Él-, sarok-, minta detektálás és követés

Dr. Loványi István
BME-IIT
2014 április

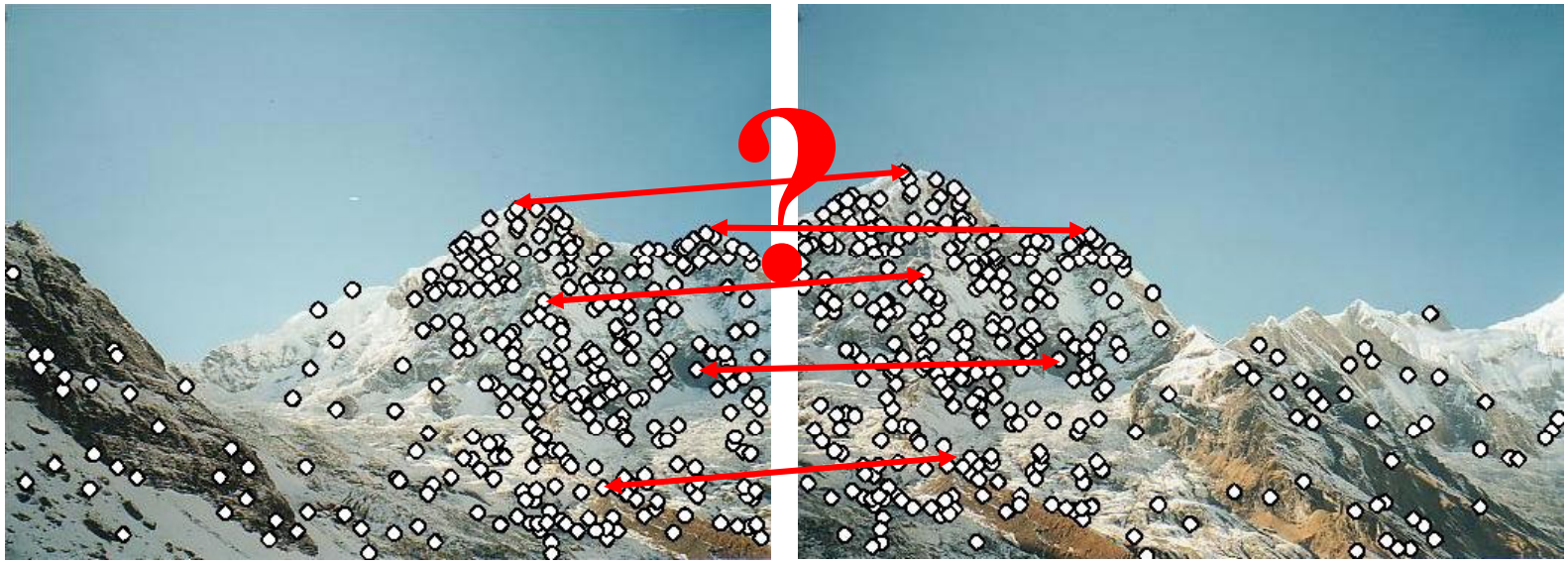
Megoldandó feladatok

- Jó jellemző felismerés (2D)
- Jó jellemzők követése (2D)

Hol van a képen … ?

- 3D rekonstrukció (3D)
  - önkalibráció
  - 3D struktúra
  - Mozgás

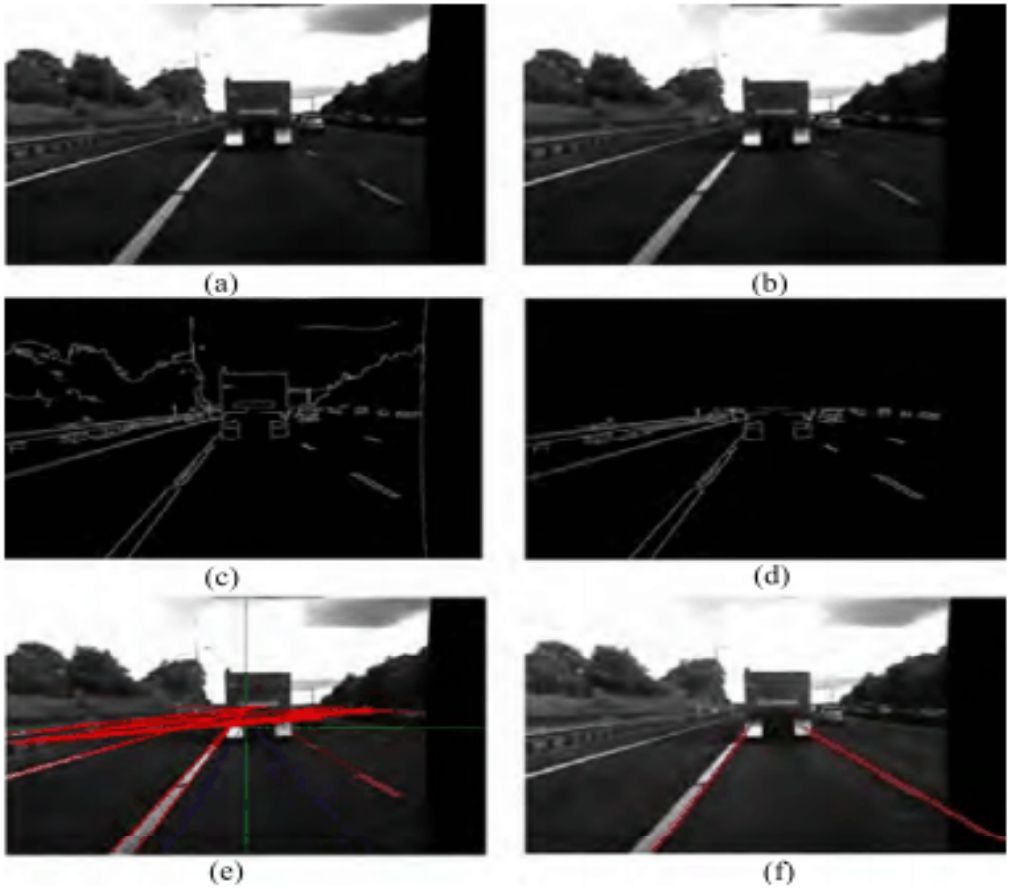
Hol van a világban … ?



Képjellemzők meghatározása

- Mit is mérjünk?
- Hogyan?
- Jellemzők „jóságának” összehasonlítása

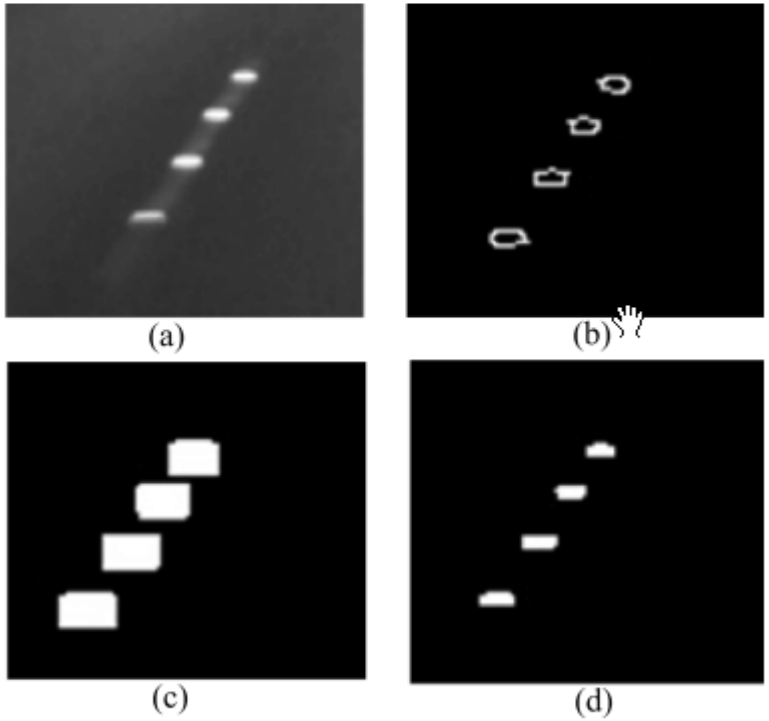
Példa: Sávdetektálás
a feldolgozás főbb lépései



a) bemeneti kép
b) szűrés
c) éldetektálás
d) AOI / ROI
e) Hough transzformáció
f) Sávdetektálás (kimenet)

(University of Aucland)

Morfológiai Operátor (Zárás)



a) sávelválasztó jel
b) éldetektálás
c) dilatáció
d) erózió

(University of Aucland)

(iPhone alapú) Sávkövetés

képsorozat (bemenet) **Hough transzformáció után** **Sávdetektálás (kimenet)**

(University of Aucland)

OpenCV – egy nyitott forrású képfeldolgozó programcsomag

- http://opencvlibrary.sourceforge.net/
- http://groups.yahoo.com/group/OpenCV/
- http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/index.html
- http://nchc.dl.sourceforge.net/sourceforge/opencvlibrary/ippocv.pdf

Minden általam ismertetett képfeldolgozó algoritmushoz letölthető implementáció (és sok minden más is)

Lineáris Szűrők Elmélete

Ez egy klasszikus téma a képfeldolgozás elméletéből

Részben M. Pollefesys diáit felhasználva

Lineáris szűrők

- Algoritmus:
  - Új kép pixeleinek értékét a lokális környezet eredeti pixel értékeinek súlyozott összege (átlaga) adja. Minden pozícióban ugyanazokat a súlyokat alkalmazzuk.
- Tulajdonságok:
  - Az eredménykép az eredeti kép lineáris függvénye

- Példa 1: Simítás átlagolással
  - Képezzük a pixel lokális környezetének átlagát
- Példa 2: Gauss szűrés
  - Képezzük a pixel lokális környezetének súlyozott átlagát
- Példa 3: Kép deriválás
  - Képezzük a pixel lokális környezetének súlyozott átlagát

Konvolúció

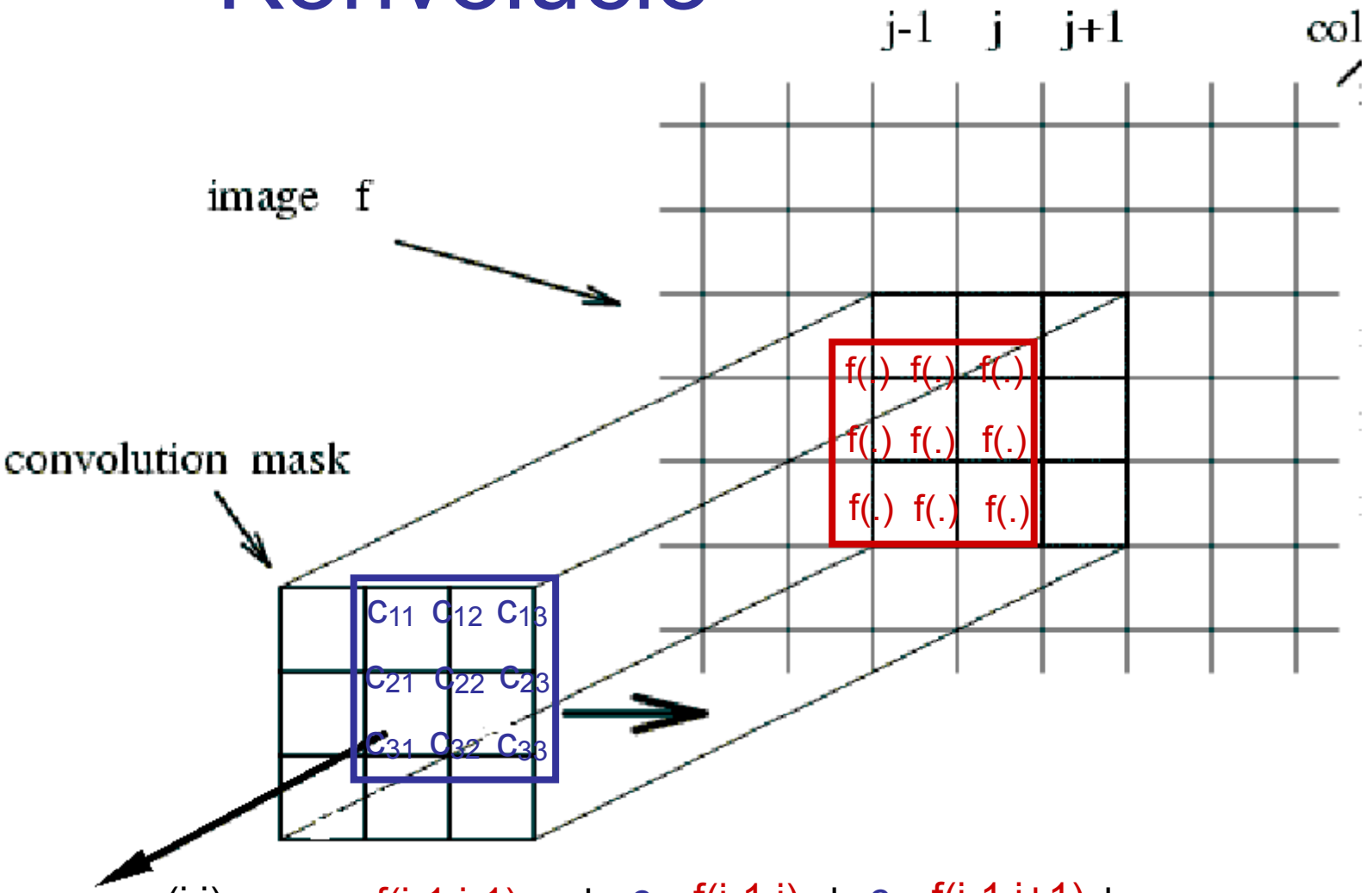
- Súlyok megadása képként: H
- H megnevezése általában: kernel
- Művelet neve: konvolúció (asszociativ)

- eredmény:

$$R_{ij} = \sum_{u,v} H_{i-u,j-v} F_{uv}$$

  - Mindegyik példánk ezzel a formulával kifejezhető
  - Eltolás független lineáris operátor

Konvolúció



$$o(i,j) = c_{11} f(i-1,j-1) + c_{12} f(i-1,j) + c_{13} f(i-1,j+1) + c_{21} f(i,j-1) + c_{22} f(i,j) + c_{23} f(i,j+1) + c_{31} f(i+1,j-1) + c_{32} f(i+1,j) + c_{33} f(i+1,j+1)$$

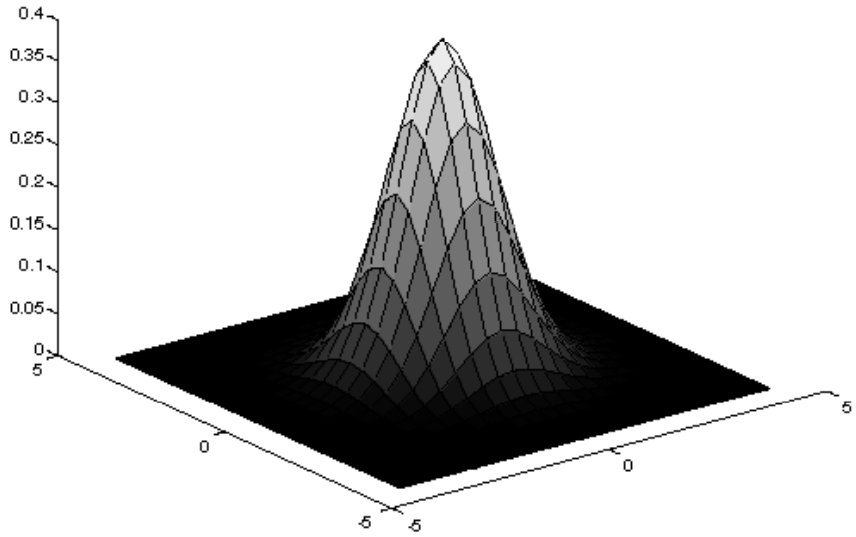
Példa 1: Simítás átlagolással

Figyeljük meg az eredményképen az átlagoláson túlmenően fellépő zajt (a kernel éles határa okozza)

Kernel

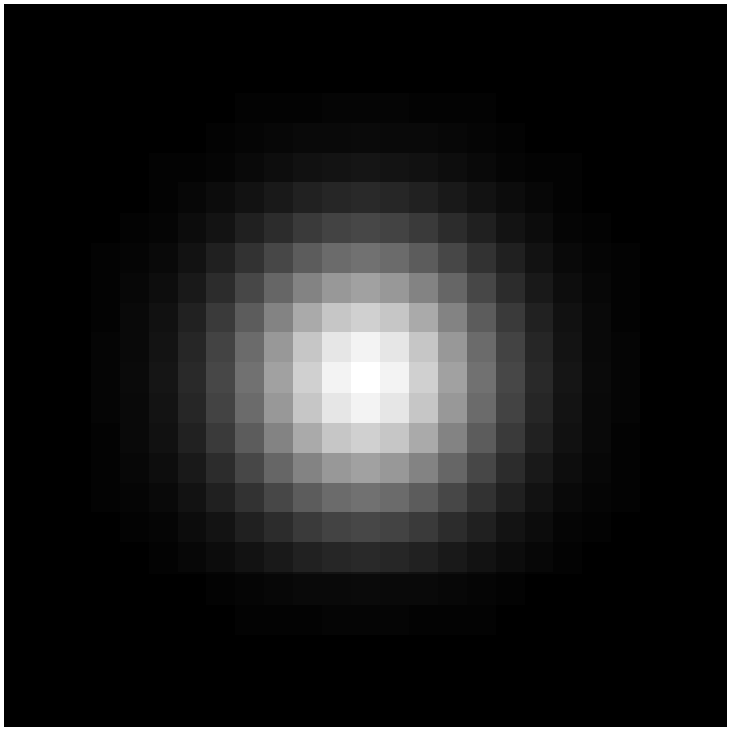
Példa 2: Simítás Gauss szűrővel

- Az előbbi átlagoló simítás nem jól modellez egy defókuszált lencsét
  - A leginkább szembetűnő különbség: egyetlen pont a defókuszált képen egy fuzzy foltként hatna, az átlagoló simítás viszont egy kis négyzetet eredményezett



