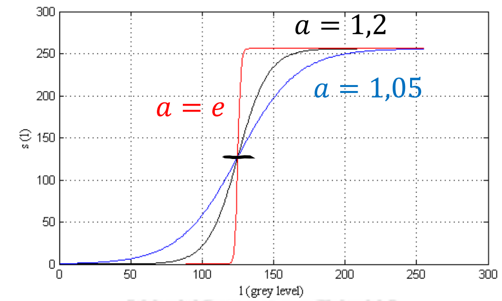
Power law	lighten or darken
Piecewise	flexible
Sigmoid	enhance the contrast

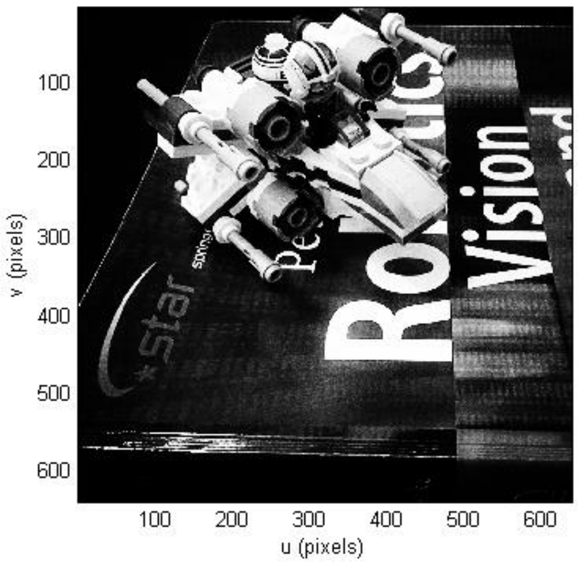
Contrast enhancement

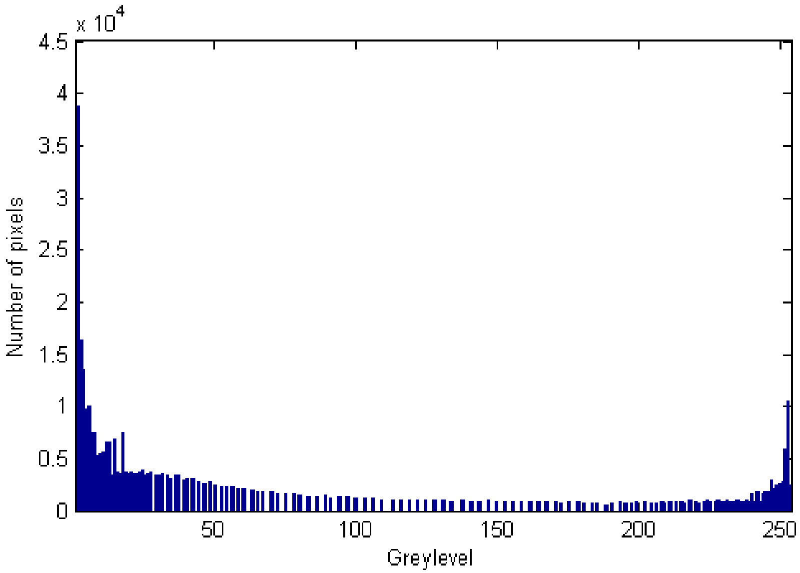


Sigmoid function

$$s(l) = \frac{256}{1 + a^{-(l-125)}}$$







Pay attention

$$\begin{aligned} L &= 21 \\ c &= 1 \\ \gamma &= 2 \\ s(r) &= c \cdot r^{\gamma} \end{aligned}$$

$$\begin{aligned} s(1) &= 1 \\ s(2) &= 2^2 = 4 \\ s(3) &= 3^2 = 9 \\ s(4) &= 4^2 = 16 \\ s(5) &= 5^2 = 25 \\ \ldots &= \ldots \\ s(20) &= 20^2 = 400 \end{aligned}$$

???

We have only 21 levels, but:

$$s(5) = 5^2 = 25$$

Monadic operations

Code sample >

```
% lightening/darkening
xwing_light=xwing_grey+50;
idisp(xwing_light);
xwing_dark=xwing_grey-50;
idisp(xwing_dark);
% select areas by levels
level48 = (xwing_grey>=40) & (xwing_grey<=50) ;
idisp(level48);
level225 = (xwing_grey>=225) &
(xwing_grey<=255) ;
idisp(level225);
% contrast enanch
xwing_contrast=zeros(r,c);
    for i=1:r
       for j=1:c
           xwing_contrast(i,j)=256./(1+1.05.^-(
double(xwing_grey(i,j))-150));    % Sigmoid
       end
    end
 idisp(xwing_contrast)
```

Pay attention

We need to remap the output between $[0, L-1]$:

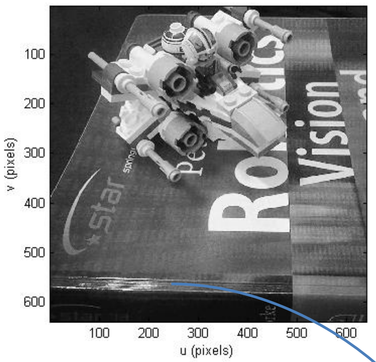
$$\frac{s'}{s} = \frac{20}{400}$$

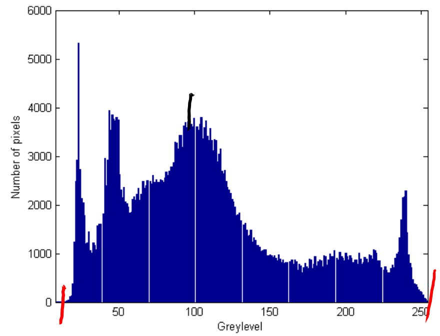
$$\frac{20}{400} = \frac{L-1}{(L-1)^\gamma} = (L-1)^{1-\gamma}$$

$$s' = (L-1)^{1-\gamma} s = c \cdot s = c \cdot r^\gamma$$

Thus c is related to L and γ.

Histogram equalization





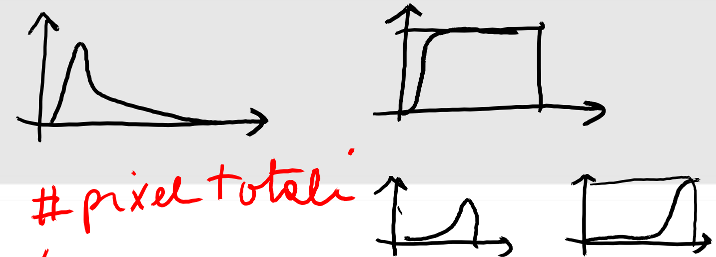
$$c(l) = \frac{1}{N}\sum_{i=0}^{l} h(l) = c(l-1) + \frac{h(l)}{N}$$

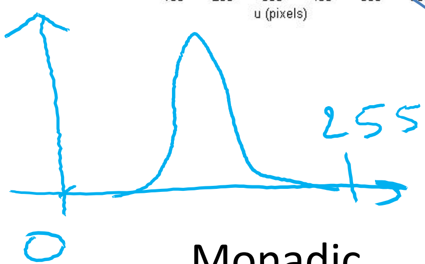
c(l)	Cumulative distribution
h(l)	histogram
l	Grey level

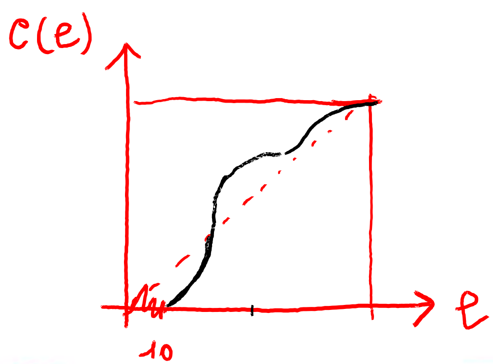
f(l)=c(l)

Monadic operation

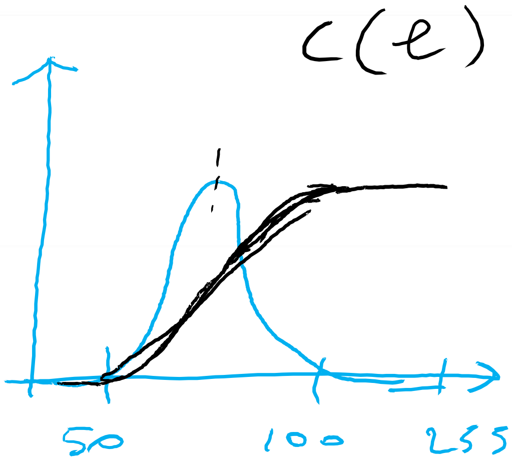
$$\boldsymbol{O}[u,v] = c(\boldsymbol{I}[u,v]), \forall (u,v) \in \boldsymbol{I}$$







N: total pixel



$$c(e) = \frac{1}{N} \sum_{i=0}^{e} h(i)$$

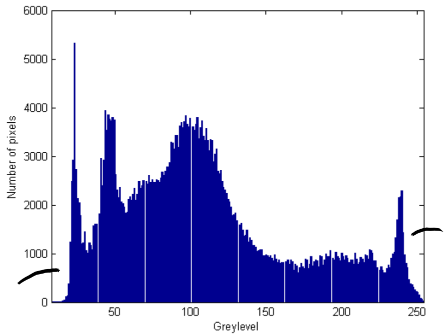
$$c(0) = \frac{1}{N} \quad h(0) = 0$$

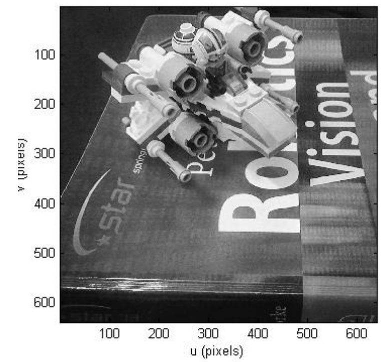
$$c(50) = \frac{1}{N} h(50) = \frac{1}{N}$$