- A Gauss szűrő viszont jól modellez egy defókuszált lencsét (fuzzy folt)

„Isotropic Gaussian”

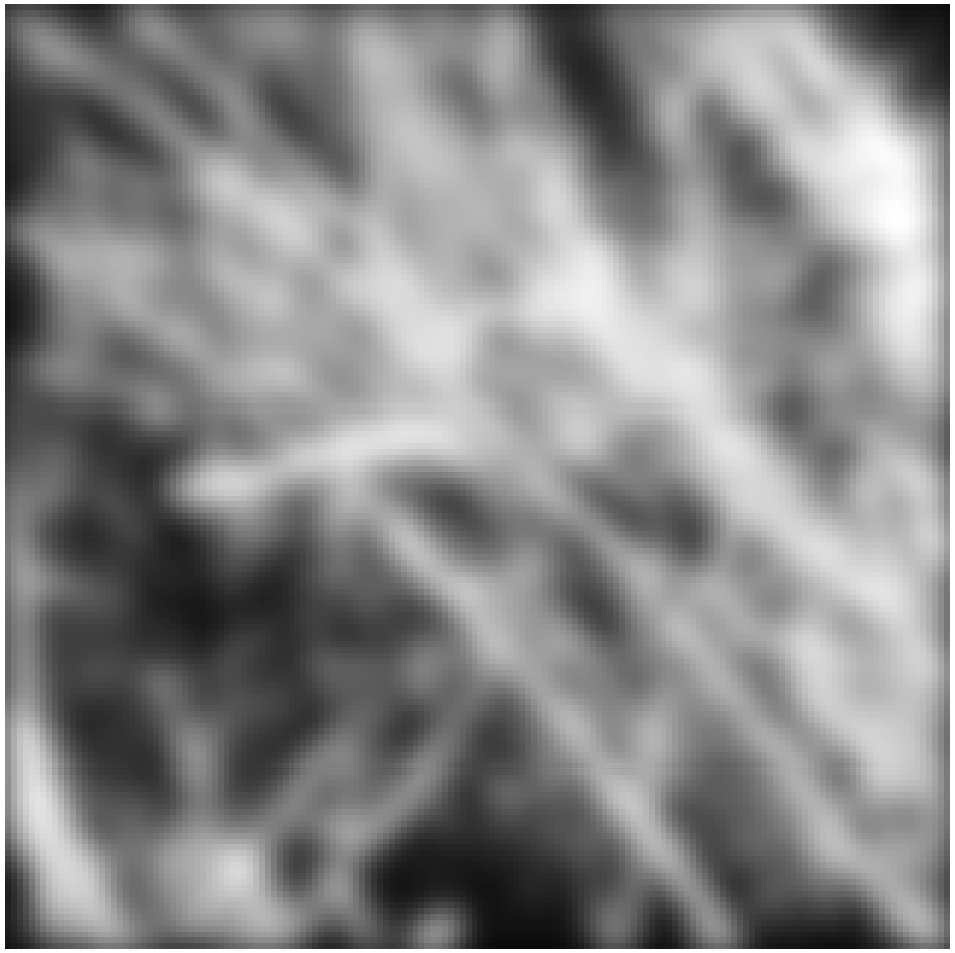


$$\exp\left(-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right)$$

(a körkörös „fuzzy folt” jó közelítése)

Gauss szűrés hatása

Kernel



Differenciálás és konvolúció kapcsolata

- Emlékeztetőül:

$$\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0} \left(\frac{f(x+\varepsilon, y)}{\varepsilon} - \frac{f(x,y)}{\varepsilon} \right)$$

- ez is lineáris és eltolás invariáns művelet (ld. konvolúció)

- A közelítése:

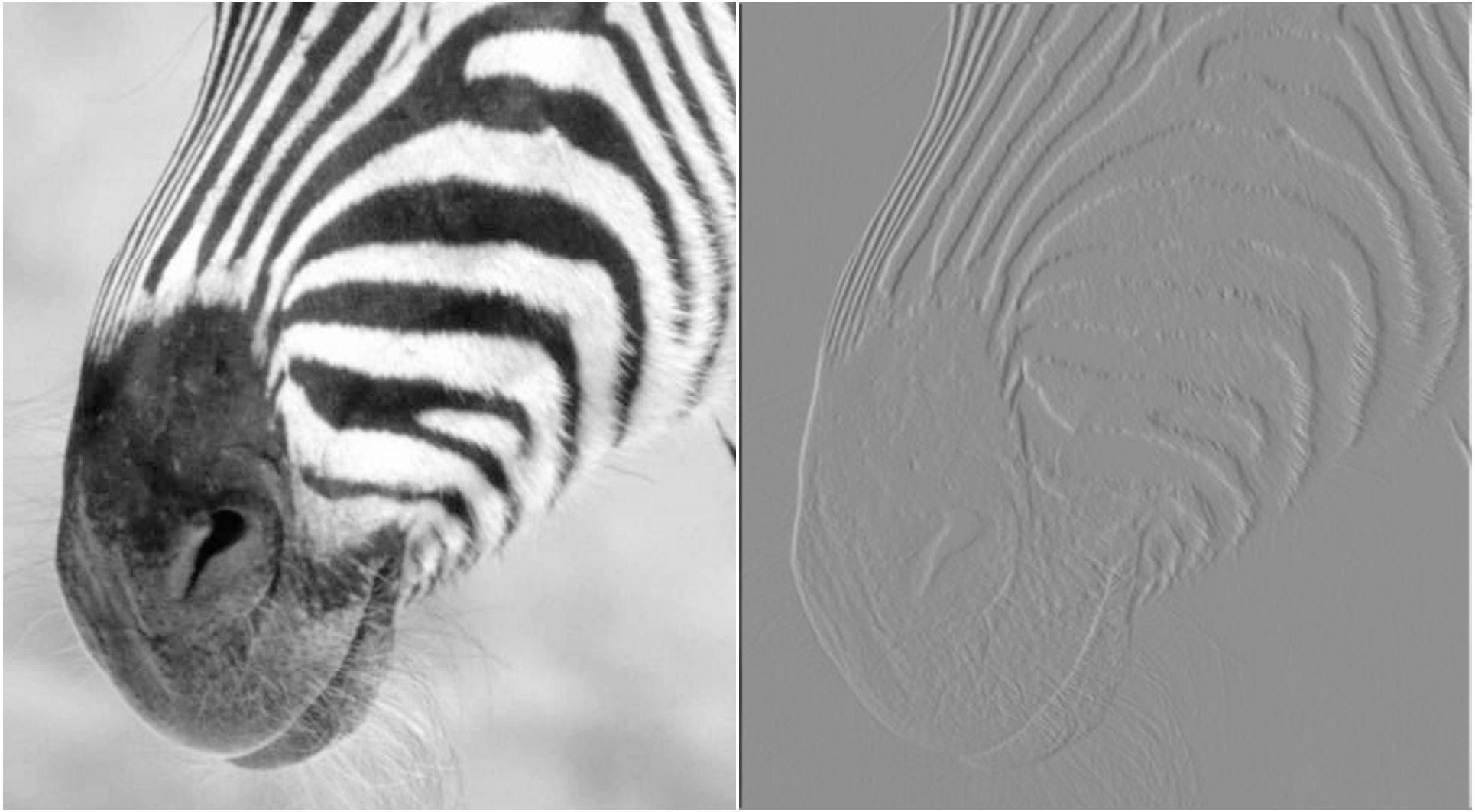
$$\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}$$

ez nyilvánvalóan konvolúció

A közelítésnek vannak minőségi hátulütői (ld. később)

Példa 3: Képdifferenciálás

(Sötét: negatív, világos: pozitív , szürke: zérus eredmény)



Zajok hatása

- Legegyszerűbb zaj modellt alkalmazva:
  - Állandó, független, additív Gauss zaj
  - A hozzáadott zaj értéke minden pixel pozícióban egy független érték egy normál valószínűségi eloszlásból választva
- Problémák:
  - A zajmodell megengedné a maximális kamera kimeneti értéknél nagyobb vagy zérusnál kisebb értékek felvételét is (gyakorlatban: telítés jelensége)
  - Kis standard szórás esetén jó közelítés a modell
  - a függetlenség nem mindig indokolt feltételezés (pl. karc/kosz a lencsén)
  - Az állandóság sem mindig indokolt (pl. a CCD érzékelő hőmérsékletfüggése)
  - A csak additív jellegű zaj sem mindig indokolt (pl. a CCD érzékelő helyfüggő multiplikatív érzékenység ingadozása)

sigma=1

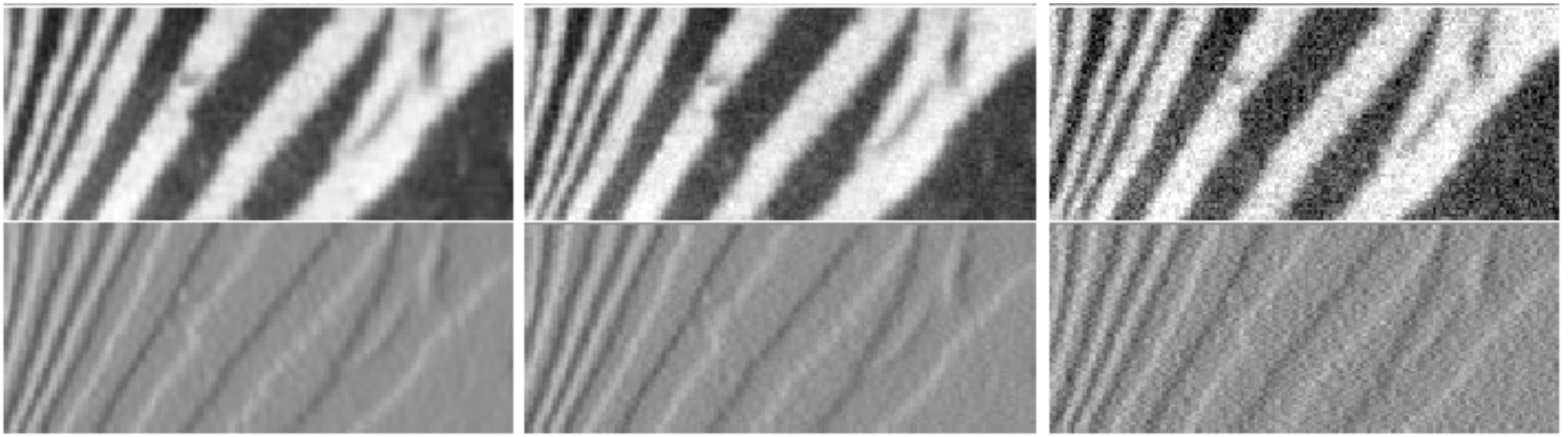
sigma=16

Képdifferenciálás zajos képeken

- Véges differenciális szűrők érzékenyen reagálnak a zajra
  - Nyilvánvaló ok: a zaj a szomszédjaitól nagyon „eltérő” pixel értékeket generálhat
- Nagyobb zaj amplitudó esetén – durvább a nem kívánt effektus

- Mi a teendő?
  - Intuíció szerint is: a legtöbb pixelnek „úgy kéne kinéznie” mint a szomszédos pixelek
  - Ez még az éleknél is részben igaz (legalábbis az élek hossza mentén)
  - Fentiek azt sugallják, hogy egy előzetes simítás segíthet: „kényszerítve” a pixeleket a környezetbe való jobb „belesímulásba”

Képdifferenciálás zajos képeken

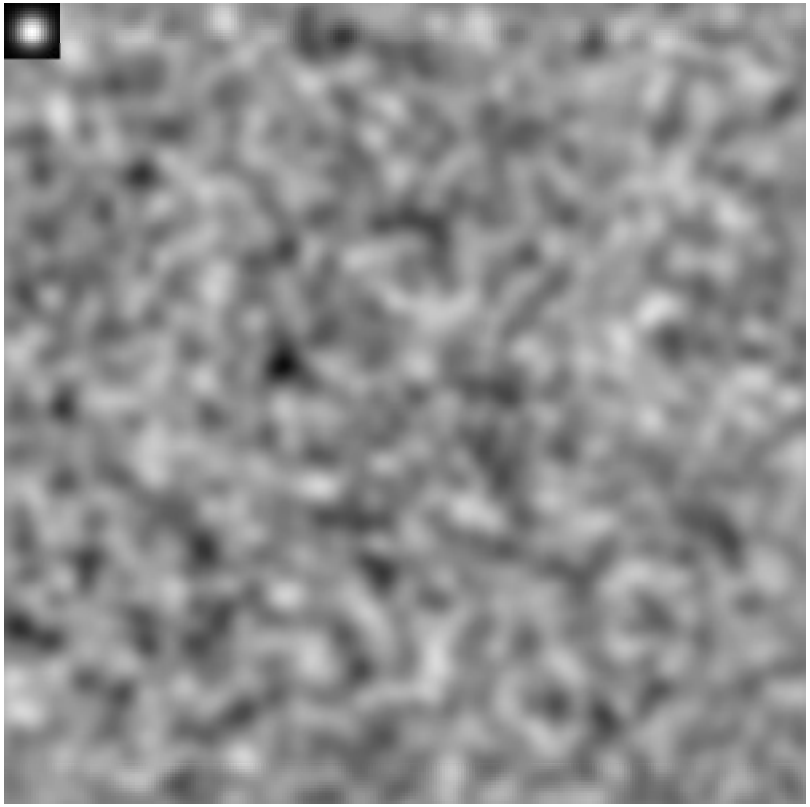


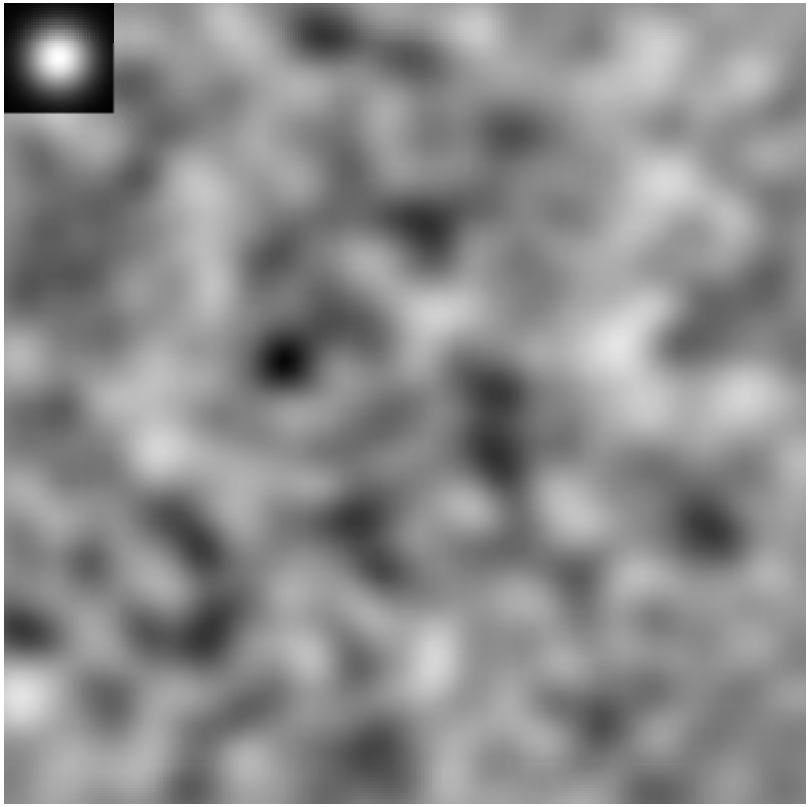
Nagyobb zaj ->
(zérus középértékű additív gauss zaj hatása)

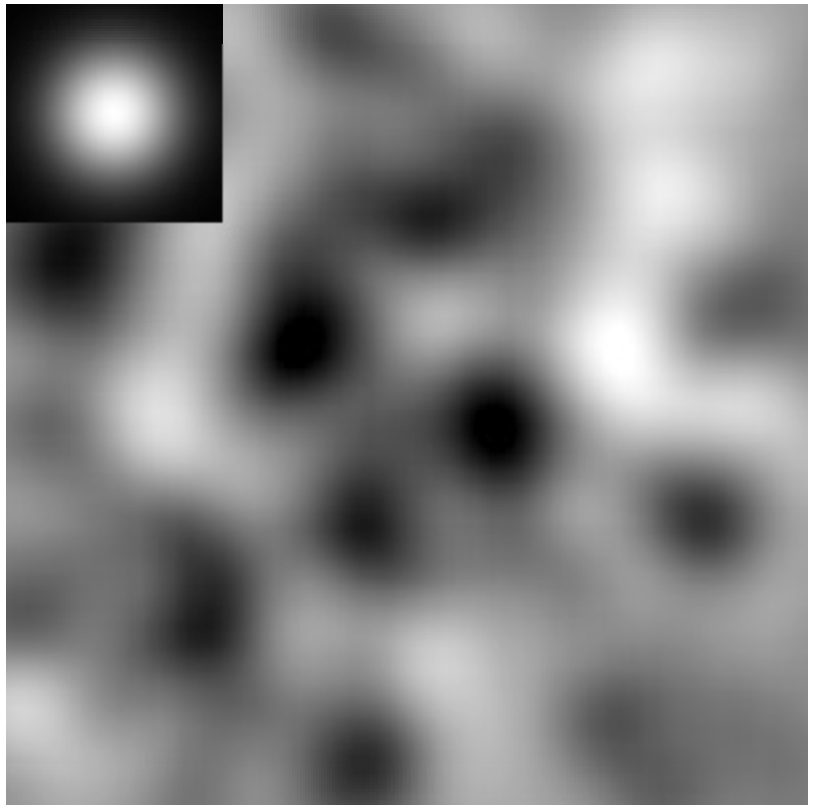
Szűrő válaszok korrelálhatnak

ha a szűrő kernel szélesebb a zajénál.