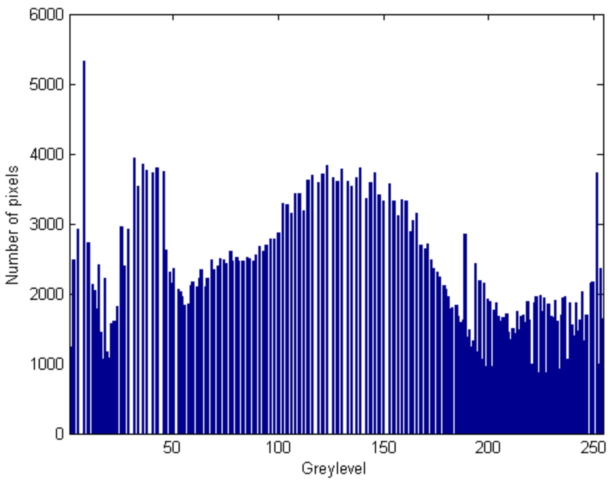
$$c(51) = \frac{1}{N} (h(50) + h(51))$$

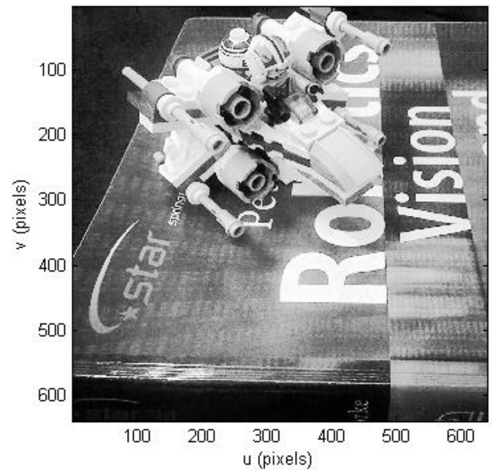
$$= c(e-1) + \frac{h(e)}{N}$$

Histogram equalization



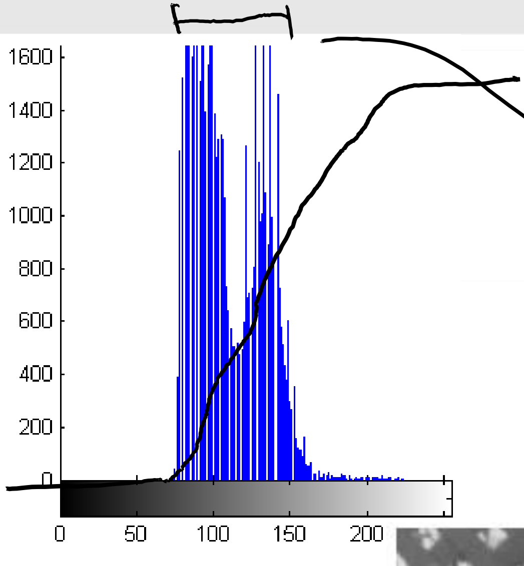






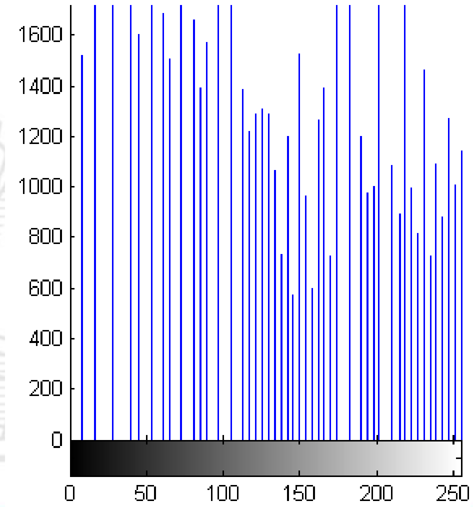
After equalization

Histogram equalization



before

after



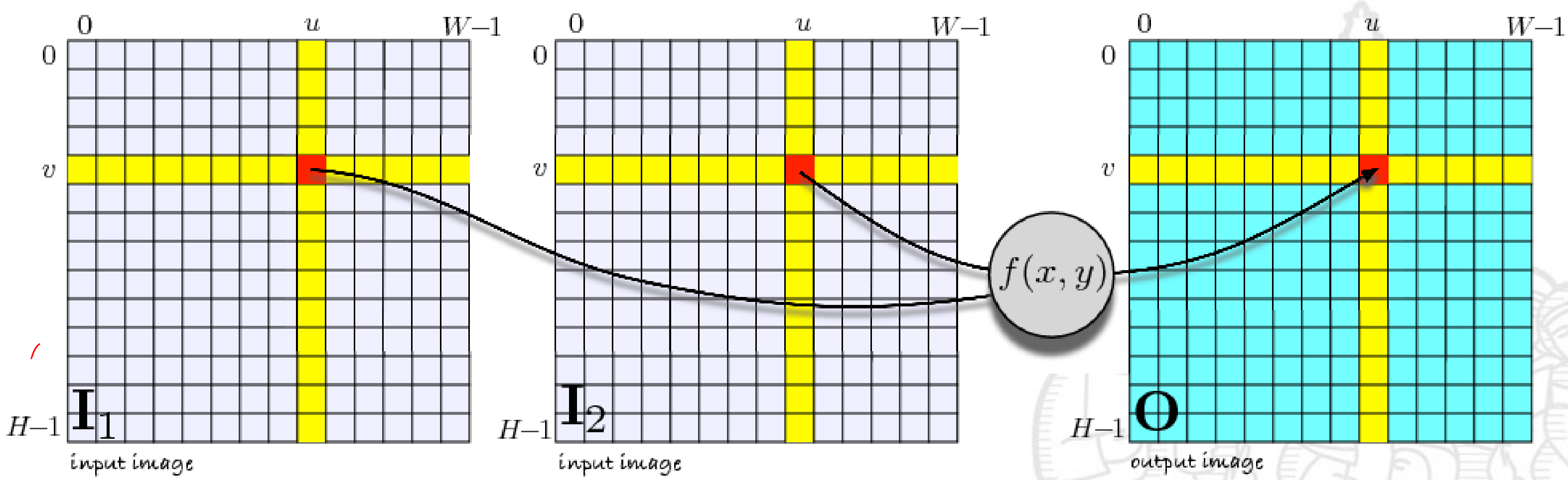
Sant'Anna
Scuola Universitaria Superiore Pisa

Code sample >

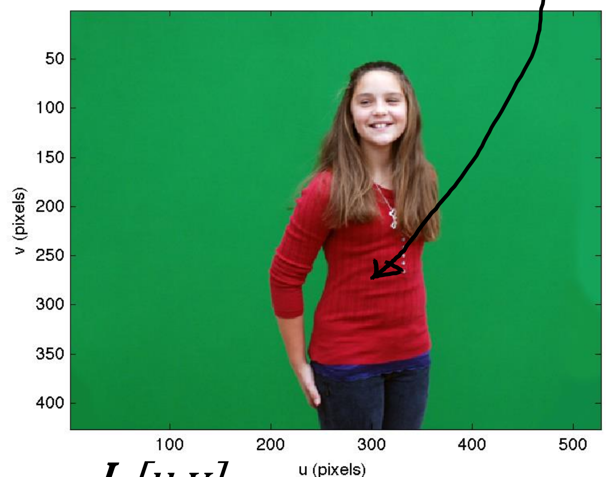
```
 %hist equalization
[n,v]=ihist(xwing_grey);
plot(v,n)
cd=zeros(length(v),1);
cd(1)=v(1)/(r*c);
    for l=2:length(v)
       cd(l)=cd(l-1)+1/(r*c)*n(l); % cumulative distribution
    end
xwing_equalized=zeros(r,c);
    for i=1:r
       for j=1:c
           xwing_equalized(i,j)=255*cd(xwing_grey(i,j)+1);    % Equalization
       end
    end
 idisp(xwing_equalized)
```

Diadic operations

$$\boldsymbol{O}[u,v] = f(\boldsymbol{I}_1[u,v], \boldsymbol{I}_2[u,v]), \qquad \forall (u,v) \in \boldsymbol{I}_1$$



Green screen



$I_1[u,v]$

$I_2[u,v]$

$I_1[u,v,2] \geq 250$

If $\boldsymbol{I}_1[u,v]$ isGreen

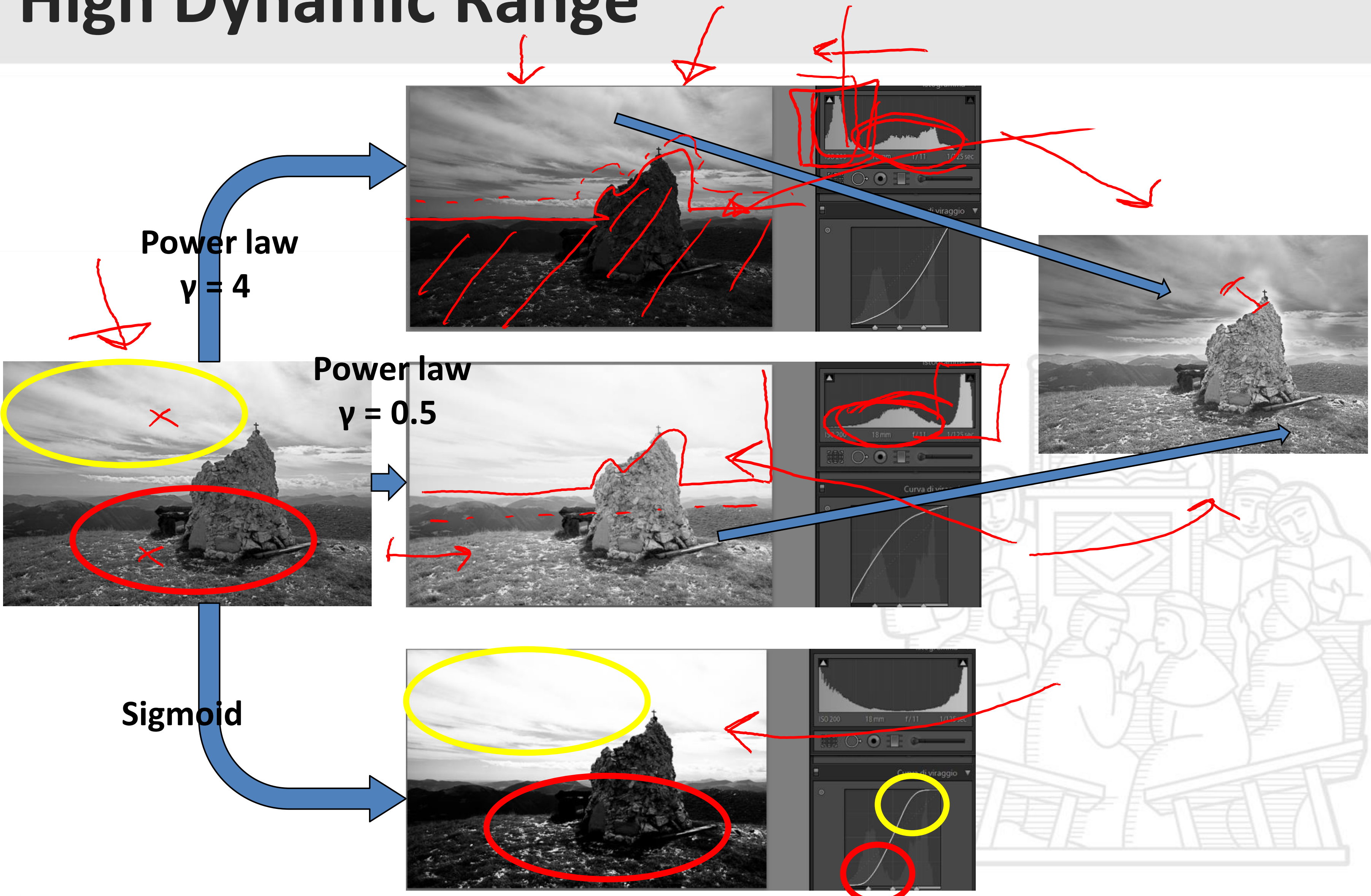
$\boldsymbol{O}[u,v] = \boldsymbol{I}_2[u,v]$

Else

$\boldsymbol{O}[u,v] = \boldsymbol{I}_1[u,v]$

$O[u,v]$

High Dynamic Range



Background subtraction

Another important diadic operation is the background subtraction to find novel elements (foreground) of a scene.

$$\boldsymbol{O}[u,v] = \boldsymbol{I}_1[u,v] - \boldsymbol{I}_2[u,v] = \boldsymbol{I}_1[u,v] - \boldsymbol{B}[u,v]$$

background

How we estimate the background $\boldsymbol{B}[u,v]$?

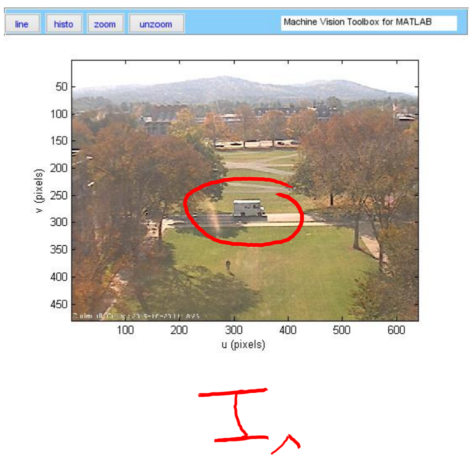
We can take a shoot when we know that only background is visible

'http://wc2.dartmouth.edu', 05:19 p.m., Rome time

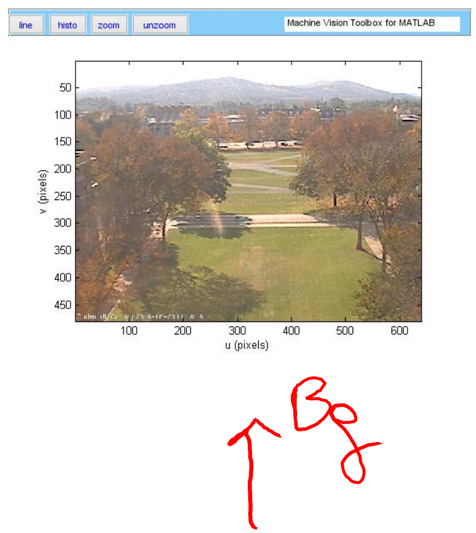
Background subtraction

'http://wc2.dartmouth.edu‘, 05:19 p.m., Rome time

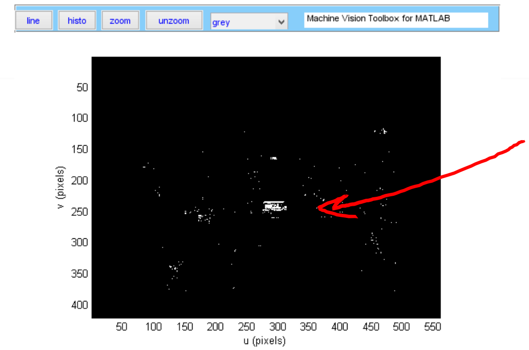
$$I_1[u,v] - B[u,v] = O[u,v]$$



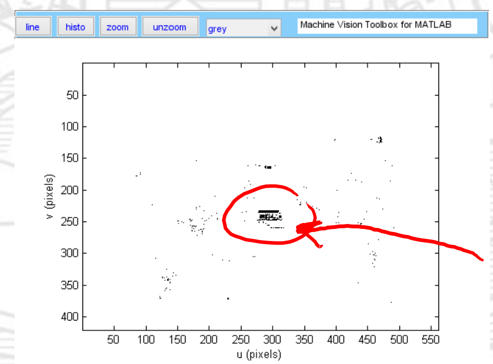
−



=



foreground

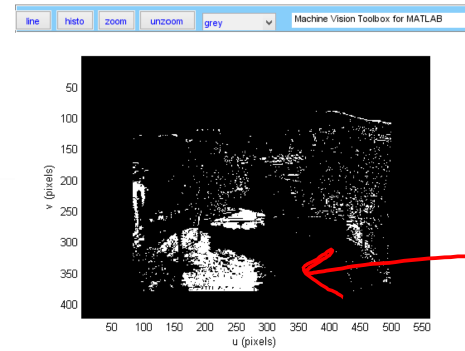


background

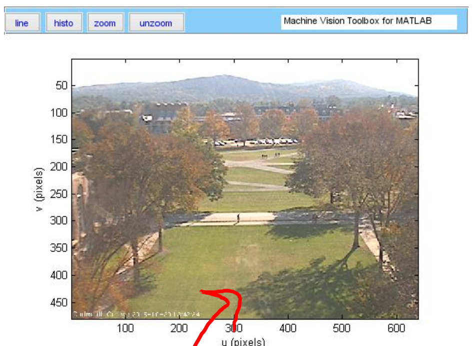
Background subtraction

'http://wc2.dartmouth.edu', 07:48 p.m., Rome time

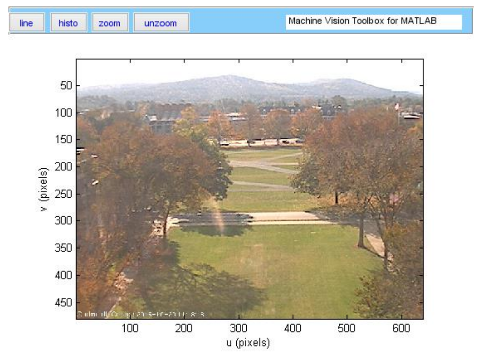
$$I_1[u,v] - B[u,v] = O[u,v]$$



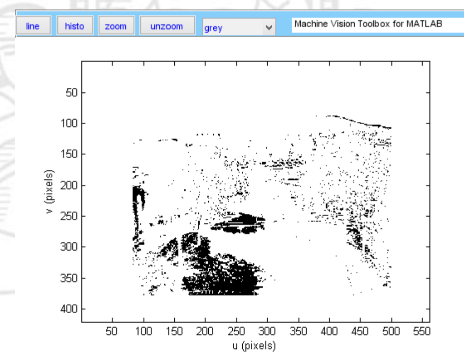
foreground



−



=



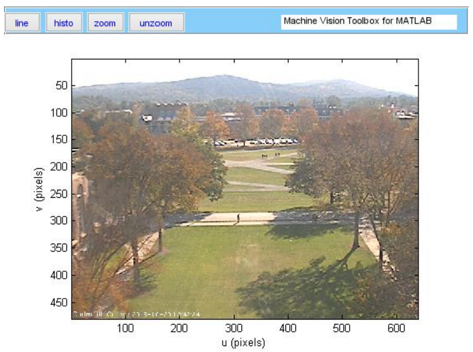
background

What went wrong?

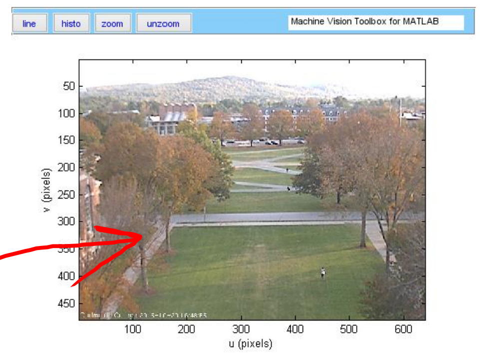
Background subtraction

'http://wc2.dartmouth.edu', 10:55 p.m., Rome time

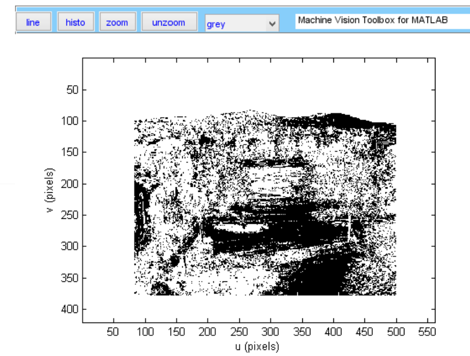
$$\boldsymbol{I}_1[u,v] - \boldsymbol{B}[u,v] = \boldsymbol{O}[u,v]$$



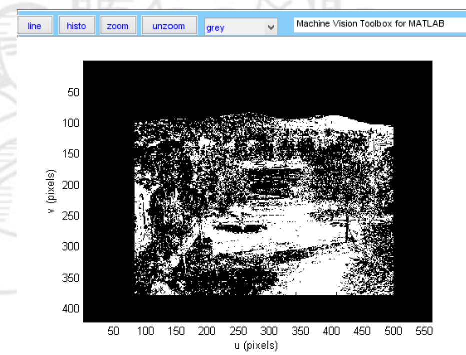
—



=



foreground



background

What went wrong?

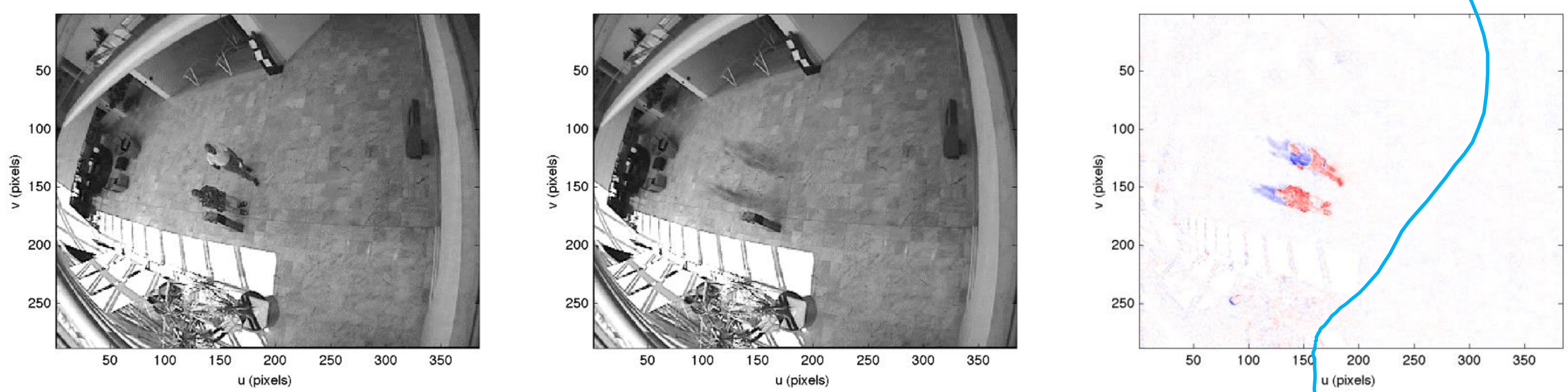
Background estimation

We require a progressive adaptation to small, persistent changes in the background.

Rather than take a static image as background, we estimated it as follow:

$$\boldsymbol{B}\langle k+1\rangle = \boldsymbol{B}\langle k\rangle + c(\boldsymbol{I}\langle k\rangle - \boldsymbol{B}\langle k\rangle)$$

$$c(x) = \begin{cases} \sigma, & x > \sigma \\ x, & -\sigma \le x \le \sigma \\ -\sigma, & x < -\sigma \end{cases}$$



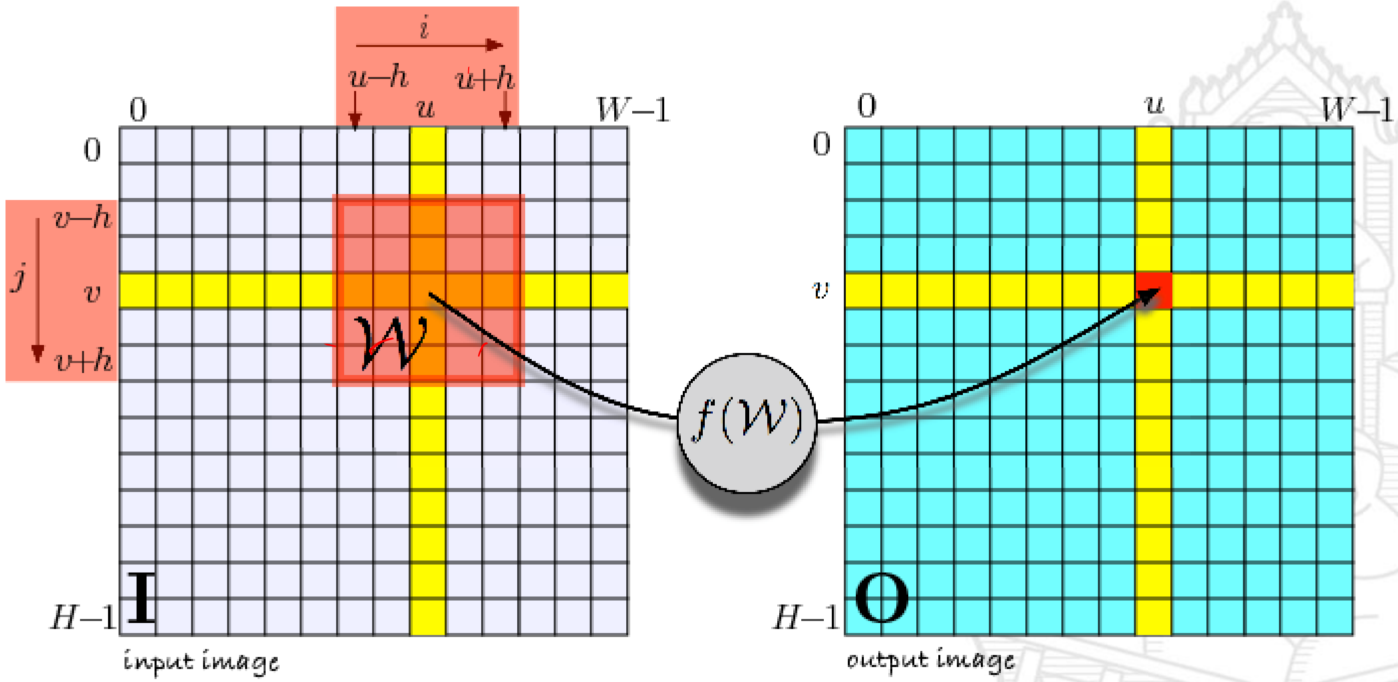
Background subtraction

Code sample >

```
% backgorund estimation
sigma=0.01;
vid = videoinput('winvideo', 1);
bg=getsnapshot(vid);
bg_small=idouble(imono(bg));
while 1
     img=getsnapshot(vid);
     img_small=idouble(imono(img));
     if isempty(img), break; end
     d=img_small-bg_small;
     d=max(min(d,sigma), -sigma);
     bg_small=bg_small+d;
     idisp(bg_small); drawnow
end
```

Spatial operation (local operators)

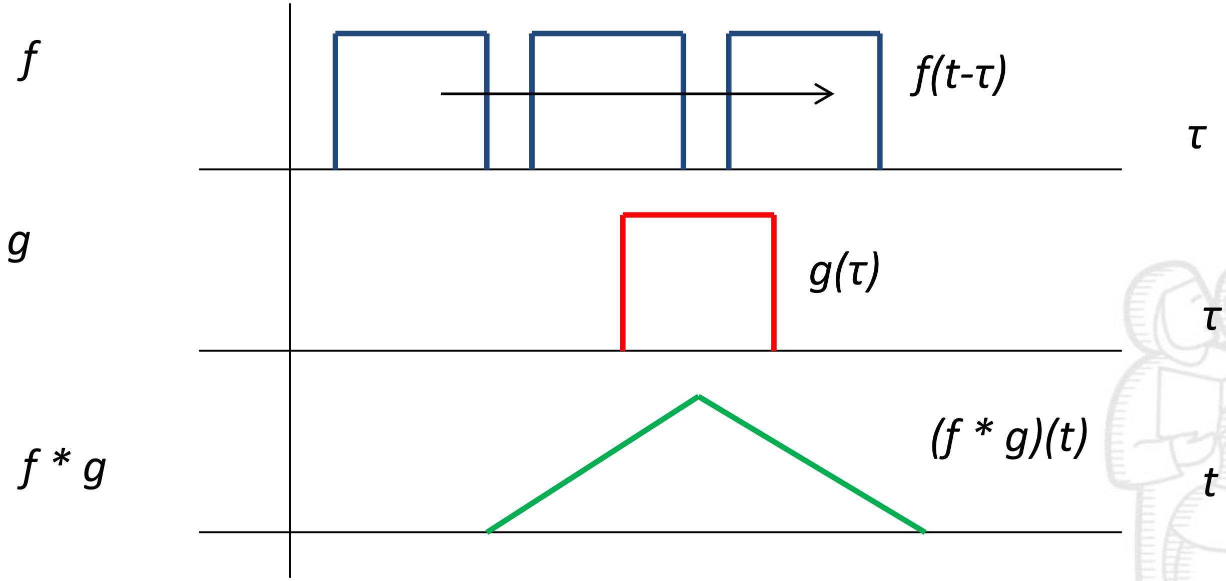
$$\boldsymbol{O}[u,v] = f(\boldsymbol{I}[u+i, v+j]), \qquad \forall (i,j) \in \boldsymbol{W}, \forall (u,v) \in \boldsymbol{I}$$