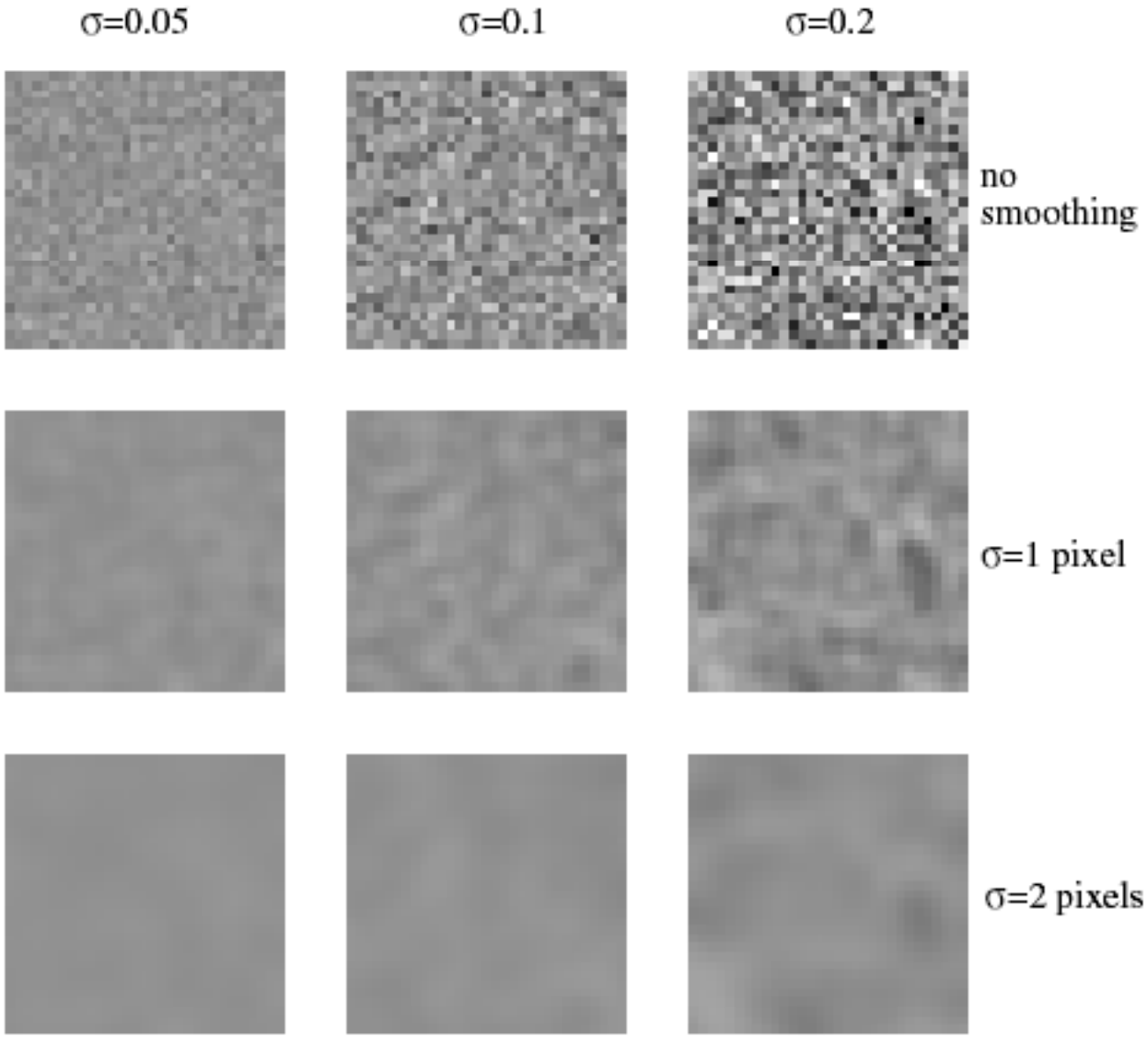
Érdekes texturált képet eredményezhet a szűrés (természetes texturák, stb. modellezésére jó)







A szűrő- és zajparaméterek kölcsönhatásai



A sorok a különböző szórású Gauss szűrők hatását mutatják
Az oszlopok a különböző szórású Gauss zajjal terhelt képeket mutatják

Medián szűrők

Medián szűrő

Nemlineáris szűrő

„Algoritmus” :

1. szomszédos pixelek sorbarendezése érték szerint
2. a középső érték vétele a rendezett sorból

Medián szűrő: „odd-man-out2

Előnyös is lehet az “odd-man-out” hatás

Például:

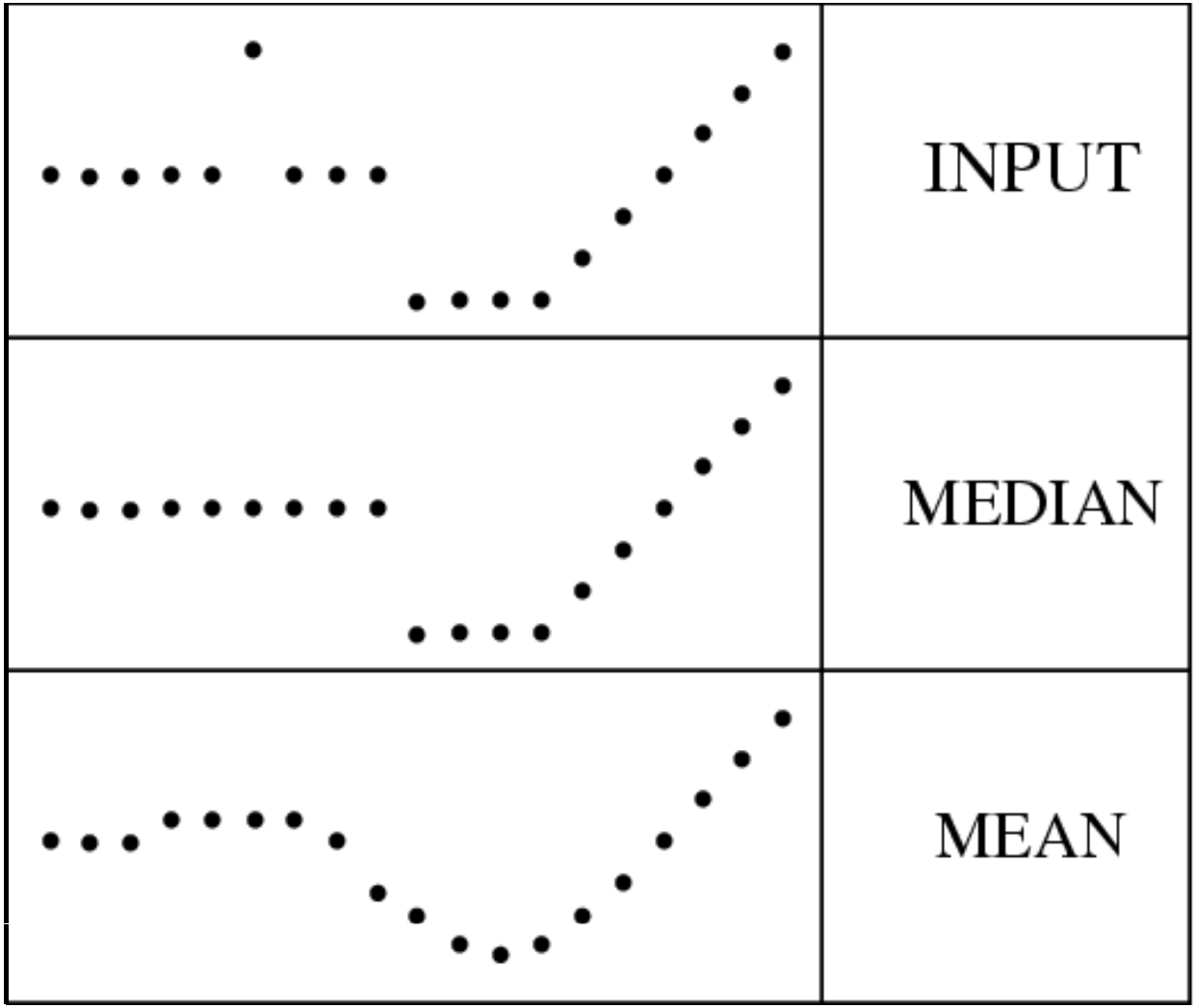
1,1,1,7,1,1,1,1

↓

?,1,1,1.1,1,1,?

Medián szűrő vs. Átlagoló szűrő (példa)

A Medián szűrő „szélessége” legyen = 5



Medián szűrő: pro és kontrák

Medián szűrő a csúcsokat hanyagolja
Lineáris szűrő viszont mindent figyelembevesz

Medián szűrő megőrzi a folytonossági hiányokat
Lineáris szűrő viszont „elkeni” azokat

Medián szűrés hatása

3 x 3 medián szűrő :

Lineáris szűrő hatása

10x Median szűrés hatása

10-szer egymás után egy 3 X 3 medián szűrés végrehajtva:

Éldetektálás

Ez is egy klasszikus probléma a képfeldolgozás elméletéből
(A Canny algoritmushoz az elméleti alapokat érdemes összefoglalni)

Részben S. Narasimhan diáit felhasználva

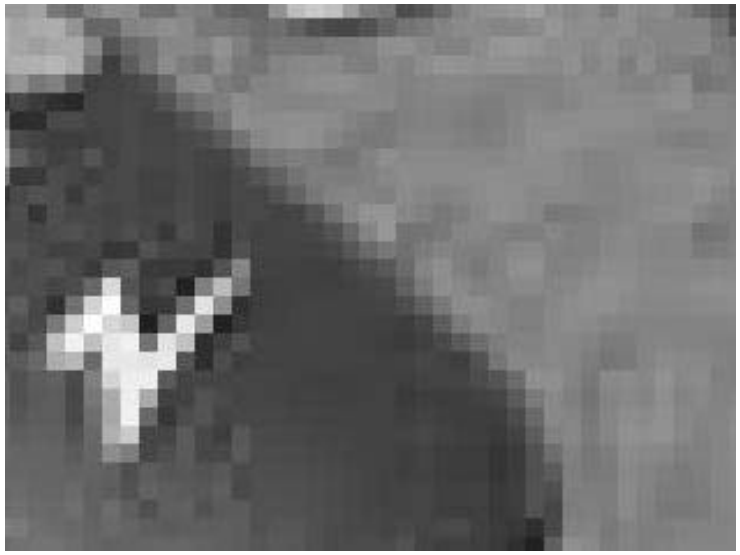
Él

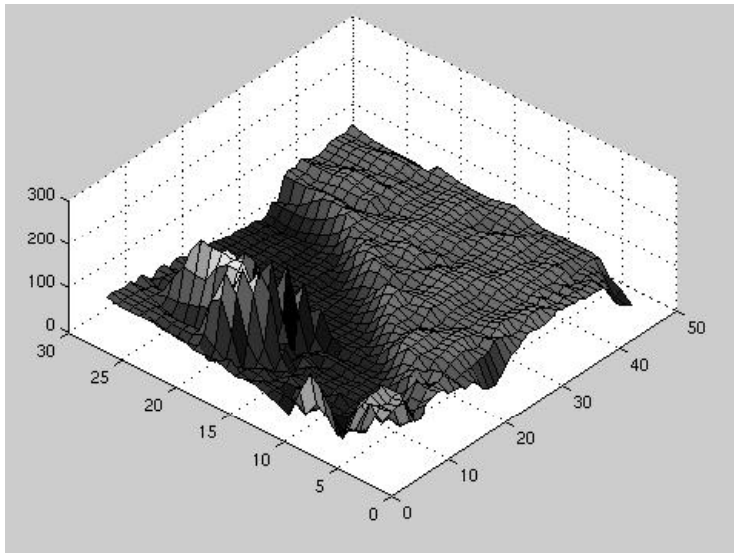
… a kép „fontos” leíró eszköze - de konkrétan miért is?

keressük meg a helyét a jellemzők hierarchiájában:

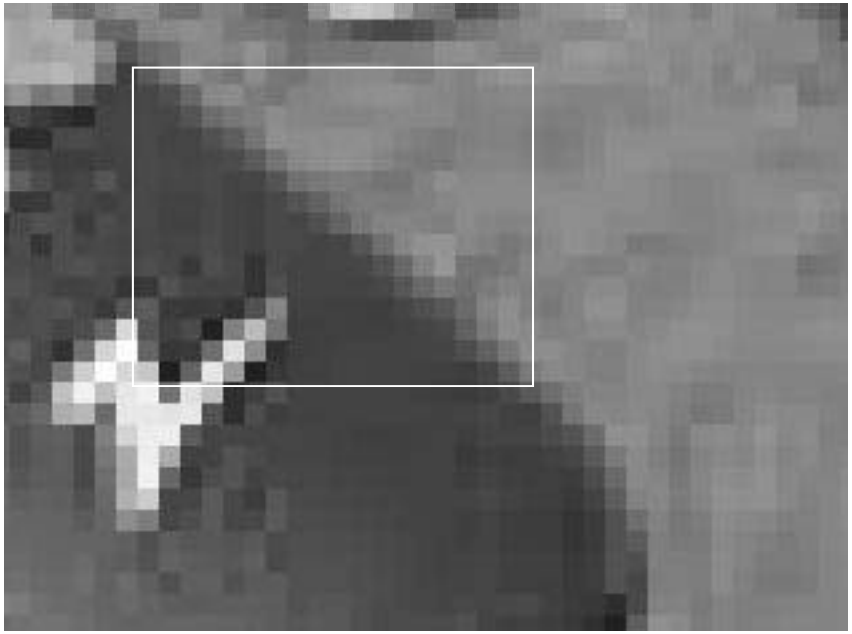
Élek tulajdonságai

Mit reprezentálnak?
Mi a megjelenési formájuk?