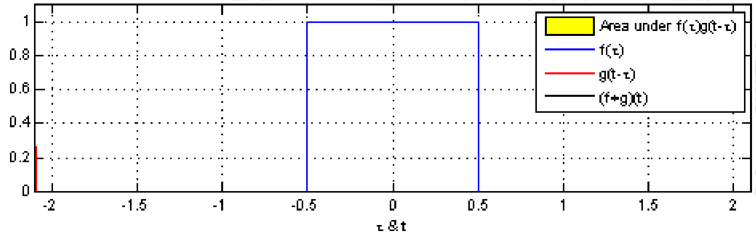


1D Convolution

One important local operator is the convolution:

$$(f * g)(t) := \int_{-\infty}^{\infty} f(t - \tau) g(\tau) d\tau$$



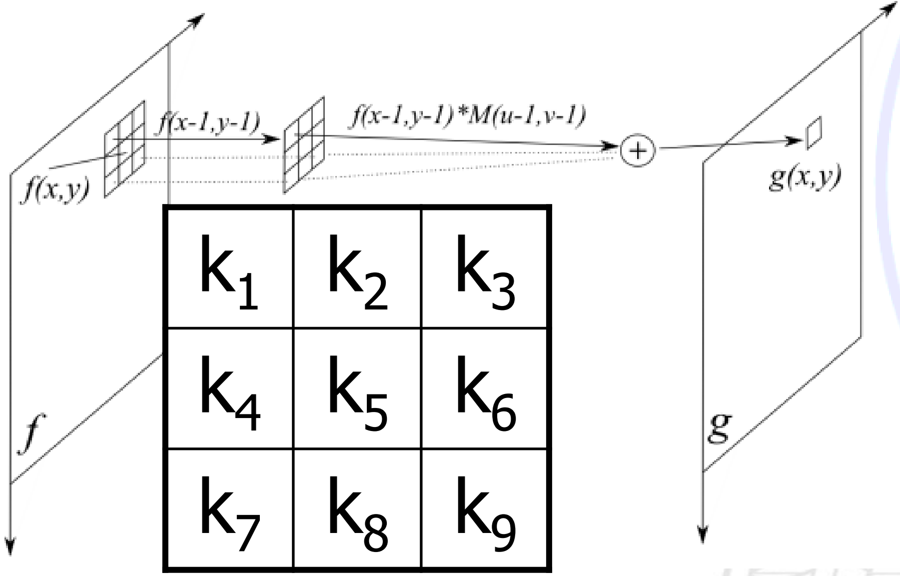


wikipedia

2D Convolution

$$\boldsymbol{O}[u,v] = \sum_{i,j \in \boldsymbol{K}} \boldsymbol{K}[i,j]\boldsymbol{I}[u-i, v-j], \qquad \forall (u,v) \in \boldsymbol{I}$$

$$\boldsymbol{O} = \boldsymbol{K} \otimes \boldsymbol{I}$$



Convolution mask

2D Convolution

kernel

·1+	·1+	·1+
·1+	·1+	·1+
·1+	·1+	·1

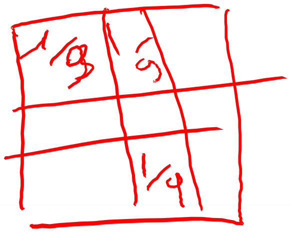
Input image

0	1	2	0	12	5	0	1
5	2	6	0	0	1	1	1
5	0	0	4	5	6	1	0
12	25	0	24	56	8	2	3
1	2	6	0	0	1	5	2
1	2	0	2	1	2	1	0
12	0	12	25	3	5	0	1
1	1	1	35	57	5	3	1

Output image

	21						

Convolution



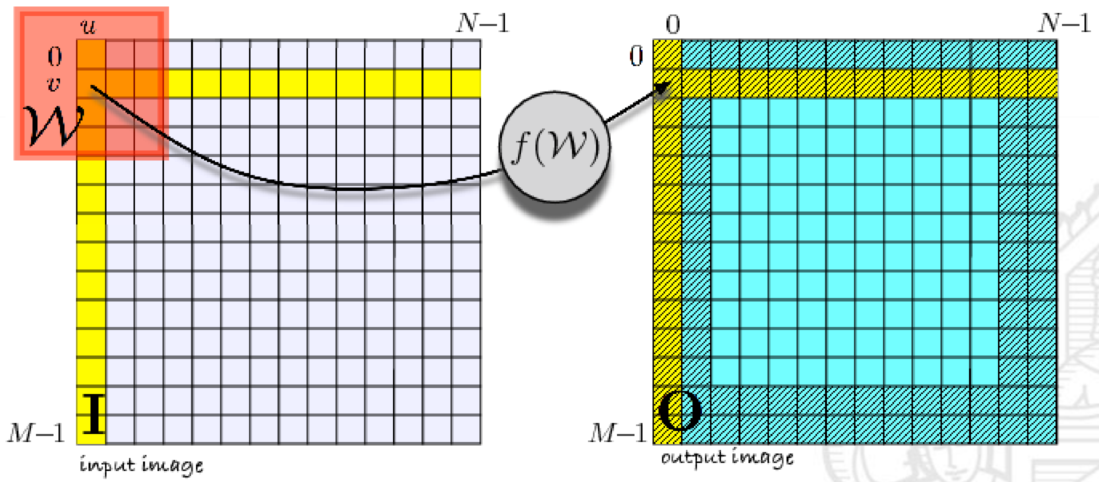
Input image

0·1+	1·1+	2·1+	0	12	5	0	1
5·1+	2·1+	6·1+	0	0	1	1	1
5·1+	0·1+	0·1	4	5	6	1	0
12	25	0	24	56	8	2	3
1	2	6	0	0	1	5	2
1	2	0	2	1	2	1	0
12	0	12	25	3	5	0	1
1	1	1	35	57	5	3	1

Output image

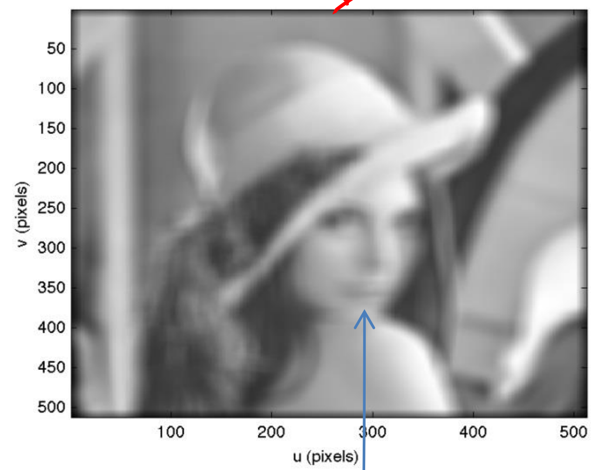
	21	15					

Boundary effect



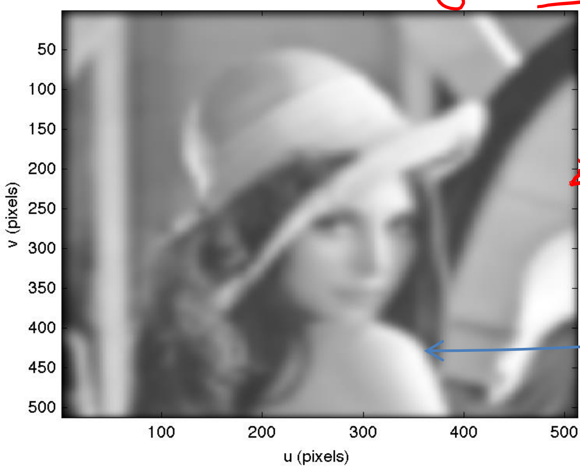
- Duplicate
- All black
- Reduce size
- …

Smoothing



$$K = ones(21,21)/21^2$$

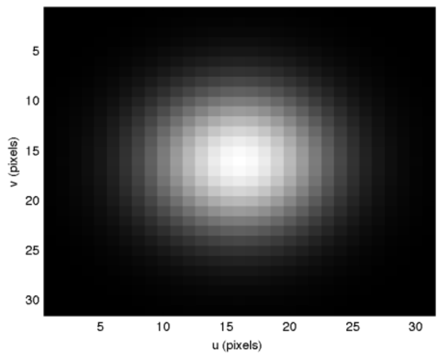
$$O = K \otimes I$$

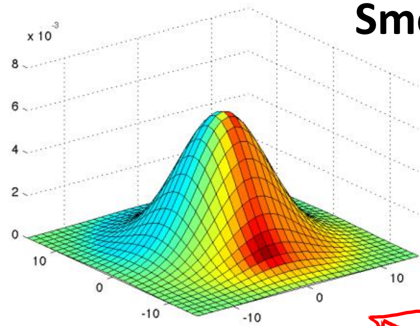


$$G[u,v] = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

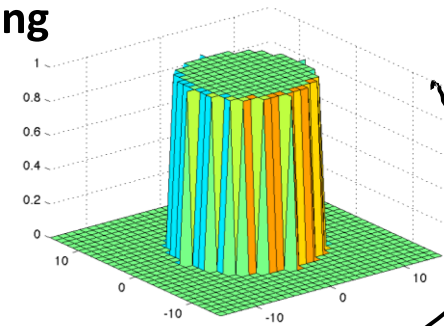
Kernel examples

Smoothing



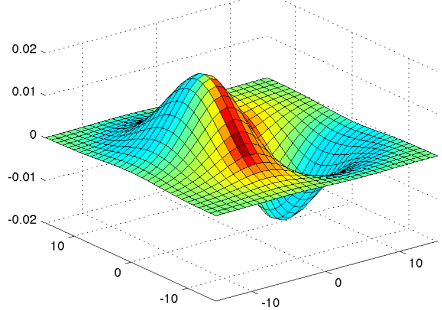


Gaussian



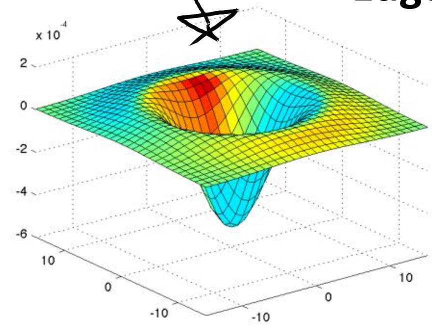
Top hat

Gradient

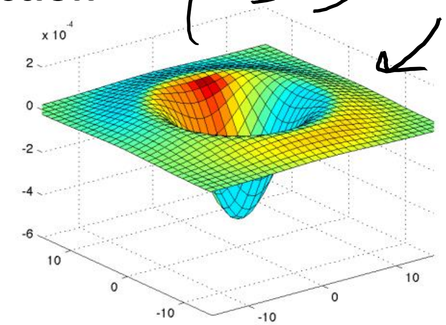


Derivative of Gaussian (DoG)

Edge detection

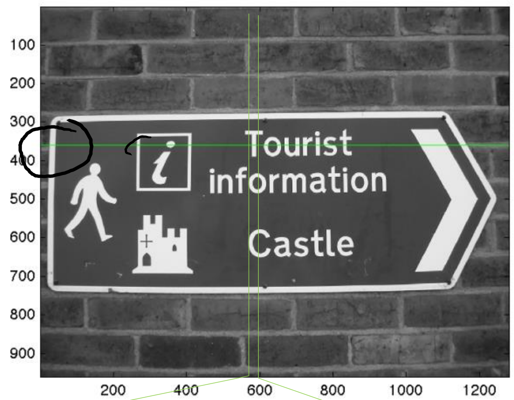


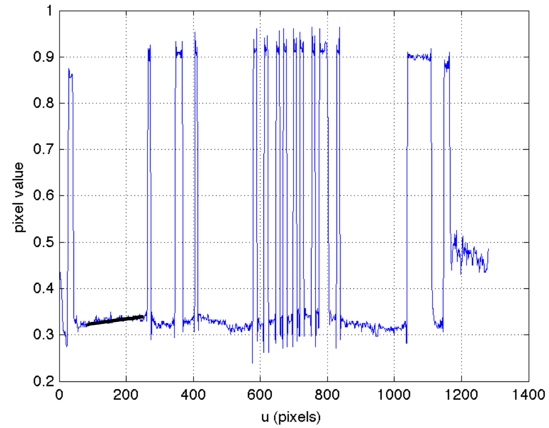
Laplacian of Gaussian (LoG)