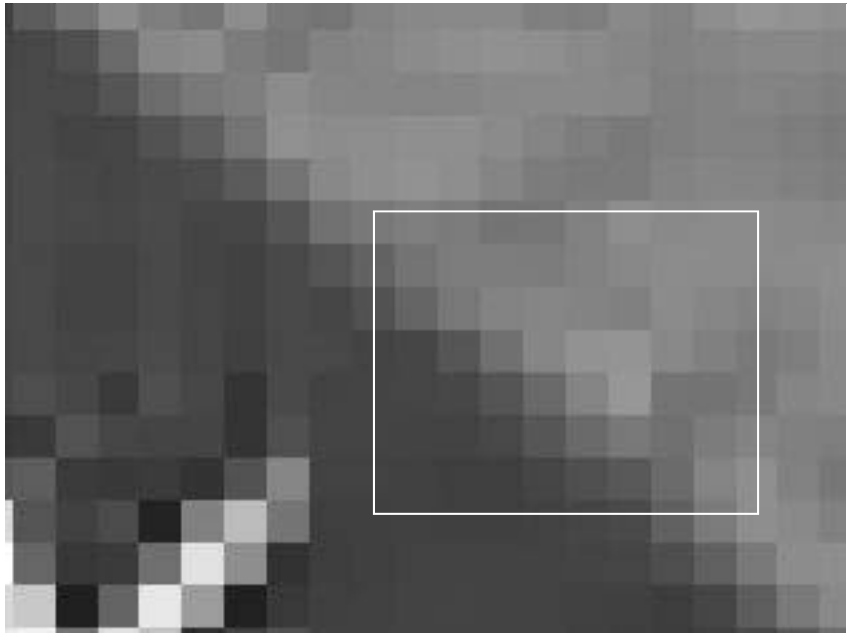


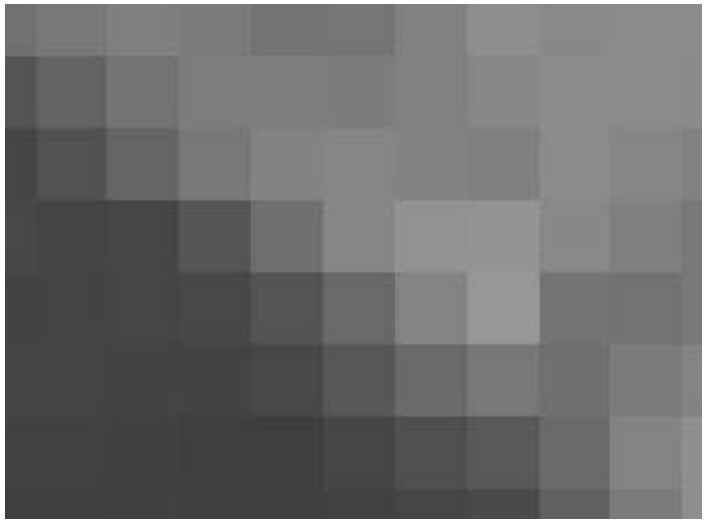


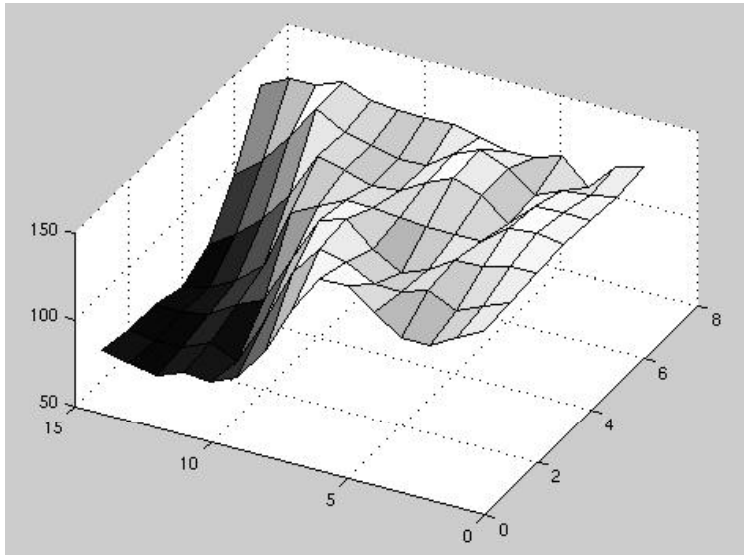
Élek tulajdonságai



Élek tulajdonságai



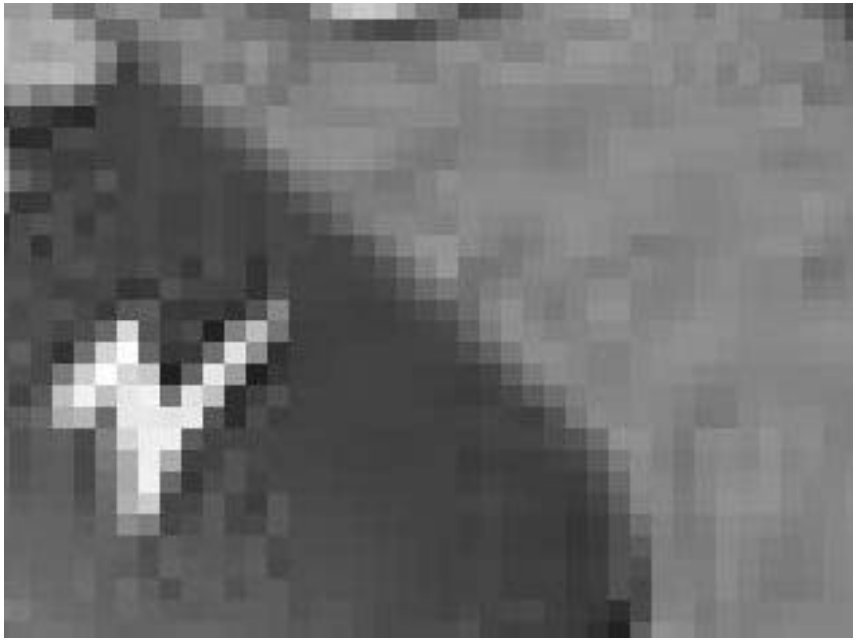




Élek → kép lokációi, ahol nagy a gradiens

Élek eredete

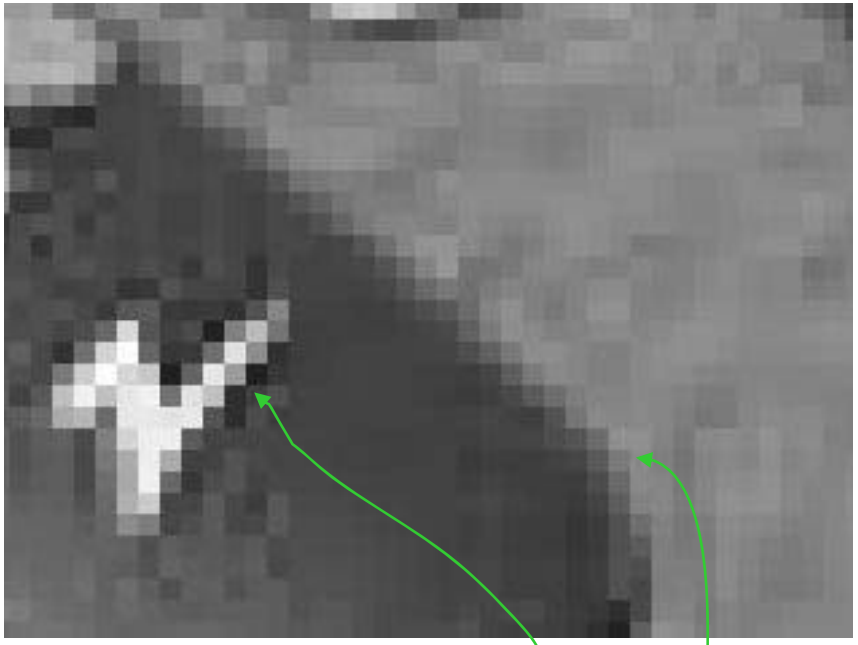
Mit reprezentálnak?



négy fizikai jelenség is eredményezhet ugyanolyan élet a képen...

Élek eredete - jórészt zajból!

Mit reprezentálnak?

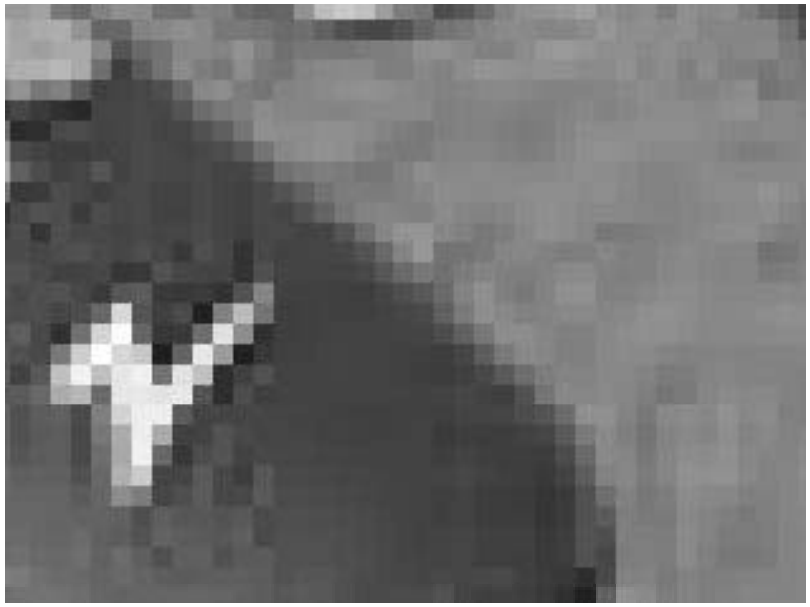


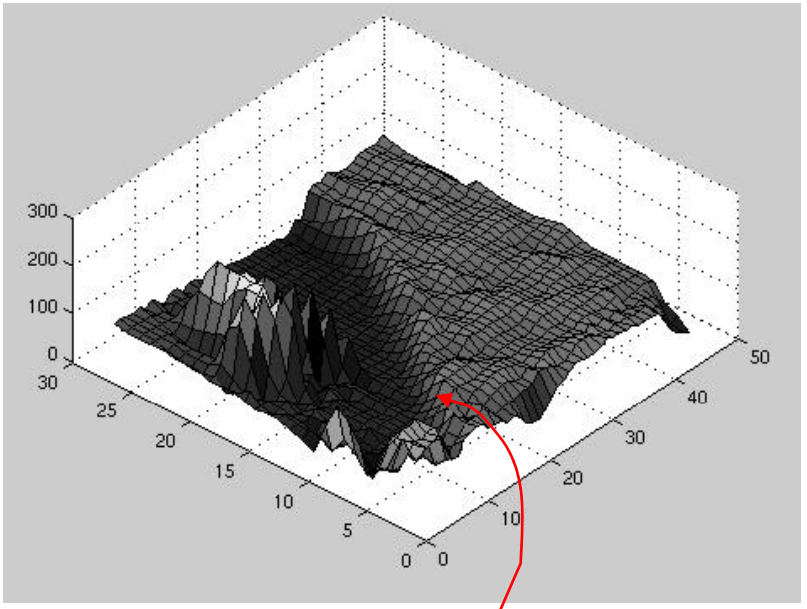
négy – teljesen eltérő tartalmú - fizikai jelenség eredményez élet a képen...

folytonosság hiány

- felületi szín / intenzitás
- felületi normális
- mélység érték
- megvilágítás (becsillogások)

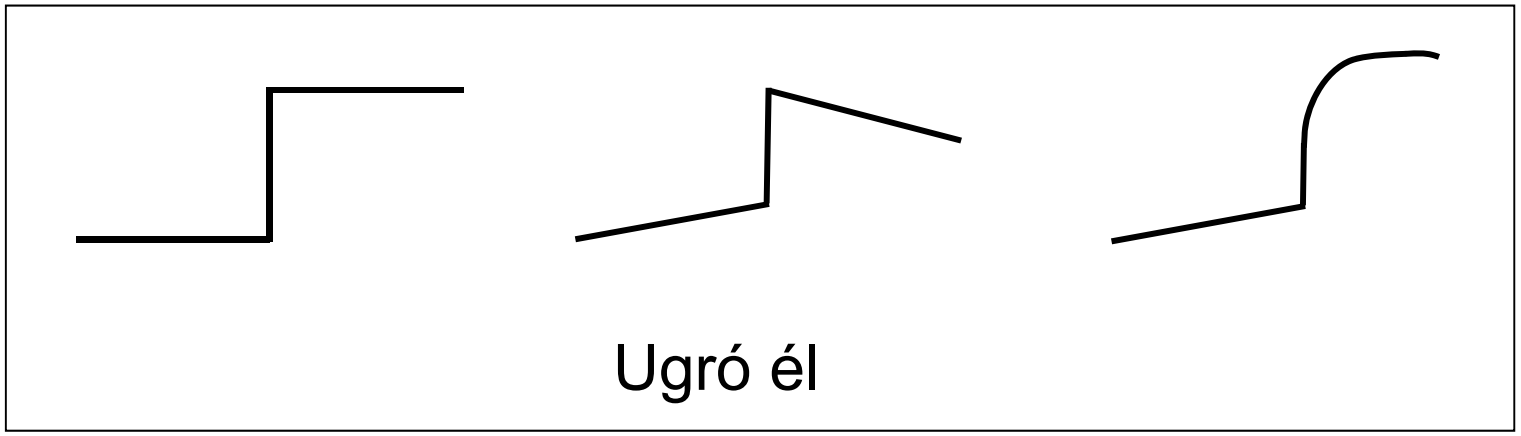
Élek jelentése

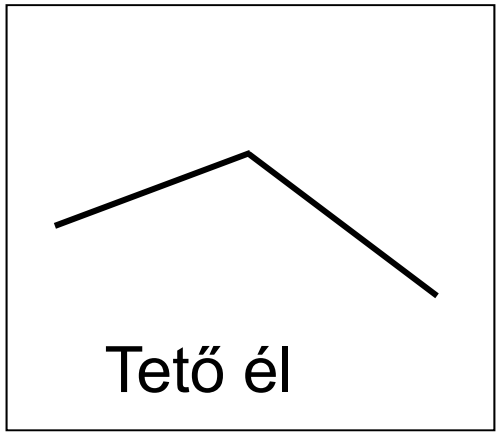


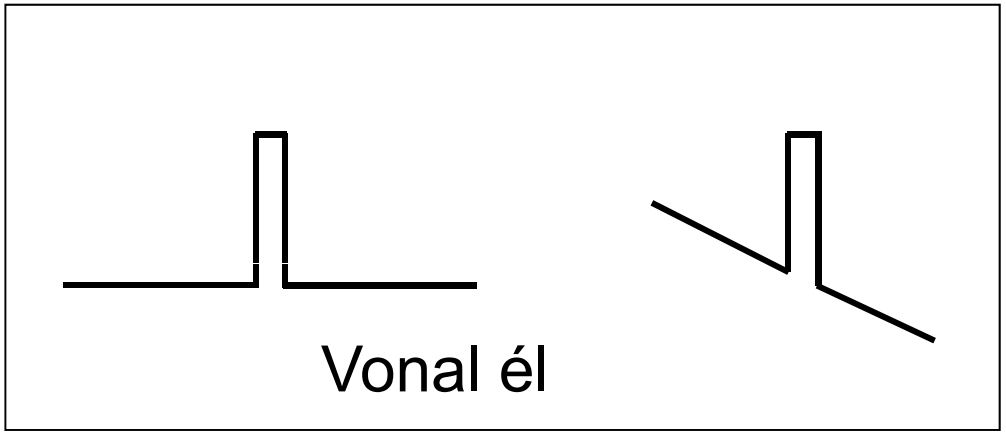


Lokális maximum a gradiensben, megadott irányban

Éltípusok

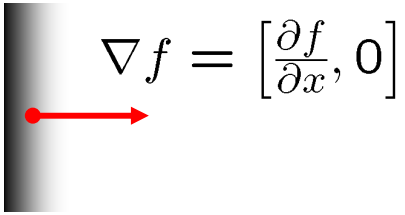


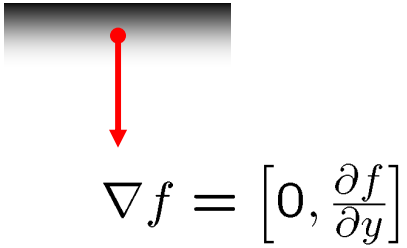


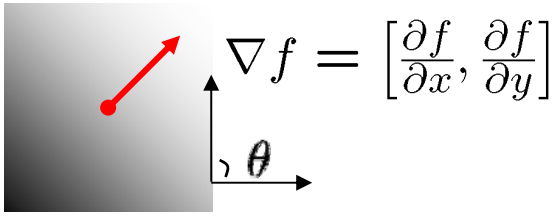


Gradiens

- Gradiens számítása: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$
- A legnagyobb intenzitás változás irányát adja meg







- Gradiens iránya: $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$
- A gradiens nagysága („ereje”)

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

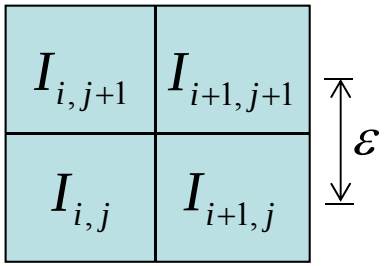
Diszkrét Él operátorok

- Hogyan differenciálunk egy ***diszkrét*** képet?

Közelítőleg:

$$\frac{\partial I}{\partial x} \approx \frac{1}{2\varepsilon}\left(\left(I_{i+1,j+1} - I_{i,j+1}\right) + \left(I_{i+1,j} - I_{i,j}\right)\right)$$

$$\frac{\partial I}{\partial y} \approx \frac{1}{2\varepsilon}\left(\left(I_{i+1,j+1} - I_{i+1,j}\right) + \left(I_{i,j+1} - I_{i,j}\right)\right)$$



Konvolúciós maszkkal (súlyok) megadva:

$$\frac{\partial I}{\partial x} \approx \frac{1}{2\varepsilon}\begin{array}{|c|c|}\hline -1 & 1 \\ \hline -1 & 1 \\ \hline \end{array}$$

$$\frac{\partial I}{\partial y} \approx \frac{1}{2\varepsilon}\begin{array}{|c|c|}\hline 1 & 1 \\ \hline -1 & -1 \\ \hline \end{array}$$

Diszkrét Él operátorok

- másodrendű parciális deriváltak közelítése:

$$\frac{\partial^2 I}{\partial x^2} \approx \frac{1}{\varepsilon^2}\left(I_{i-1,j} - 2I_{i,j} + I_{i+1,j}\right)$$

$$\frac{\partial^2 I}{\partial y^2} \approx \frac{1}{\varepsilon^2}\left(I_{i,j-1} - 2I_{i,j} + I_{i,j+1}\right)$$

$I_{i-1,j+1}$	$I_{i,j+1}$	$I_{i+1,j+1}$
$I_{i-1,j}$	$I_{i,j}$	$I_{i+1,j}$
$I_{i-1,j-1}$	$I_{i,j-1}$	$I_{i+1,j-1}$

- **Laplace operátor** :

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Konvolúciós maszkkal (súlyok) megadva:

$$\nabla^2 I \approx \frac{1}{\varepsilon^2}$$

0	1	0
1	−4	1
0	1	0

$$\frac{1}{6\varepsilon^2}$$

1	4	1
4	−20	4
1	4	1

(utóbbi pontosabb)

Diszkrét Él operátorok

Más gradiens közelítések is használatosak

Sobel operátor:

$\frac{1}{8}$

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

s_x

$\frac{1}{8}$

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

s_y

Diszkrét El operátorok összehasonlítása

Gradiens: $$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Jó lokalizáció
Zajérzékeny
Gyenge detektálás

Roberts (2 x 2):

0	1
-1	0

1	0
0	-1

Sobel (3 x 3):

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	1

Sobel (5 x 5):

-1	-2	0	2	1
-2	-3	0	3	2
-3	-5	0	5	3
-2	-3	0	3	2
-1	-2	0	2	1

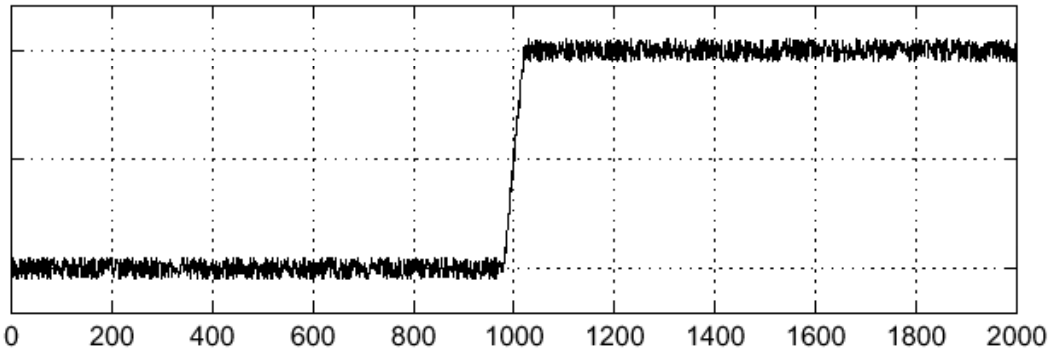
1	2	3	2	1
2	3	5	3	2
0	0	0	0	0
-2	-3	-5	-3	-2
-1	-2	-3	-2	-1

Pontatlan lokalizáció
Kevésbé zajérzékeny
Jó detektálás

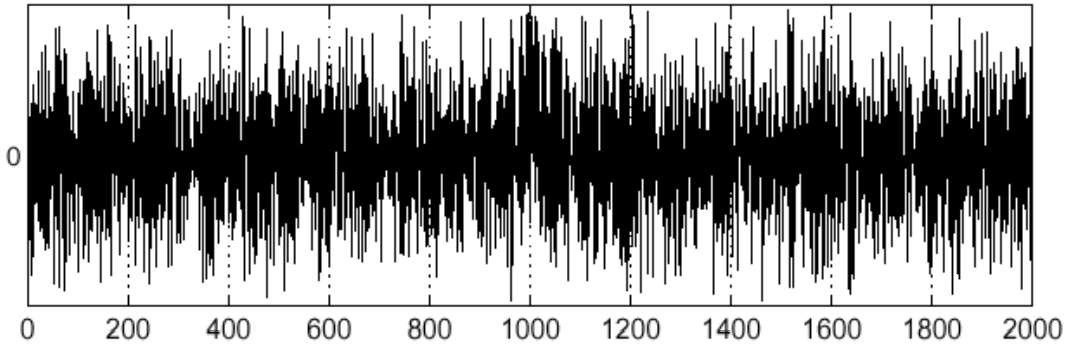
Zajok hatása

A kép egy sorát tekintve (intenzitás az x pozíció függvényében)

$f(x)$

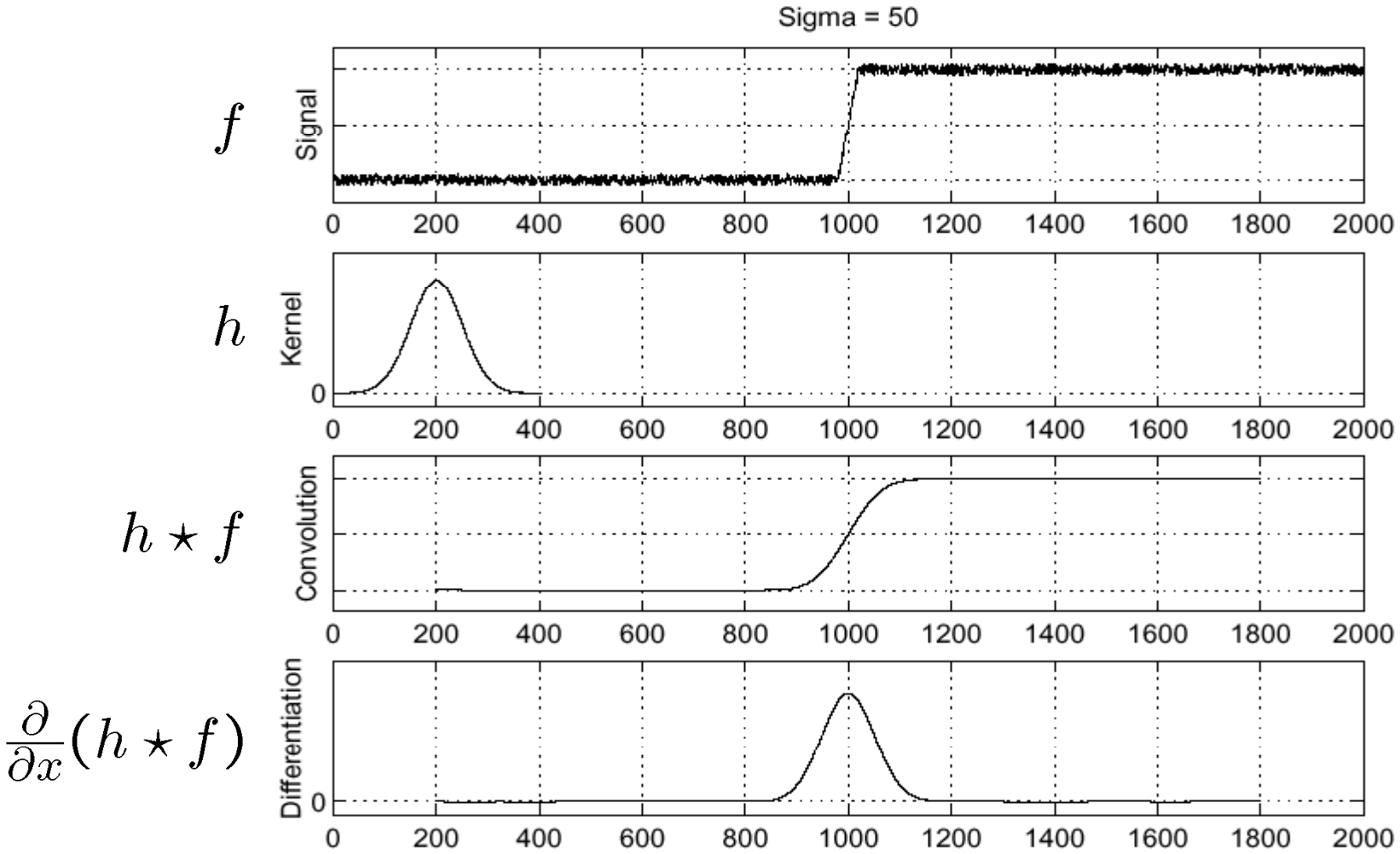


$\frac{d}{dx}f(x)$



Hol az él??

Megoldás: először simítani kell



f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$

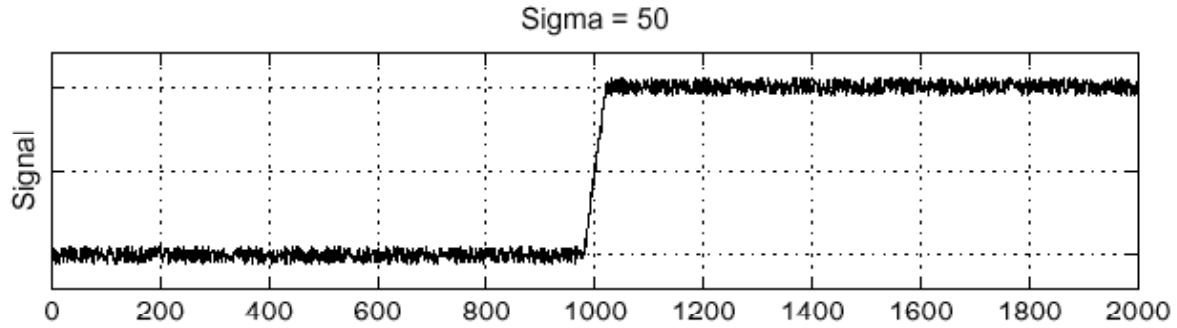
Hol az él? a maximumát keressük: $\frac{\partial}{\partial x}(h \star f)$

A konvolúció elméletéből következően

$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

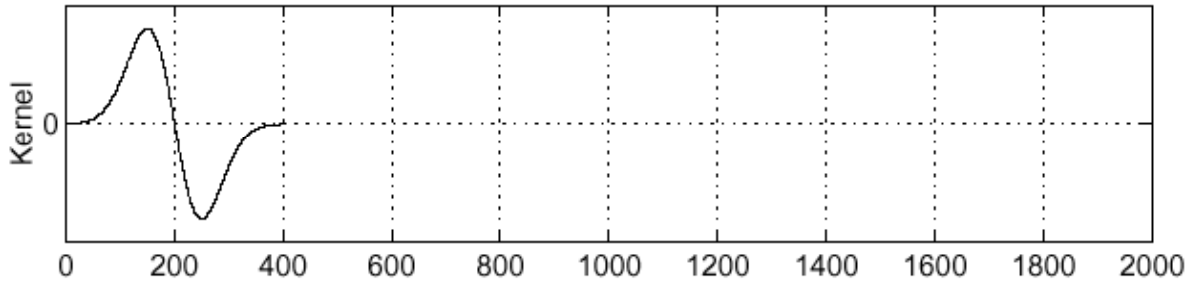
…egy lépés megspórolható

f

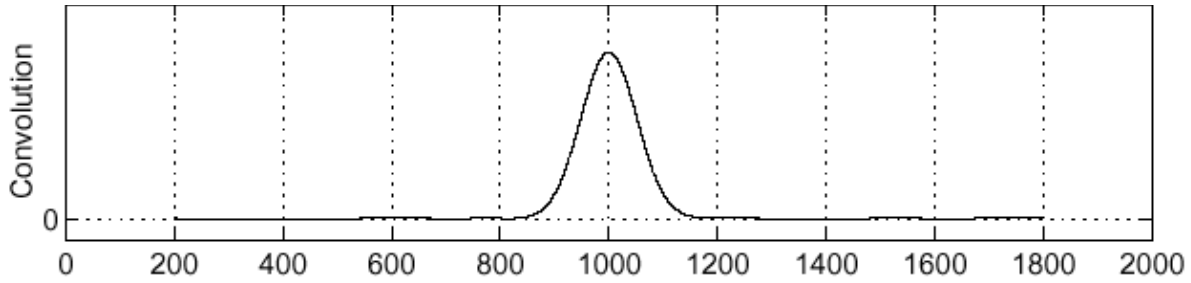


Differenciális operátor

$\frac{\partial}{\partial x}h$



$(\frac{\partial}{\partial x}h) \star f$



„Laplacian of Gaussian” (LoG)

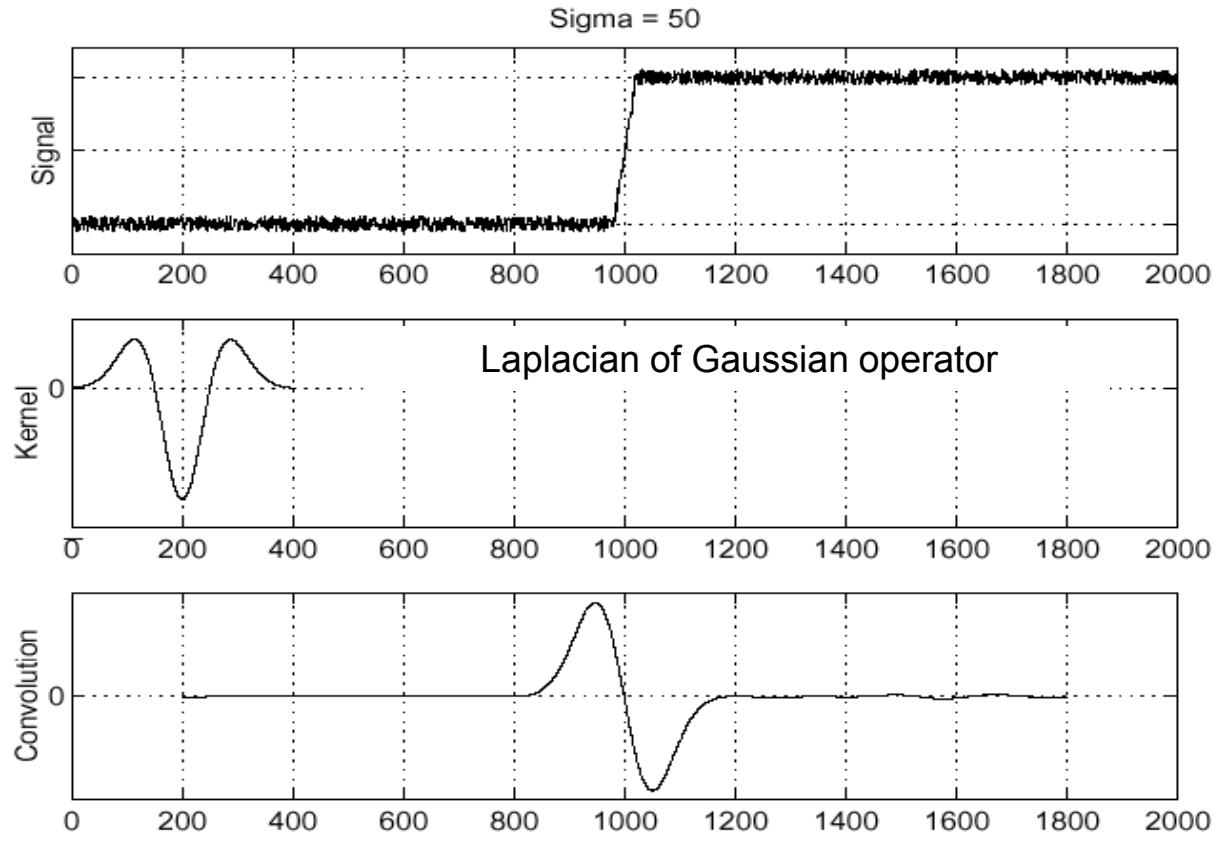
$$\frac{\partial^2}{\partial x^2}(h * f) = \left(\frac{\partial^2}{\partial x^2} h\right) * f$$

Laplacian of Gaussian

f

$\frac{\partial^2}{\partial x^2} h$

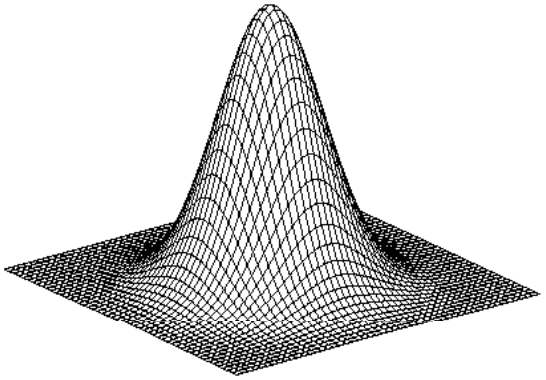
$(\frac{\partial^2}{\partial x^2} h) \star f$



Hol az él?

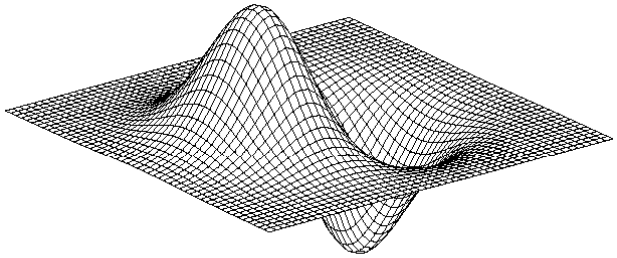
A nulla - átmenetnél!