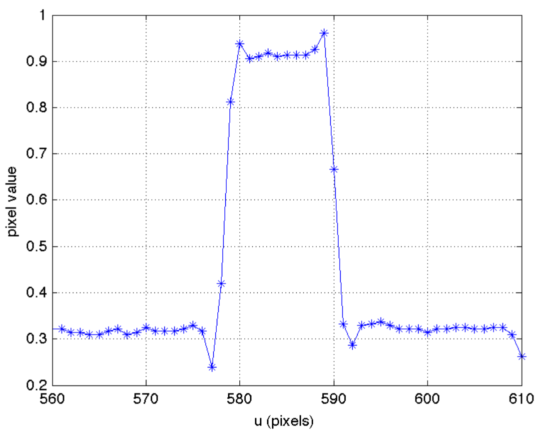
Difference of Gaussian (DiffG)

Edge detection





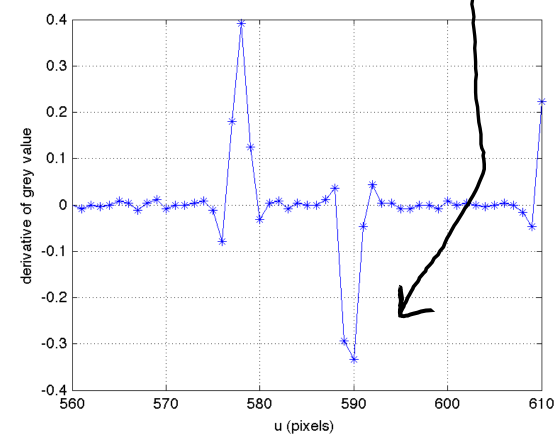
Horizontal profile of the image at *v*=360



$$p'[u] = p[u] - p[u-1]$$

$$p'[u] = \frac{1}{2}(p[u+1] - p[u-1])$$

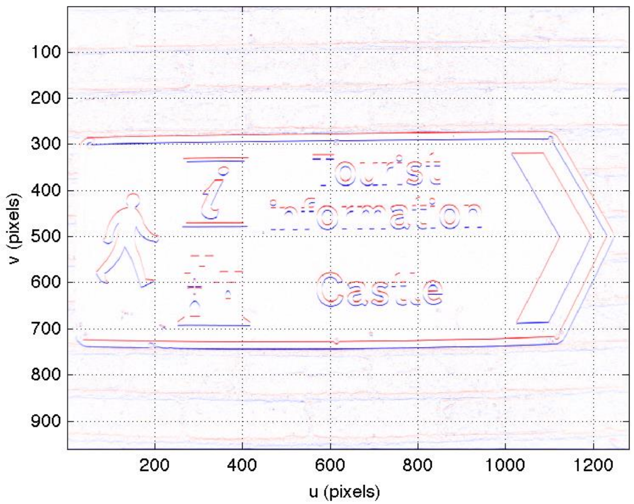
$$\boldsymbol{K} = \left[-\frac{1}{2} \;\; 0 \;\; \frac{1}{2}\right]$$



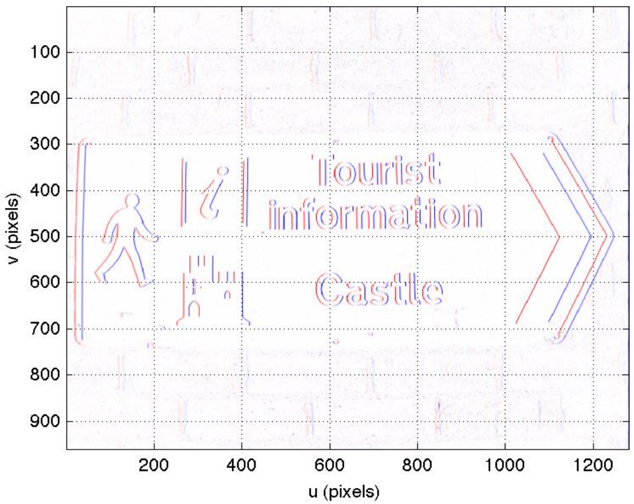
Gradient computation

Common convolution kernel: Sobel, Prewitt, Roberts, …

Sobel $\boldsymbol{D}_v = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ $\boldsymbol{D}_u = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

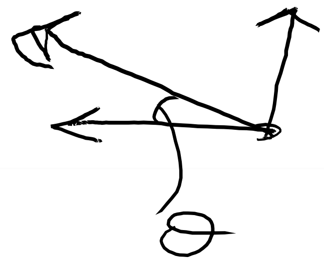


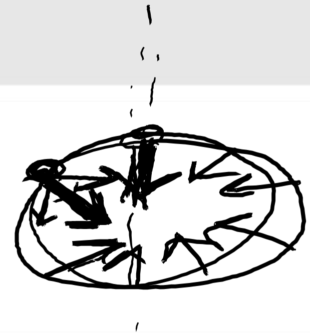
$\boldsymbol{I}_v = \boldsymbol{D}_v \otimes \boldsymbol{I}$



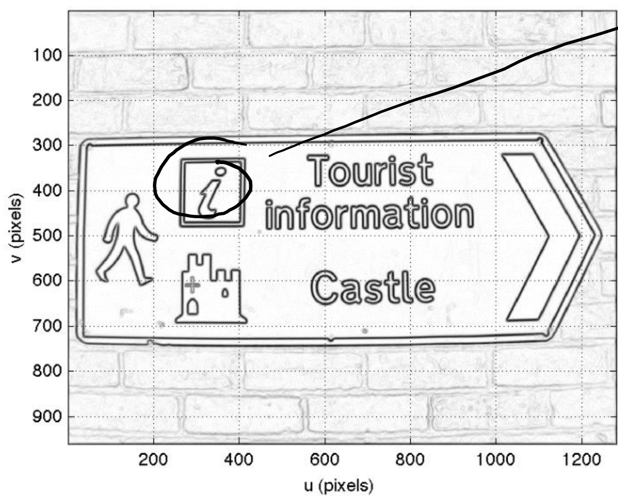
$\boldsymbol{I}_u = \boldsymbol{D}_u \otimes \boldsymbol{I}$

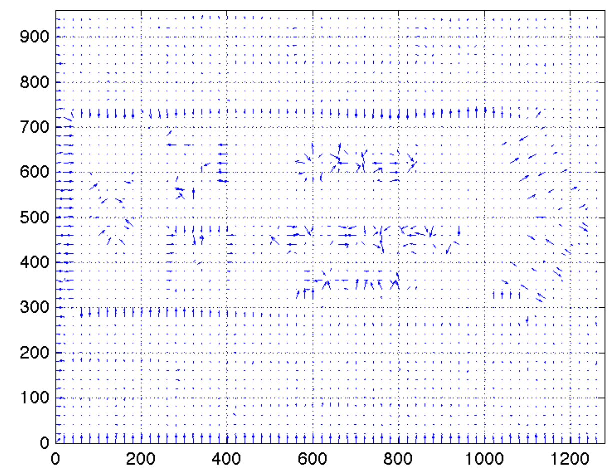
Direction and magnitude





$$m = \sqrt{\boldsymbol{I}_v^2 + \boldsymbol{I}_u^2} \qquad \theta = \mathrm{atan}(\boldsymbol{I}_v, \boldsymbol{I}_u)$$





Noise amplification

Derivative amplifies high-frequency noise. So, firstly we can smooth the image, after that we can take the derivative:

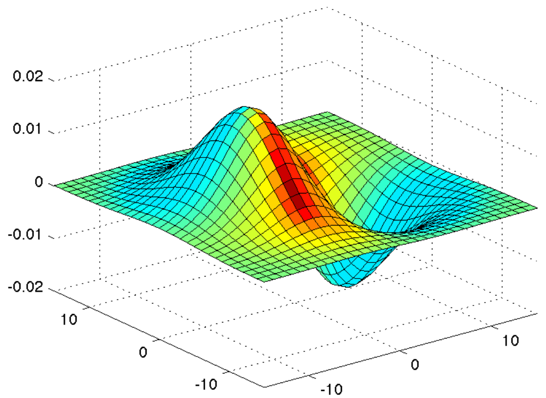
$$I_u = D_u \otimes (G \otimes I)$$

Associative property:

$$I_u = \underbrace{(D_u \otimes G)}_{} \otimes I$$

Derivative of Gaussian (DoG)

$$G_u = -\frac{u}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



Derivative of Gaussian (DoG)

<<DoG acts as a bandpass filter!>>

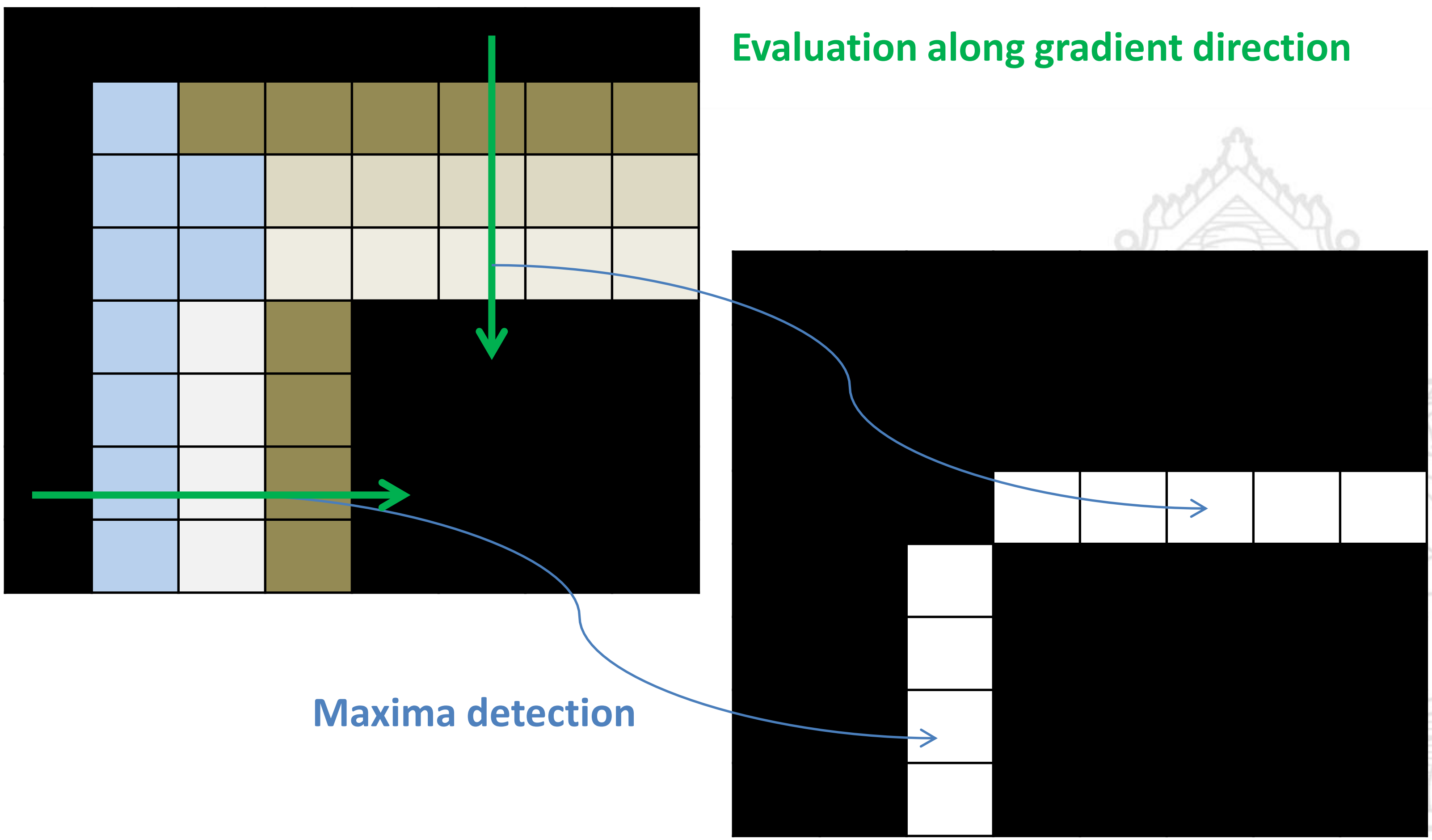
Canny edge detection

The algorithm is based on a few steps:

1. Gaussian filtering
2. Gradient intensity and direction
3. non-maxima suppression (edge thinning)
4. hysteresis threshold

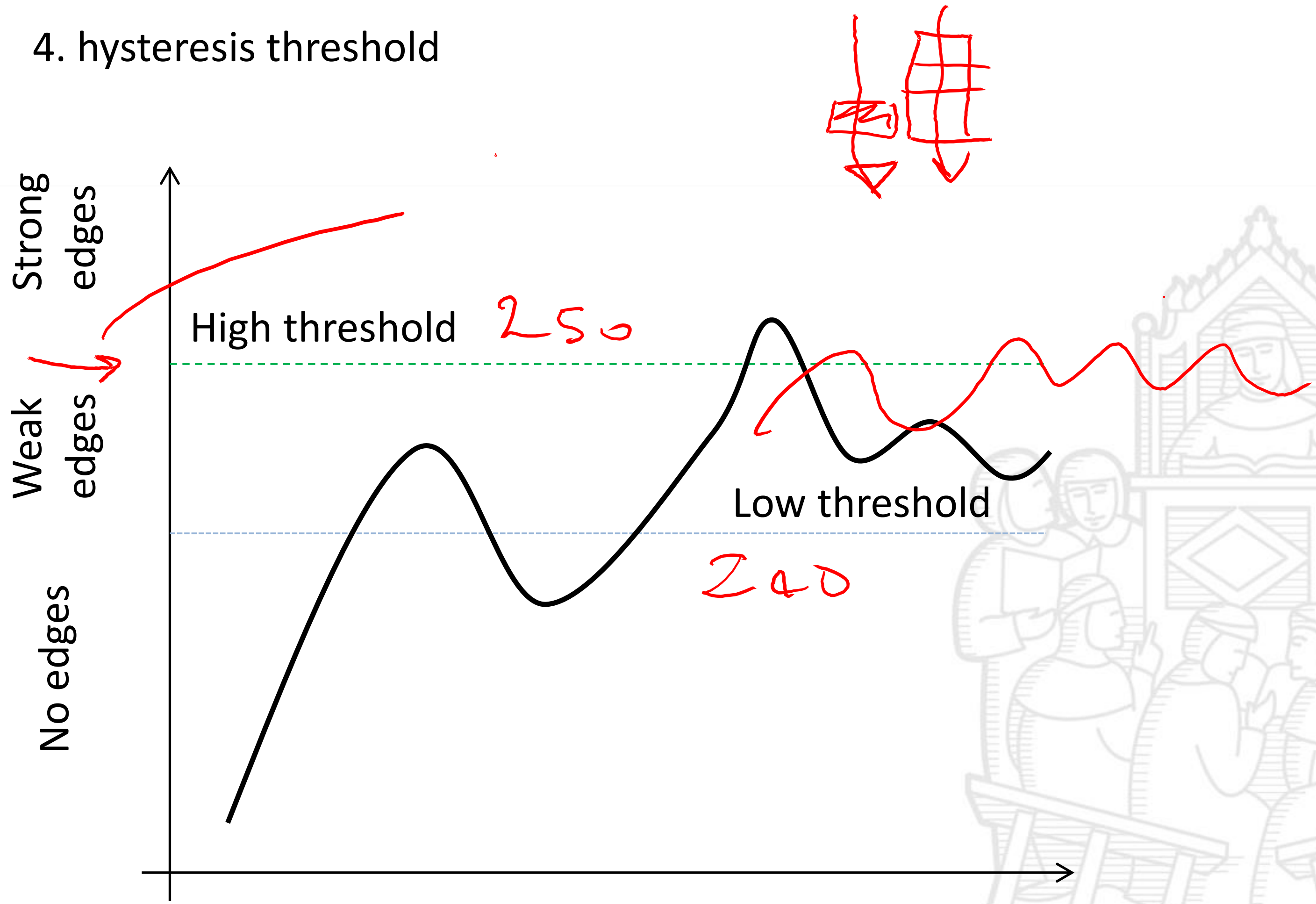
Canny edge detection

3. Non local maxima suppression

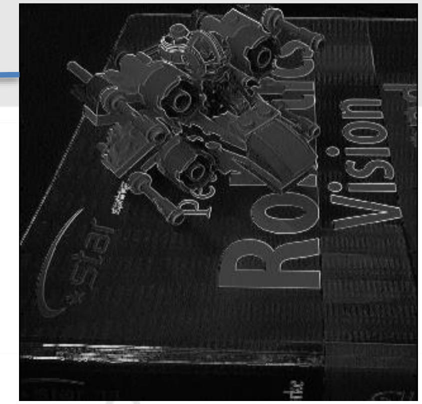


Canny edge detection

4. hysteresis threshold

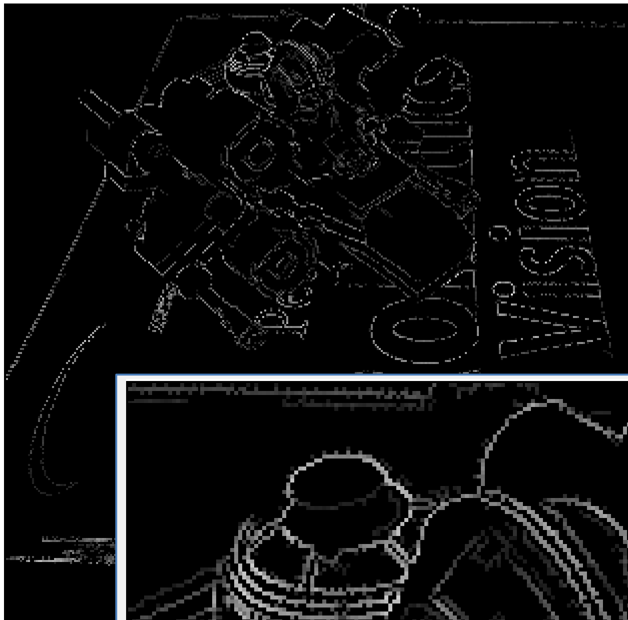


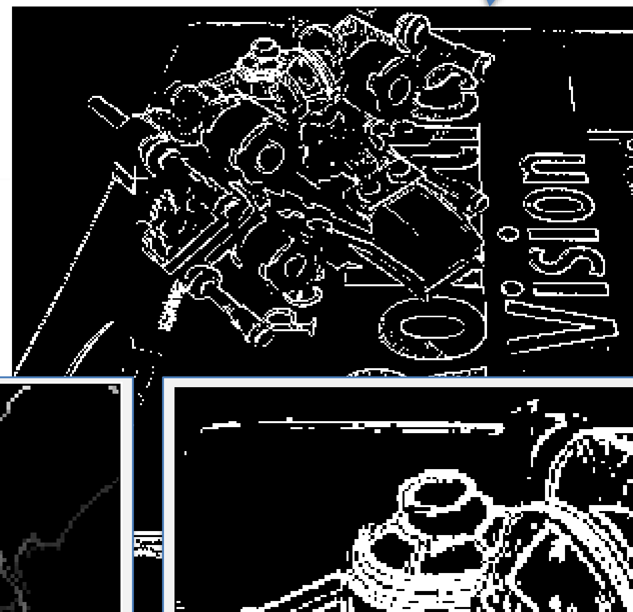
Magnitude of the gradient

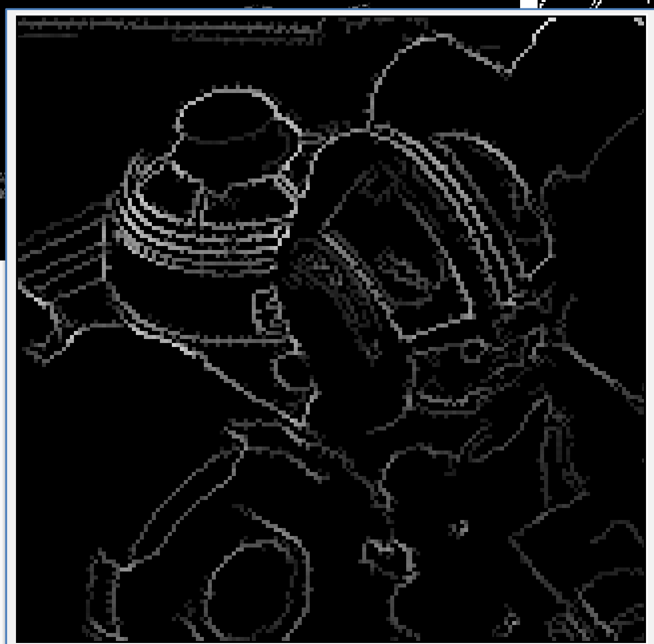


Thresholding

canny







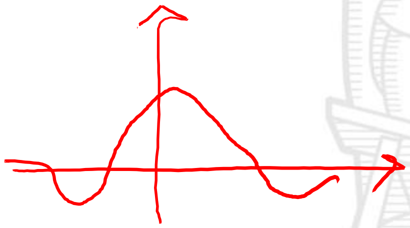
Edge detection

Alternative approach is to use second derivative and to find where there is a zero

Laplacian operator

$$\nabla \boldsymbol{I}^2 = \frac{\partial^2 \boldsymbol{I}}{\partial u^2} + \frac{\partial^2 \boldsymbol{I}}{\partial v^2} = \boldsymbol{I}_{uu} + \boldsymbol{I}_{vv} = \boldsymbol{L} \otimes \boldsymbol{I}$$

$$\boldsymbol{L} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$



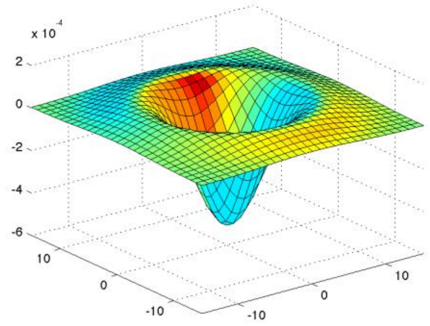
Noise sensitivity

Again, derivative amplifies high-frequency noise. So firstly we can smooth the image, after that we take the derivative:

$$\boldsymbol{L} \otimes (\boldsymbol{G} \otimes \boldsymbol{I}) = \underbrace{(\boldsymbol{L} \otimes \boldsymbol{G})}_{\text{Laplacian of Gaussian (LoG)}} \otimes \boldsymbol{I}$$

Laplacian of Gaussian
(LoG)

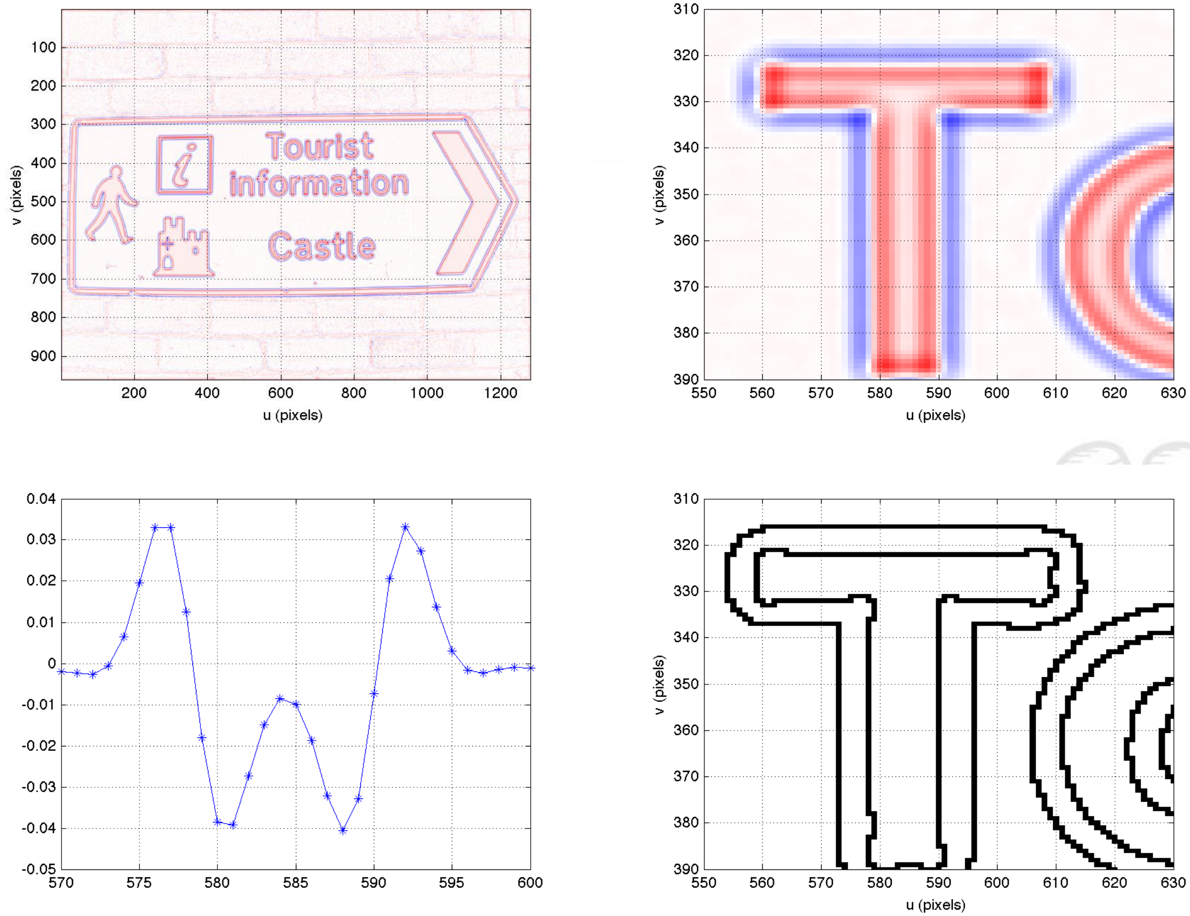
$$\boldsymbol{LoG}(u,v) = \frac{1}{\pi\sigma^4}\left(\frac{u^2+v^2}{2\sigma^2} - 1\right)e^{-\frac{u^2+v^2}{2\sigma^2}}$$