2D Gauss Él Operátorok



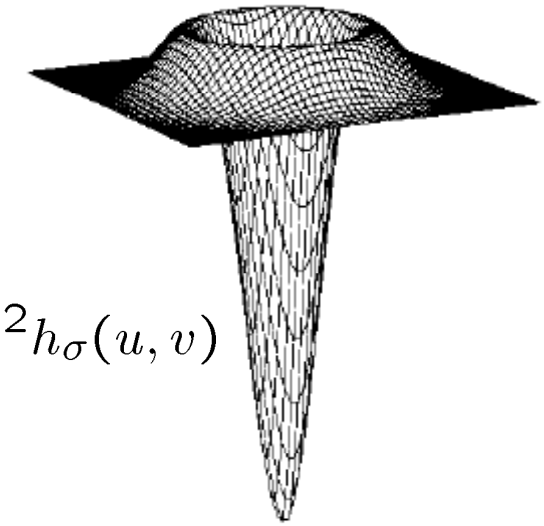
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

Gaussian



$$\frac{\partial}{\partial x} h_\sigma(u,v) \qquad \nabla$$

Derivative of Gaussian (DoG)



$$\nabla^2 h_\sigma(u,v)$$

∇^2 Laplacian of Gaussian

Mexican Hat (Sombrero)

- ∇^2 a **Laplace** operátor:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Az elméleti előkészítés után

Canny ÉI Operátor

Canny Él Operátor

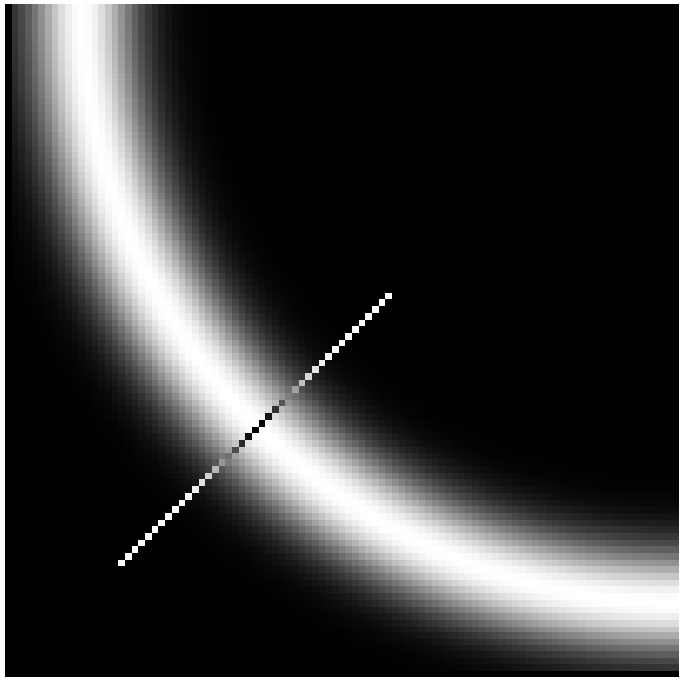
- Képszűrés 2D Gauss szűrővel: $G * I$
- Minden pixelre a gradienst irányt (az él normálisát) meghatározni

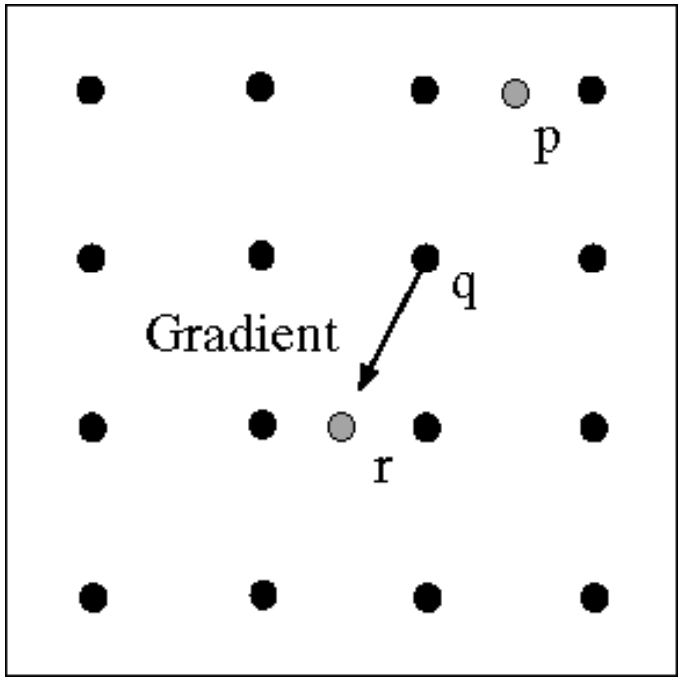
$$\bar{\mathbf{n}} = \frac{\nabla(G * I)}{|\nabla(G * I)|}$$

- Gradiens nagyságát meghatározni $|\nabla(G * I)|$
- A normális irányok mentén a nulla-átmenetet megkeresni (**nem-maximális értékű pontok elhagyása)**

$$\frac{\partial^2(G * I)}{\partial \bar{\mathbf{n}}^2} = 0$$

Nem-maximális értékű pontok elhagyása





- A gradiens iránya mentén a lokális maximum megkeresése
  - p és r interpolált pontok ellenőrzése kell

Canny éldetektor hatása

Eredeti kép (Lena)

Canny éldetektor hatása

gradiens nagyság

Canny éldetektor hatása

Nem-maximális értékű pontok elhagyása után

Canny éldetektor

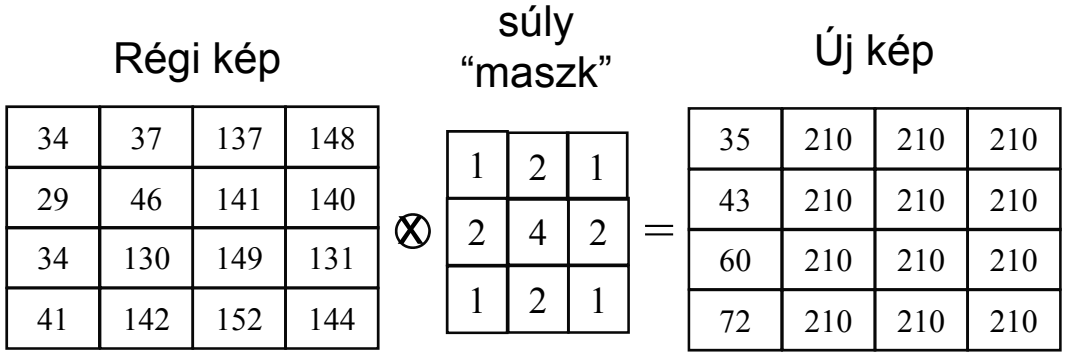
Cél : megtalálni azokat a képpontokat, ahol a gradiensnek lokális maximuma van

(1) Zajszűrés képsimítással (súlyozott átlag képzés „konvolúcióval”)

A pixelértékeket a szomszédos pixelek súlyozott összegével helyettesítjük.

$$I_{\text{új}}(x,y) = \sum_{i,j=-1}^{1} w_{ij}\, I_{\text{régi}}(x+i,y+j)$$

súly



Canny éldetektor

Cél : megtalálni azokat a képpontokat, ahol a gradiensnek lokális maximuma van

(1) Zajszűrés képsimítással (aluláteresztő / átlagoló szűrő)

A pixelértékeket a szomszédos pixelek súlyozott összegével helyettesítjük.

$$I_{\text{új}}(x,y) = \sum_{i,j=-1}^{1} w_{ij}\, I_{\text{régi}}(x+i,y+j)$$

súly

Eredeti kép

34	37	137	148
29	46	141	140
34	130	149	131
41	142	152	144

⊗

súly “maszk”

1	2	1
2	4	2
1	2	1

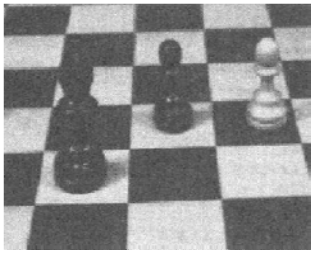
=

Új kép

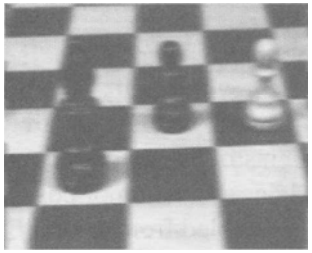
35	62	116	143
43	76	122	141
60	102	136	140
72	112	145	143

(normálva)

Eredeti kép

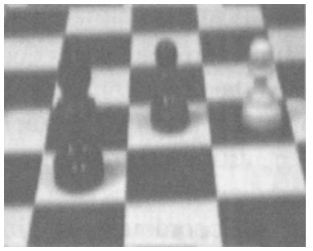


gyorsabb irányonként külön-külön, egymás után simítani...



Függőleges irányban →

← Vízszintes irányban



Canny éldetektor

Cél : megtalálni azokat a képpontokat, ahol a gradiensnek lokális maximuma van

(1) Zajszűrés képsimítással

A pixelértékeket a szomszédos pixelek súlyozott összegével helyettesítjük

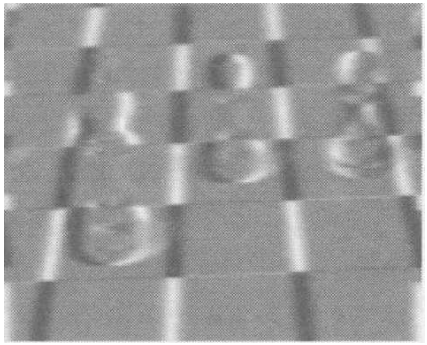
$$I_{\text{új}}(x,y) = \sum_{i,j=-1}^{1} w_{ij}\, I_{\text{régi}}(x+i,y+j)$$

súly

(2) Gradiens becslése minden pontban

Most egy másik súlymaszkot alkalmazunk – vele a deriváltakat becsüljük

Ugyanaz a művelet (konvolúció), csak más súlyokkal



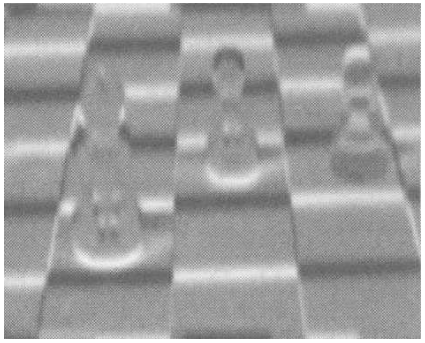
$$\frac{dI}{dx} = I_x$$

Függőleges irányban

1
-1

-1	1

Vízszintes irányban



$$\frac{dI}{dy} = I_y$$

Canny éldetektor

Cél : megtalálni azokat a képpontokat, ahol a gradiensnek lokális maximuma van

(1) Zajszűrés képsimítással

A pixelértékeket a szomszédos pixelek súlyozott összegével helyettesítjük

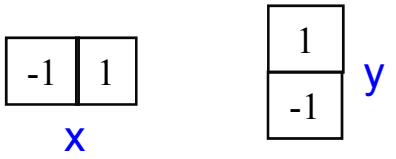
$$I(x,y)_{\text{új}} = \sum_{i,j=-1}^{1} w_{ij}\, I(x+i,y+j)_{\text{régi}}$$

súly

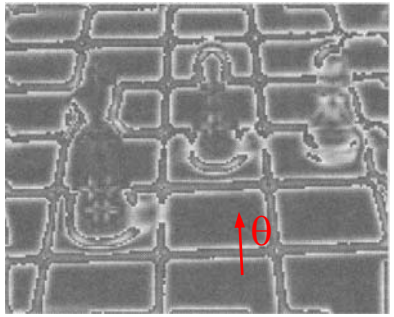
(2) Gradiens becslése minden pontban

Most egy másik súlymaszkot alkalmazunk – vele a deriváltakat becsüljük

Ugyanaz a művelet (konvolúció) csak más súlyokkal



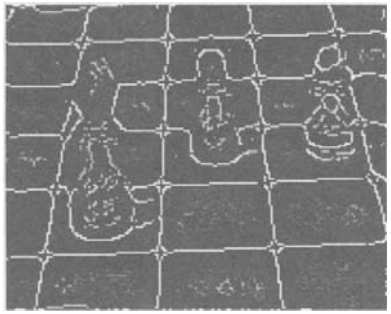
(3) Gradiens abszolút értékének és irányának meghatározása



$$|\nabla I(x,y)| = \text{sqrt}(I_x^2 + I_y^2)$$

$$\theta = \text{atan}(I_x, I_y)$$

Gradiens maximum keresése θ irányban



Canny éldetektor

Cél : megtalálni azokat a képpontokat, ahol a gradiensnek lokális maximuma van

(1) Zajszűrés képsimítással

A pixelértékeket a szomszédos pixelek súlyozott összegével helyettesítjük

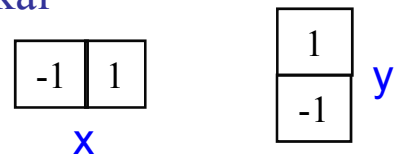
$$I(x,y)_{\text{új}} = \sum_{i,j=-1}^{1} w_{ij}\, I(x+i,y+j)_{\text{régi}}$$

súly

(2) Gradiens becslése minden pontban

Most egy másik súlymaszkot alkalmazunk – vele a deriváltakat becsüljük

Ugyanaz a művelet (konvolúció) csak más súlyokkal



(3) Gradiens abszolút értékének és irányának meghatározása

A gradiens irányában lokális maximumot keresünk

$$|\nabla I(x,y)| = \text{sqrt}(I_x^2 + I_y^2)$$

$$\theta = \text{atan}(I_x, I_y)$$

(4) választunk egy „optimális” küszöböt – a felette lévő gradiens érték esetén a pontot valóban élpontnak tekintjük!

A küszöb megválasztása

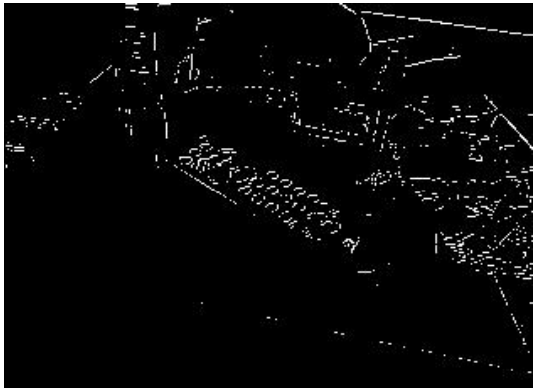
Eredeti kép

Nagyon magas
küszöbérték

A küszöb megválasztása

Eredeti kép

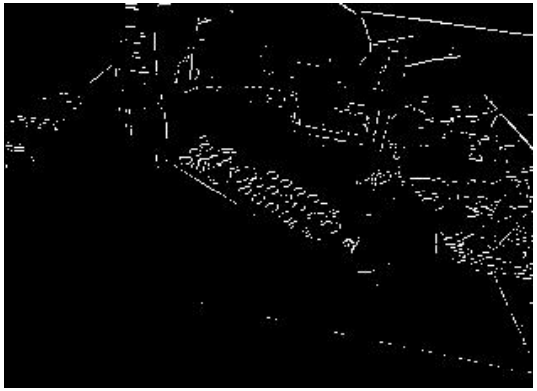
Nagyon magas küszöbérték

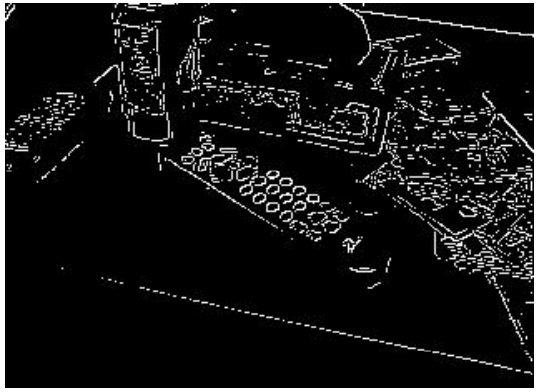


A küszöb megválasztása

Eredeti kép

Nagyon magas küszöbérték

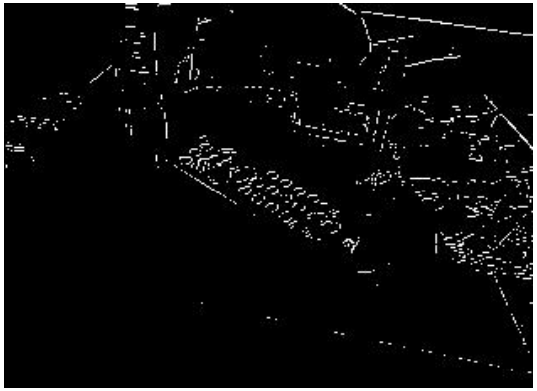


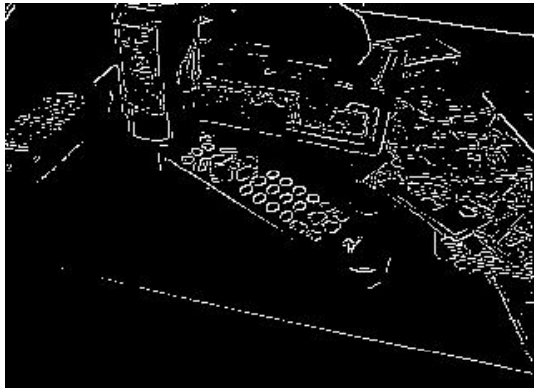


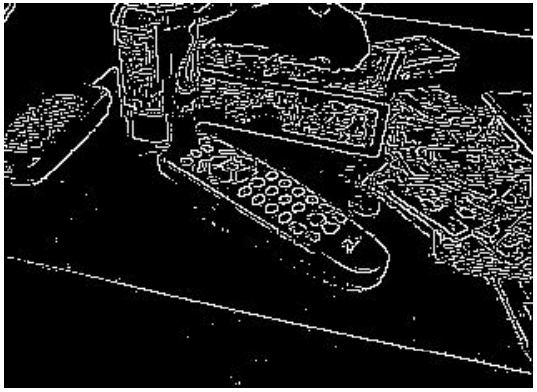
A küszöb megválasztása

Eredeti kép

Nagyon magas küszöbérték





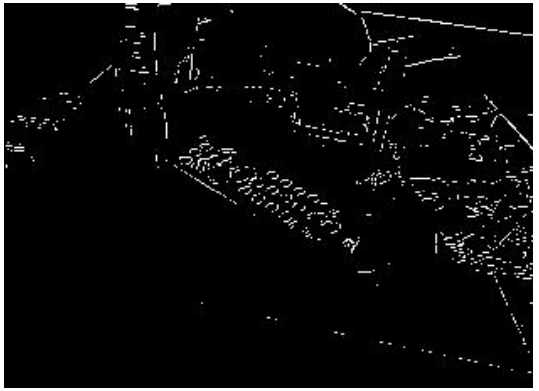


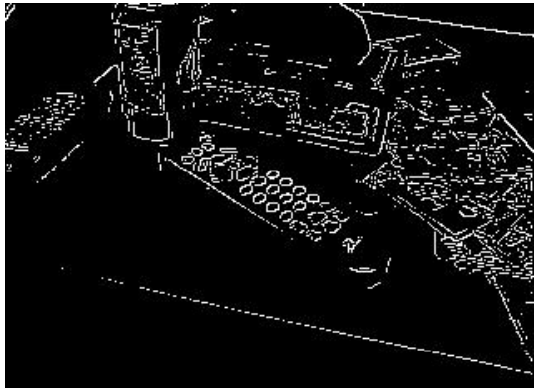
Racionálisnak tűnő

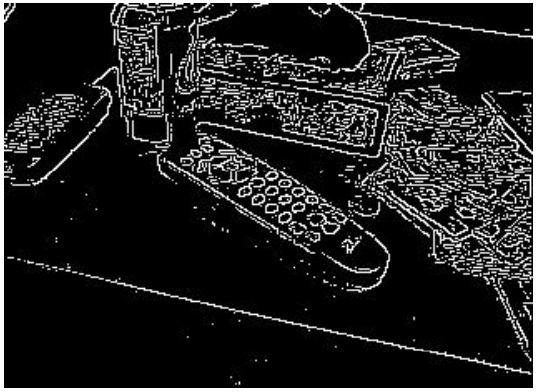
A küszöb megválasztása

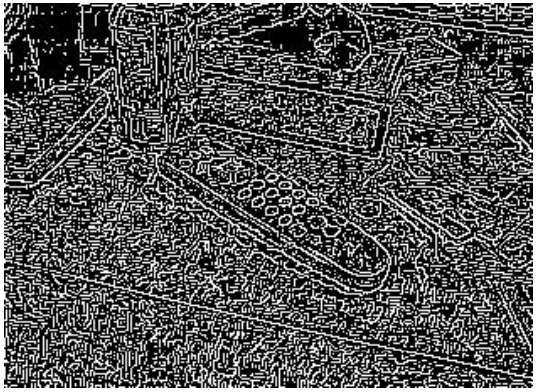
Eredeti kép

Nagyon magas küszöbérték →









→ Racionálisnak tűnő → **Túl alacsony !**

algoritmikus szinten megtalálni az optimális küszöböt – nem triviális feladat

Élpontok szegmentálása

Hough Transzformáció

Mit csináljunk az előbb megtalált élpontokkal?

A Hough transzformáció is egy régi történet…

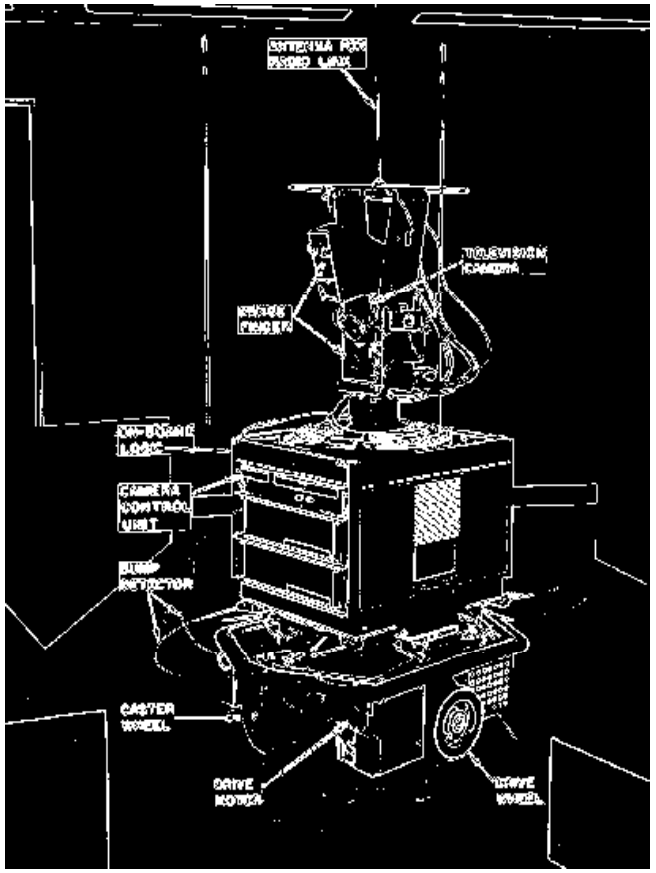
Részben Jeremy Wyatt diáit felhasználva

Élszegmensek keresése

A Canny éldetektor csak élpontokat, de nem élszegmenseket talált

Hogyan találhatunk meg összetettebb képjellemzőket?

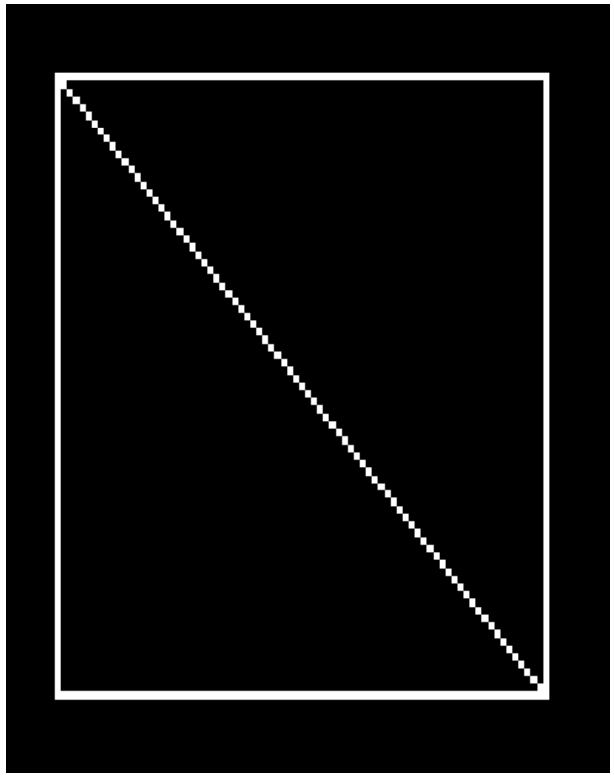
A Hough transzformáció képes parametrizált élszegmenseket (egyenes, kör, ellipszis,…) is megtalálni



Az alapötlet

Mindegyik egyenes leírható egy képlettel

Mindegyik fehér élpont önmagában végtelen sok egyeneshez tartozhatna



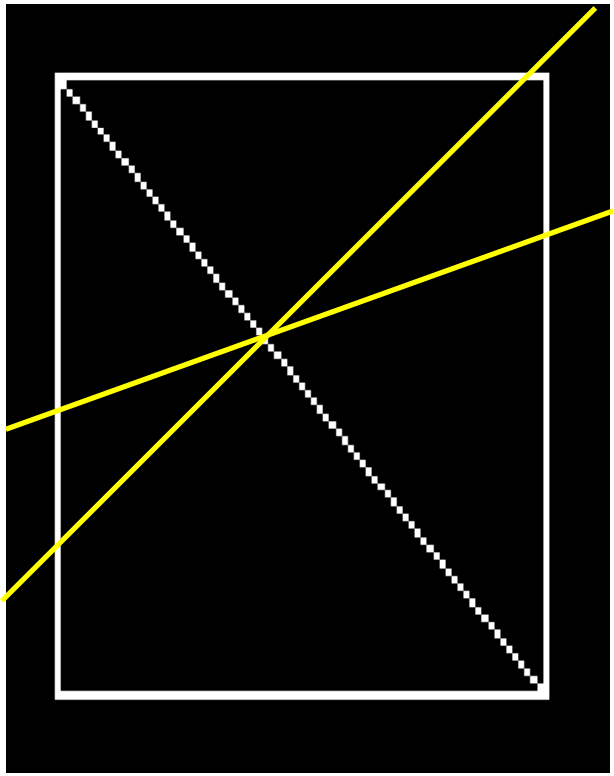
Az alapötlet

Mindegyik egyenes leírható egy képlettel

Mindegyik fehér élpont önmagában végtelen sok egyeneshez tartozhatna

A Hough transzformációs síkban mindegyik élpont „szavaz” azokra az egyenesekre melyekhez tartozhatna

A legtöbb szavazatot kapó egyenesek a befutók

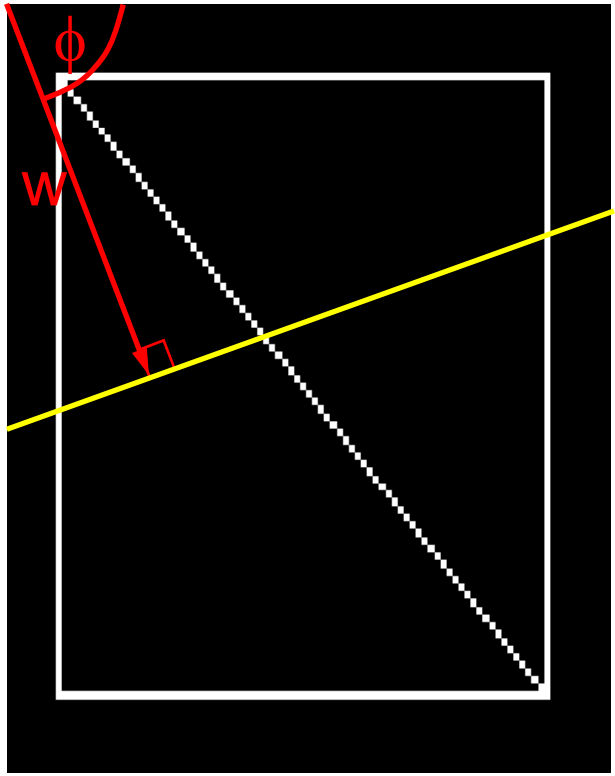


Különféle egyenes reprezentációk a Hesse-féle normálalak

Minden egyenest 2 szám jellemez

A sárga egyenes jellemzése: (w,ϕ)

- w távolság az origótól
- ϕ szög a vízszinteshez képest



Megjegyzés:

Az y=mx+c reprezentációhoz képest az az előny, hogy w csak véges értéket vehet fel.

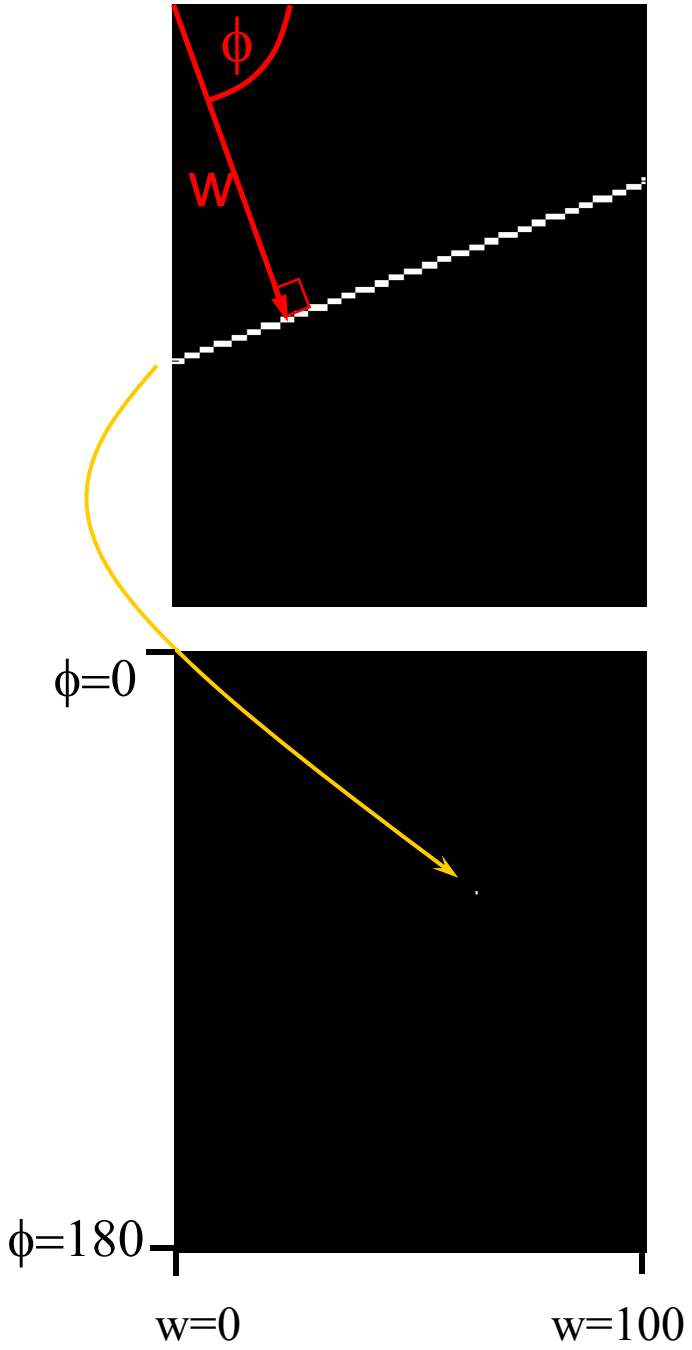
Függőleges egyeneseknél m=∞ lenne (implementációs problémákat okoz)

Amúgy (m,c) térbe is lehetne Hough transzformáció

Hough tér

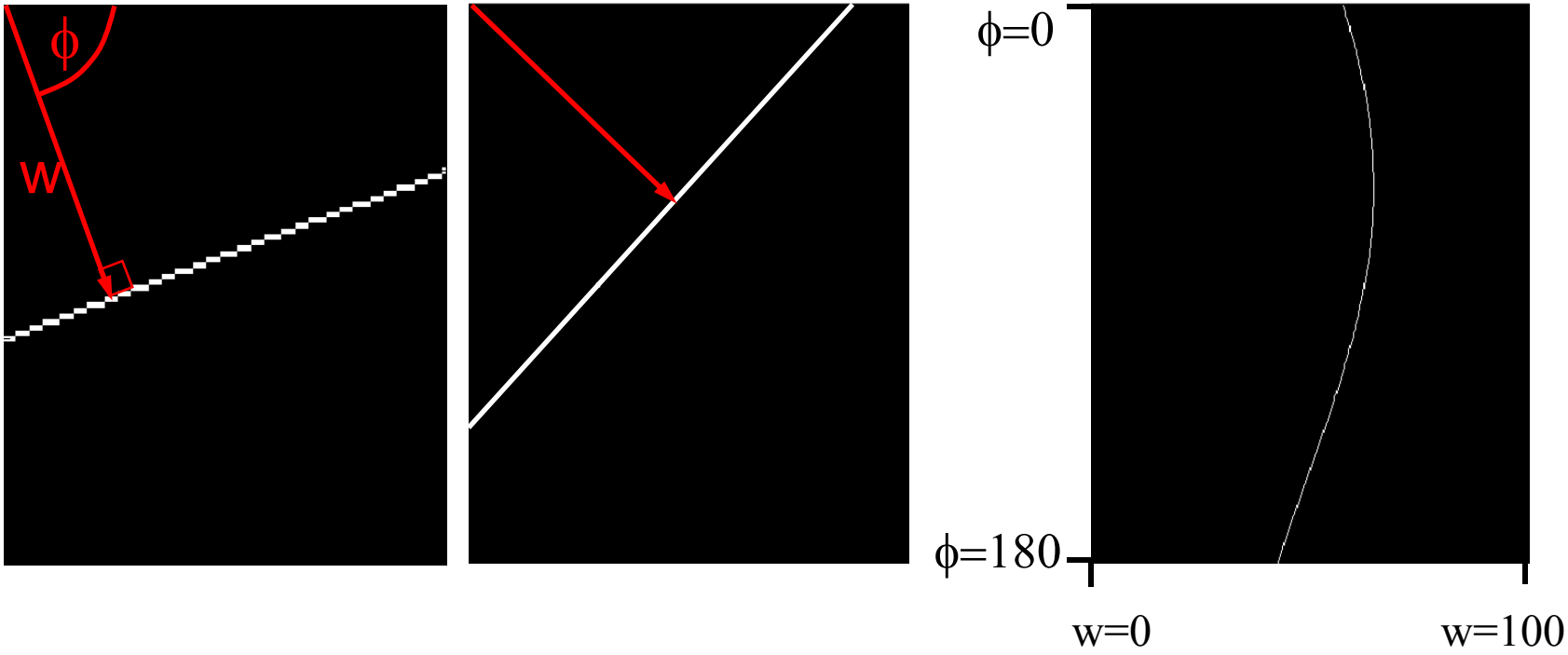
Mivel egy (w,ϕ) érték reprezentál egy egyenest az (x,y) képtérben

egy-egy pont reprezentál egy egyenest a (w,ϕ) Hough térben



Hogyan szavaz egy képtérbeli pont a Hough térben?