Laplacian of Gaussian
(LoG)

Marr-Hildreth operator or the Mexican hat kernel

Edge detection



Gradient and Laplacian

Example

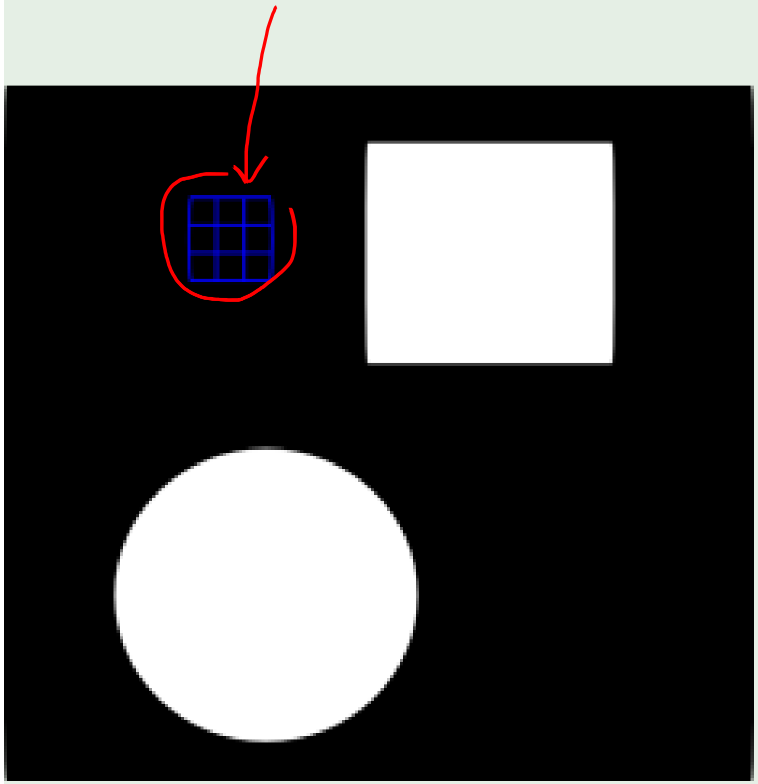


Image window:

$f(x,y) =$

0	0	0
0	0	0
0	0	0

$\nabla^2 =$

0	-1	0
-1	4	-1
0	-1	0

$G_x =$

-1	0	1
-2	0	2
-1	0	1

Products are:

$$\nabla^2 \otimes f(x,y) = 0$$

$$G_x \otimes f(x,y) = 0$$

Gradient and Laplacian

Example

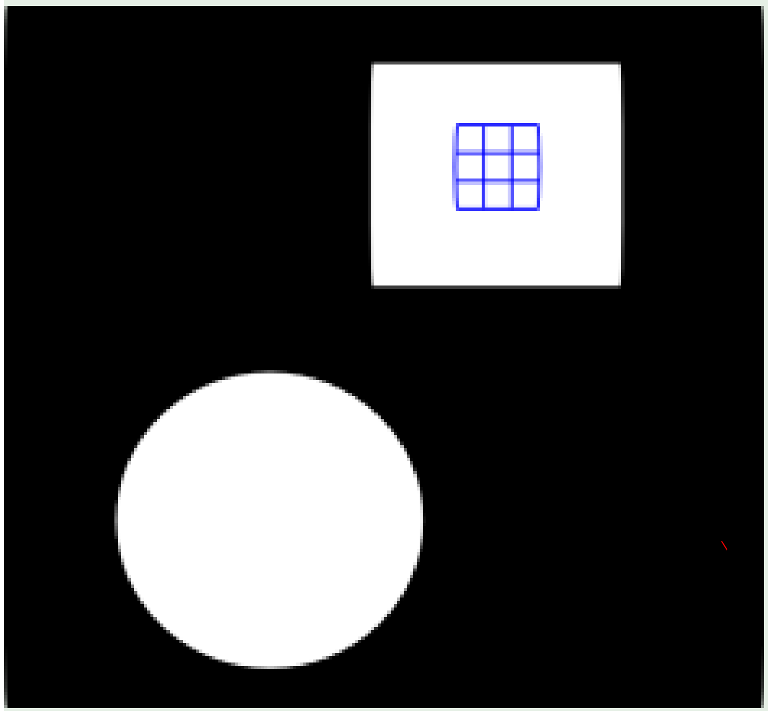


Image window:

$f(x, y)$ =

1	1	1
1	1	1
1	1	1

∇^2 =

0	-1	0
-1	4	-1
0	-1	0

G_x =

-1	0	1
-2	0	2
-1	0	1

Products are:

$$\nabla^2 \otimes f(x, y) = 0$$

$$G_x \otimes f(x, y) = 0$$

Gradient and Laplacian

Example

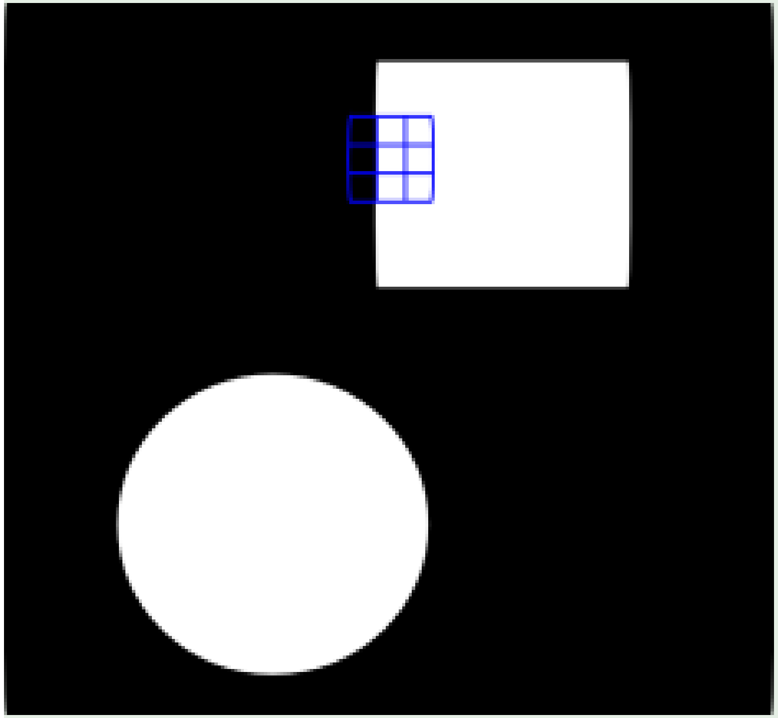


Image window:

$f(x, y) =$

0	1	1
0	1	1
0	1	1

$\nabla^2 =$

0	-1	0
-1	4	-1
0	-1	0

$G_x =$

-1	0	1
-2	0	2
-1	0	1

Products are:

$$\nabla^2 \otimes f(x, y) = 1$$

$$G_x \otimes f(x, y) = 4$$

Gradient and Laplacian

Example

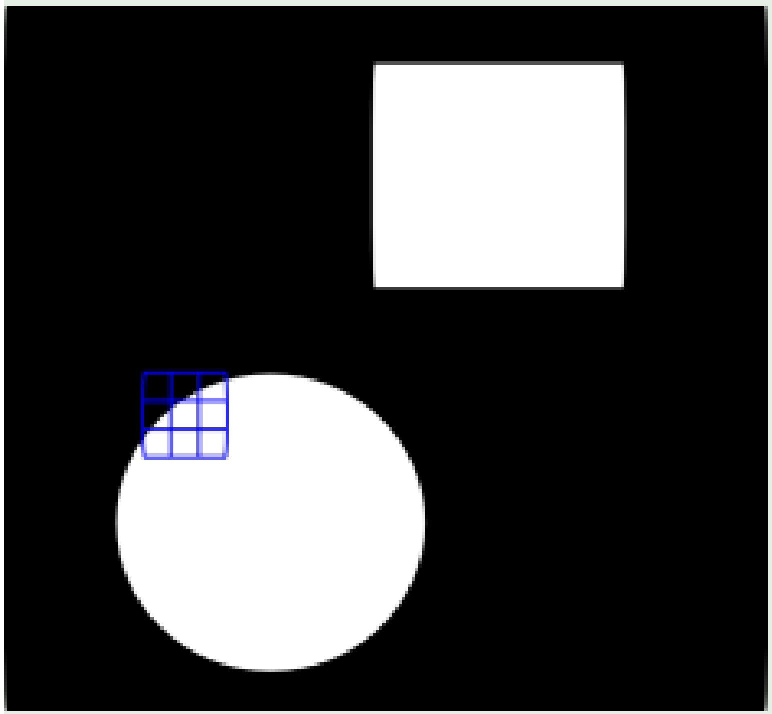


Image window:

$f(x, y)$ =

0	0	1
0	1	1
1	1	1

∇^2 =

0	-1	0
-1	4	-1
0	-1	0

G_x =

-1	0	1
-2	0	2
-1	0	1

Products are:

$$\nabla^2 \otimes f(x, y) = 2$$

$$G_x \otimes f(x, y) = 3$$

Gradient and Laplacian

Example

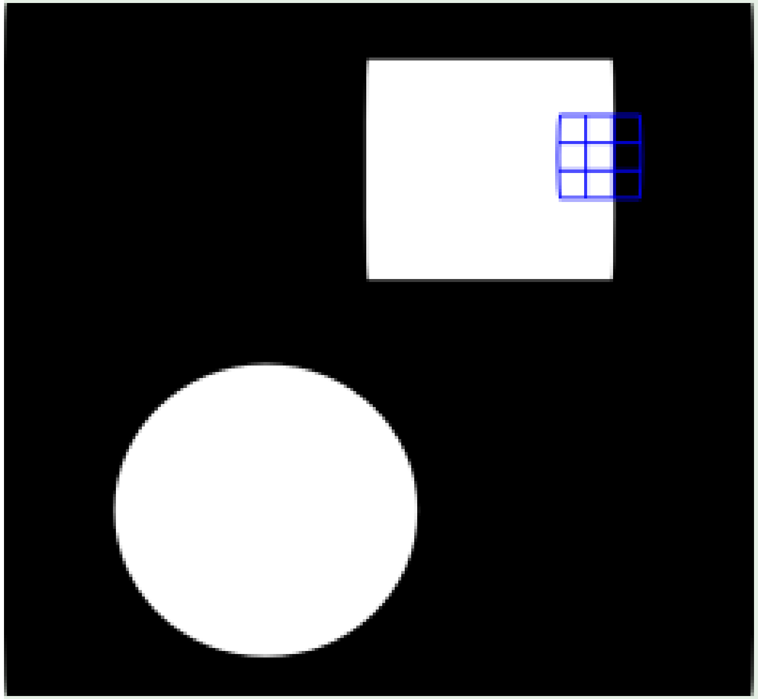


Image window:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 1 & 1 & 0 \\ \hline 1 & 1 & 0 \\ \hline 1 & 1 & 0 \\ \hline \end{array}$$

$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

Products are:

$$\nabla^2 \otimes f(x,y) = 1$$

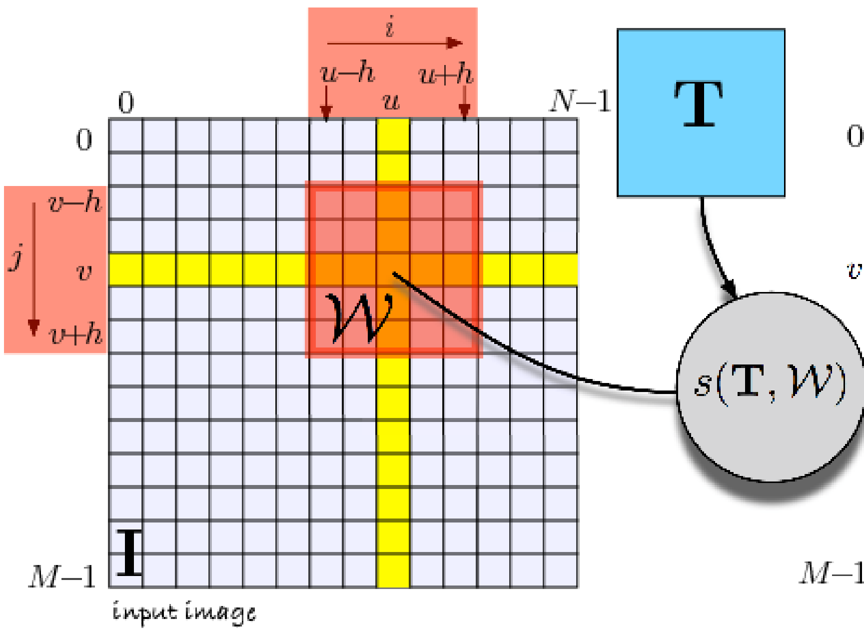
$$G_x \otimes f(x,y) = -4$$

Code sample >

```
% denoising/edge detection
dx=[-1 0 1;-2 0 1; -1 0 1];
dy=[-1 -2 -1;0 0 0;1 2 1];
K=kgauss(3);
K1=ones(19,19).*1/(19*19);
xwingDenoisMean=iconv(K1,xwing_grey);
idisp(xwingDenoisMean)
xwingDenoisGaus=iconv(K,xwing_grey);
idisp(xwingDenois)
xwingIx=iconv(dx,xwing_grey);
idisp(xwingIx)
xwingIy=iconv(dy,xwing_grey);
idisp(xwingIy)
magnGrad=sqrt(xwingIx.^2+xwingIy.^2);
idisp(magnGrad)
edgeGrad=magnGrad>250;

edgeLapl=iconv(klog(2),xwing_grey);
idisp(iint(edgeLapl)>250);

edgeLapl=iconv(klog(1),xwing_grey);
idisp(iint(edgeLapl)>250);

edgeLapl=iconv(klog(3),xwing_grey);
idisp(iint(edgeLapl)>250);
```

Template matching

$$\boldsymbol{O}[u, v] = s(\boldsymbol{T}, W), \qquad \forall (u, v) \in \boldsymbol{I}$$



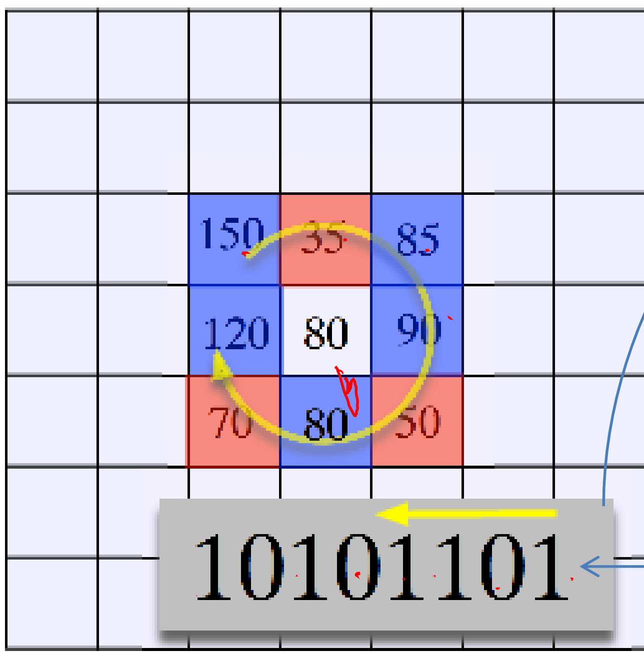
Template matching

Similarity measures

Sum of absolute differences	
SAD	$s = \sum_{(u,v)\in \boldsymbol{I}} \lvert \boldsymbol{I}_1[u,v] - \boldsymbol{I}_2[u,v] \rvert$
ZSAD	$s = \sum_{(u,v)\in \boldsymbol{I}} \lvert (\boldsymbol{I}_1[u,v] - \bar{\boldsymbol{I}}_1) - (\boldsymbol{I}_2[u,v] - \bar{\boldsymbol{I}}_2) \rvert$
Sum of squared differences	
SSD	$s = \sum_{(u,v)\in \boldsymbol{I}} (\boldsymbol{I}_1[u,v] - \boldsymbol{I}_2[u,v])^2$
ZSSD	$s = \sum_{(u,v)\in \boldsymbol{I}} \left((\boldsymbol{I}_1[u,v] - \bar{\boldsymbol{I}}_1) - (\boldsymbol{I}_2[u,v] - \bar{\boldsymbol{I}}_2) \right)^2$
Cross correlation	
NCC	$s = \dfrac{\sum_{(u,v)\in \boldsymbol{I}} \boldsymbol{I}_1[u,v] \cdot \boldsymbol{I}_2[u,v]}{\sqrt{\sum_{(u,v)\in \boldsymbol{I}} \boldsymbol{I}_1^2[u,v] \cdot \sum_{(u,v)\in \boldsymbol{I}} \boldsymbol{I}_2^2[u,v]}}$
ZNCC	$s = \dfrac{\sum_{(u,v)\in \boldsymbol{I}} (\boldsymbol{I}_1[u,v] - \bar{\boldsymbol{I}}_1) \cdot (\boldsymbol{I}_2[u,v] - \bar{\boldsymbol{I}}_2)}{\sqrt{\sum_{(u,v)\in \boldsymbol{I}} (\boldsymbol{I}_1[u,v] - \bar{\boldsymbol{I}}_1)^2 \cdot \sum_{(u,v)\in \boldsymbol{I}} (\boldsymbol{I}_2[u,v] - \bar{\boldsymbol{I}}_2)^2}}$

Non-parametric similarity measures

Census



$$s(x) = \begin{cases} 1, if\ x > R \\ 0, otherwise \end{cases}$$

Census representation

Hamming distance

Rank transform = 5

Non-parametric similarity measures

Rank transform is more compact but does not encode position information

50	10	205
1	25	2
102	250	240

Census: 01110101

Rank: 5

Hamming distance: 6!

10	26	2
101	25	202
1	250	214

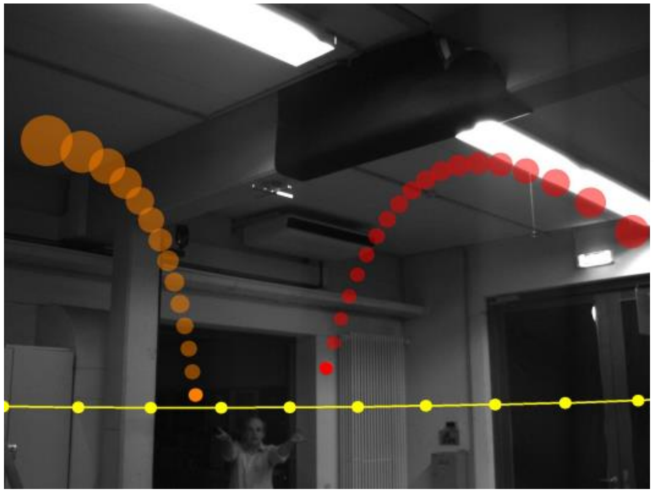
Census: 10111010

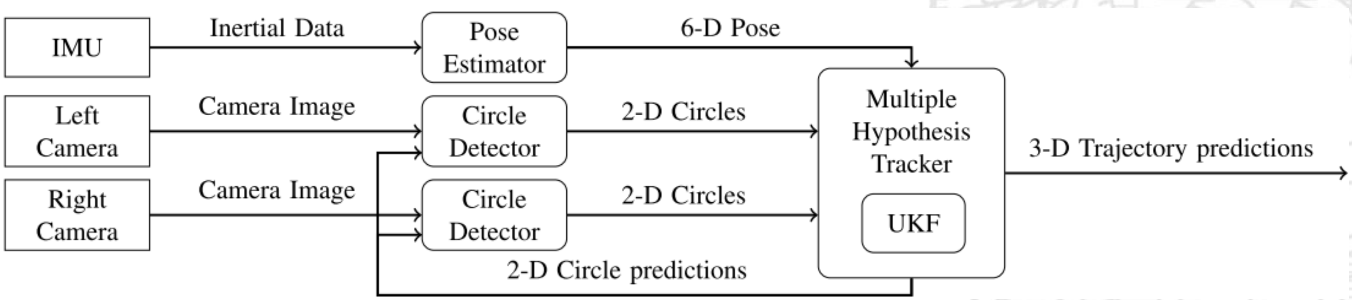
Rank: 5

Non-linear operators

- Variance measure (on windows): Edge detection
- Median filter: noise removal
- Rank transform: non-local maxima suppression

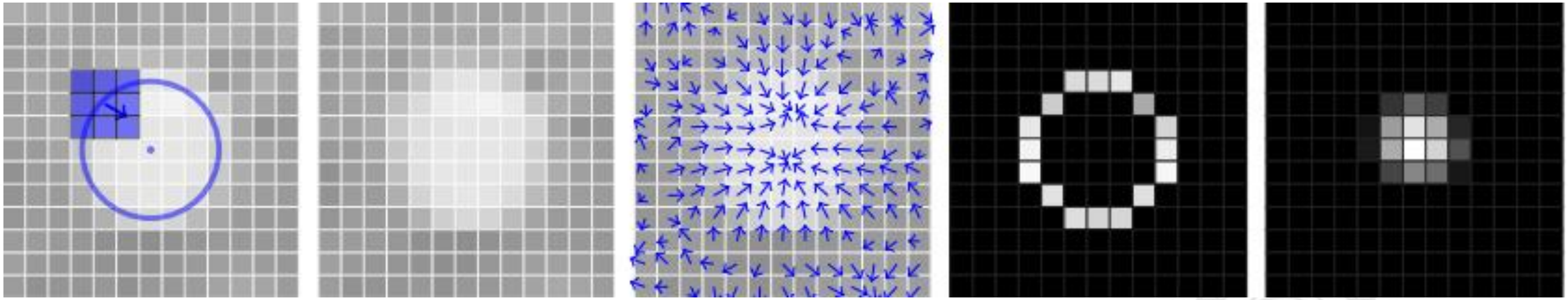
Example DLR





Oliver Birbach, Udo Frese and Berthold Bauml, (2011) ‘Realtime Perception for Catching a Flying Ball with a Mobile Humanoid’

Example DLR



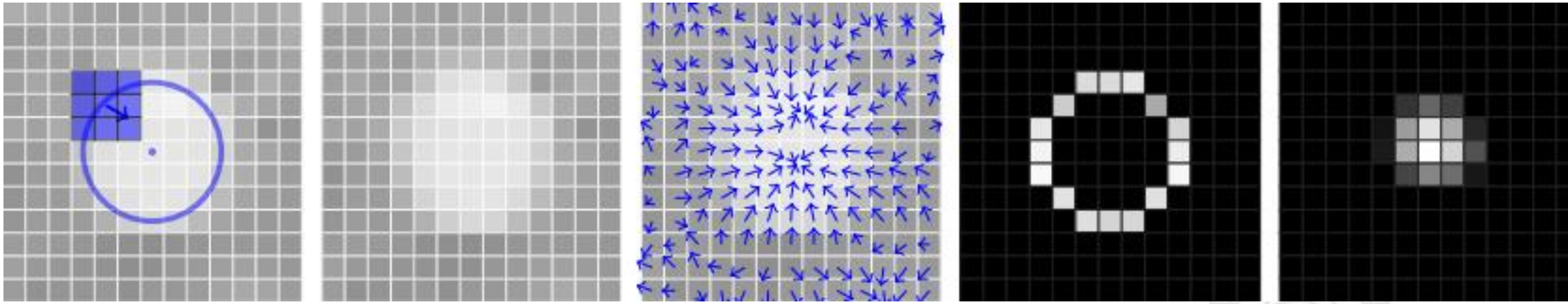
$$C = \frac{\sqrt{2}\left(\begin{pmatrix}+1 & 0 & -1\\ +2 & 0 & -2\\ +1 & 0 & -1\end{pmatrix} * I,\ \begin{pmatrix}+1 & +2 & +1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix} * I\right)^T}{\sqrt{16\begin{pmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{pmatrix} * I^2 - \left(\begin{pmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{pmatrix} * I\right)^2 + \varepsilon^2}}$$

$$R(x,y,\alpha) = \left(\begin{pmatrix}\cos\alpha\\ \sin\alpha\end{pmatrix} \cdot C(x,y)\right)^2 = |C(x,y)|^2 \cdot \cos^2\delta$$

$$CR(x_c,y_c,r) = \frac{1}{2\pi}\int_{\alpha=0}^{2\pi} R(x_c + r\cos\alpha, y_c + r\sin\alpha, \alpha)\,d\alpha$$

Sant'Anna
Scuola Universitaria Superiore Pisa

Example DLR



Filtraggio con Sobel

$$C = \frac{\sqrt{2}\left(\begin{pmatrix}+1&0&-1\\+2&0&-2\\+1&0&-1\end{pmatrix}*I,\ \begin{pmatrix}+1&+2&+1\\0&0&0\\-1&-2&-1\end{pmatrix}*I\right)^T}{\sqrt{16\begin{pmatrix}1&2&1\\2&4&2\\1&2&1\end{pmatrix}*I^2-\left(\begin{pmatrix}1&2&1\\2&4&2\\1&2&1\end{pmatrix}*I\right)^2+\varepsilon^2}}$$

Normalizzazione rispetto alla varianza locale

$$R(x,y,\alpha) = \left(\begin{pmatrix}\cos\alpha\\ \sin\alpha\end{pmatrix}\cdot C(x,y)\right)^2 = |C(x,y)|^2\cdot\cos^2\delta$$

$$CR(x_c,y_c,r) = \frac{1}{2\pi}\int_{\alpha=0}^{2\pi} R(x_c+r\cos\alpha, y_c+r\sin\alpha,\alpha)\,d\alpha$$