$$w = x\cos(\phi) + y\sin(\phi)$$

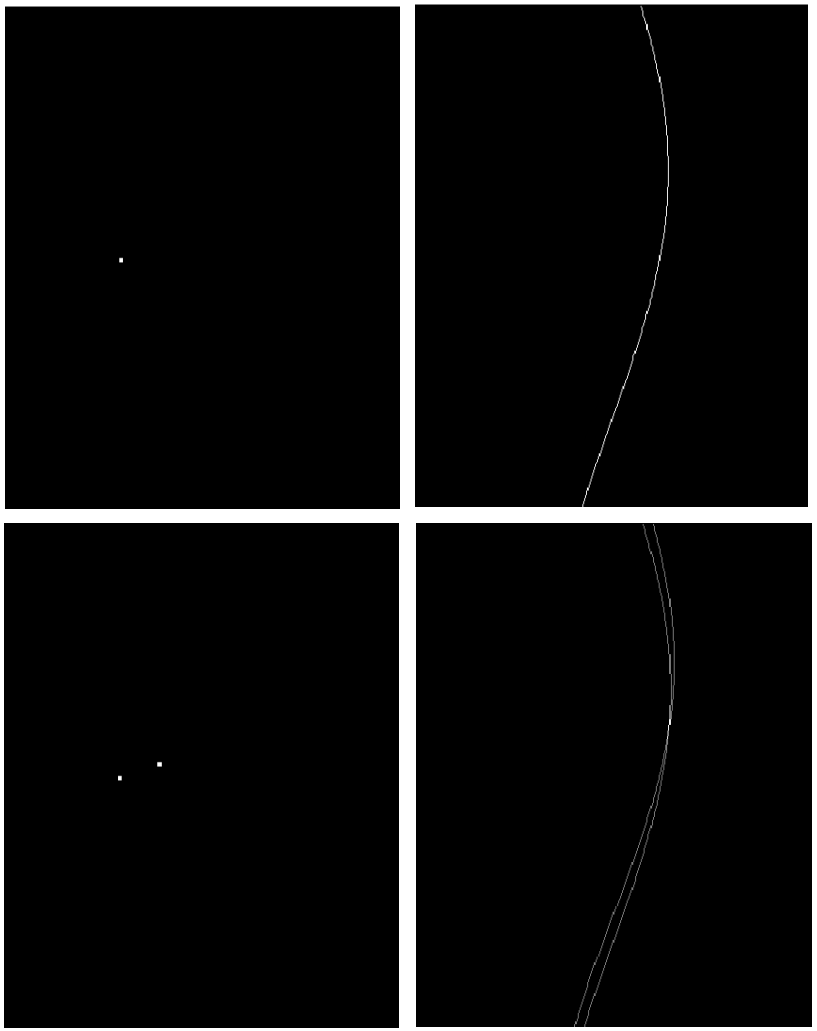


Hogyan szavaz egy képtérbeli pont a Hough térben?

Egy képtérbeli pont egy sinusoid görbének felel meg a Hough térben – az összes a ponton átmehető egyenes (w,ϕ) értékeit megadva

Két képtérbeli pont két görbének felel meg a Hough térben

A két görbének a metszéspontja a Hough térben egy „szavazat” a képtérbeli két pontot összekötő egyenesre



Hough Transzformáció - formalizálva

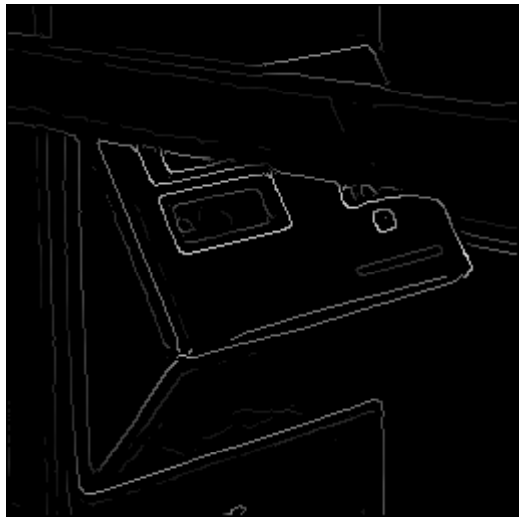
```
Create ϕ and w for all possible lines
Create an array A indexed by ϕ and w
for each point (x,y)
  for each angle ϕ
     w = x*cos(ϕ)+ y*sin(ϕ)
     A[ϕ,w] = A[ϕ,w]+1
  end
end

where A > Threshold return a line
```

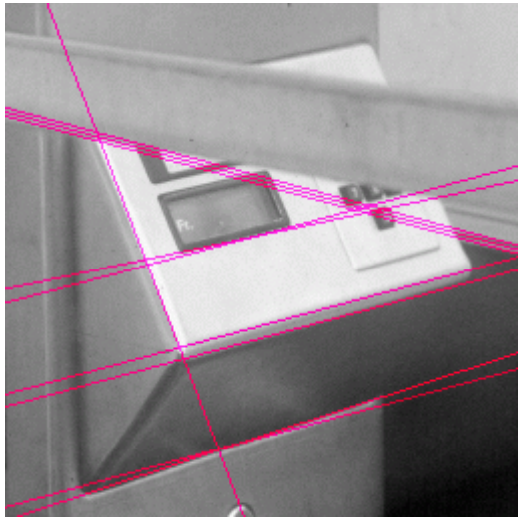
Implementáció: ld. megadott OpenCV források

Hough Transzformáció - példa

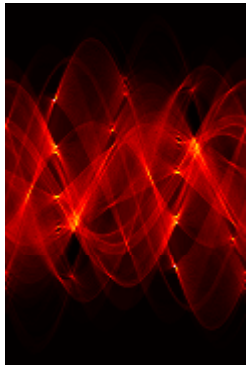
Eredeti kép



Canny éldetektálás



Hough egyenes keresés

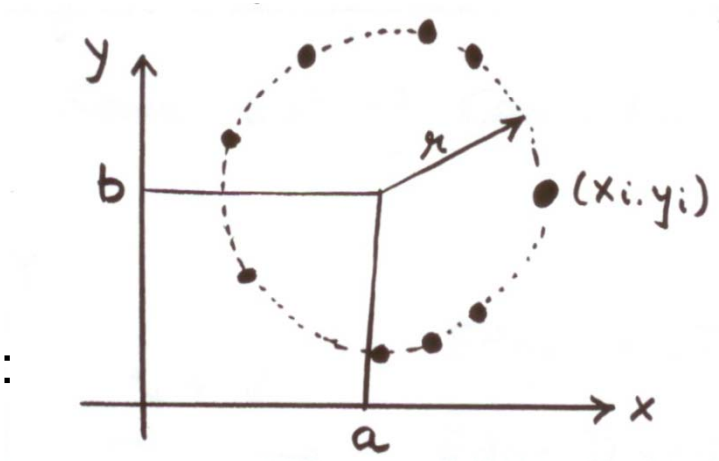


Hough egyenes-paraméter tér

Hough transzformáció kör keresésre

Kör egyenlete:

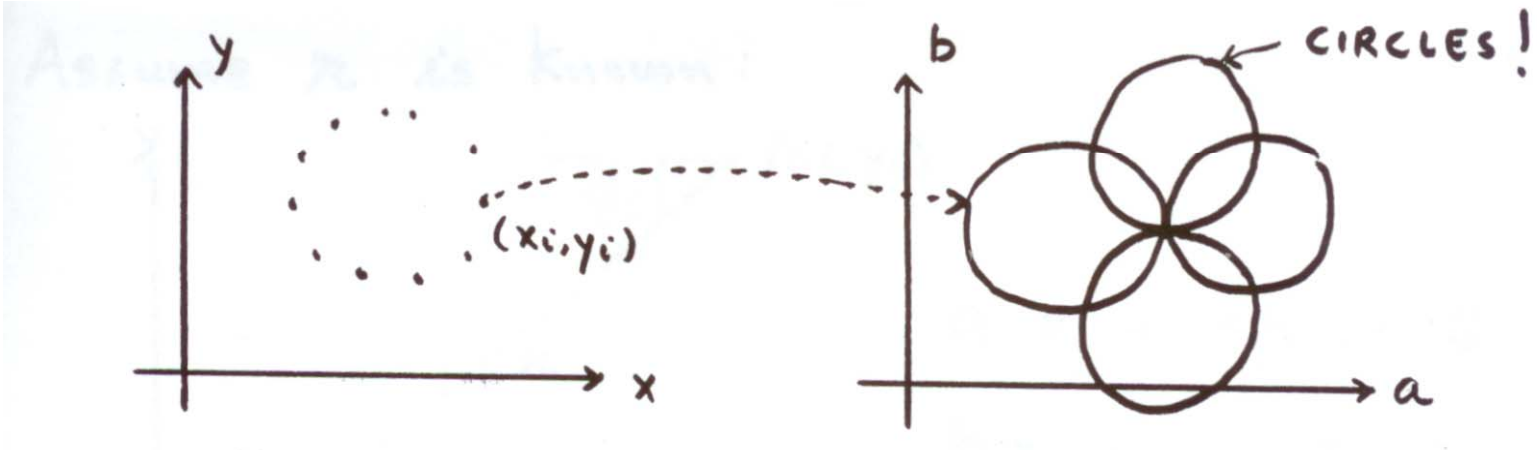
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$



Ha az r sugár előre ismert (az egyszerűbb eset):

A 2D Hough tér koordinátái: (a, b)

A képtér-beli pontoknak Hough tér-beli körök felelnek meg:



Hough Transzformáció

- A módszer kör, ellipszis paramétereire is alkalmazható
- Egyenes kereséskor figyelmen kívül kell hagyni a nem-lokális maximumokat
- At input képet ezért kellett az Canny operátorral és morfológiai zárással előfeldolgozni
- Texturált képre nem jó a Hough transzformáció

Sarokpont detektálás

Egyre összetettebb jellemzők követése?

A már jól ismert skálán továbbhaladva az összetettebb jellemzők irányába:
A „sarokság” egy pontra nem vizsgálható: legalább egy kis ablakban kell!

egyszerű, de nem elég robusztus

intenzítás

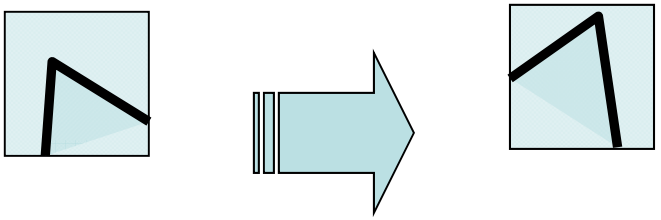
szín

él

textúra

kontúrszegmensek (pl. sarokpontok)

Bonyolultabb, de robusztusabb is



Sarokdetektáló algoritmusok

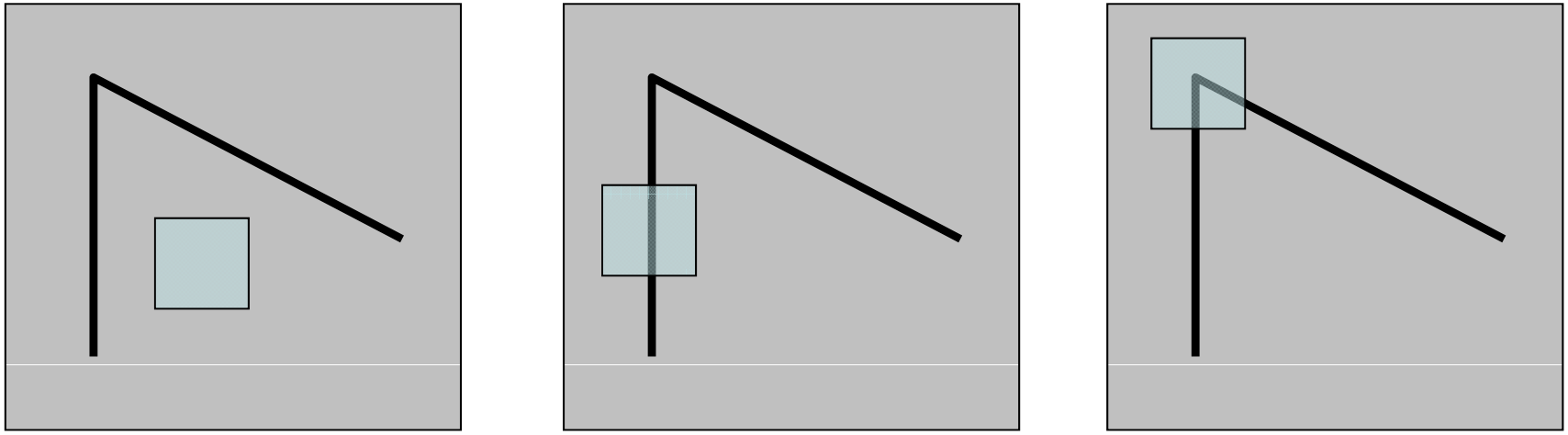
- Moravecz
- SUSAN
- FAST
- Harris - mi majd ezt nézzük meg
- ....

Mitől tekinthető egy jellemző jónak?

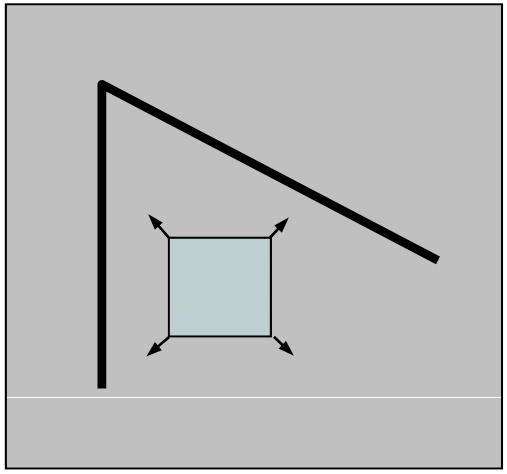
- Egyedi → a következő képen is könnyen beazonosítható

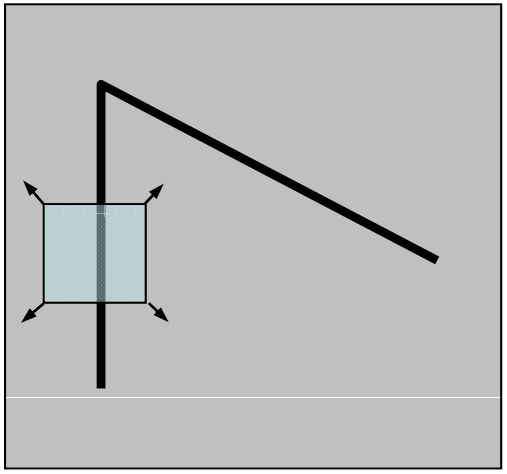
Egyediség keresése lokális ablakban

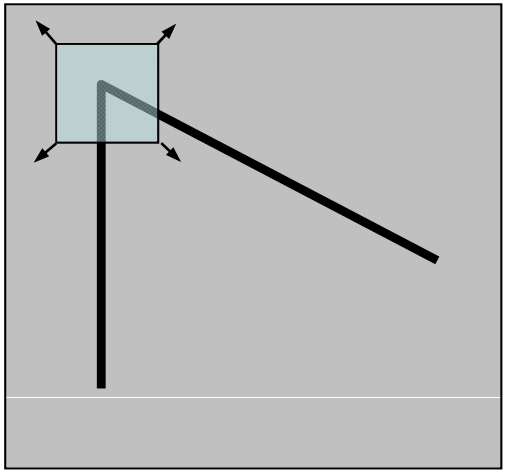
Jó vagy rossz jelölt az adott pont?



Egyediség keresése lokális ablakban







“homogén” régió
Bármely irányban:
Semmi változás!

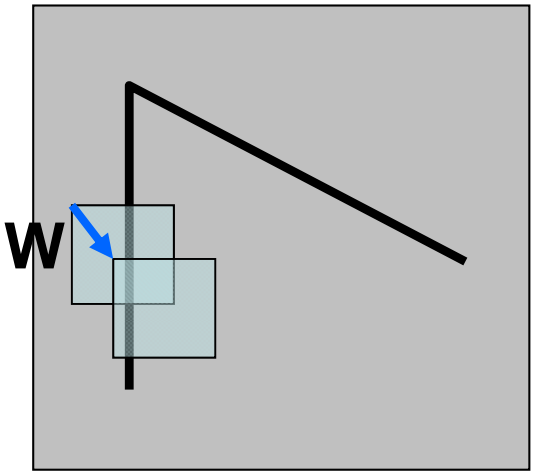
“él”:
Az él mentén nincs változás

“sarok”:
minden irányban jelentős a változás

Egyediség keresése lokális ablakban

W ablakot **kis** (u,v) értékkel elmozgatjuk:

- Hogyan változnak a pixelek **W-ben**?
- Megint SSD “hibát” számolunk: *E(u,v)*:



$$E(u,v) = \sum_{(x,y)\in W} [I(x+u, y+v) - I(x,y)]^2$$

Elsőrendű közelítés kis mozgásokra

Taylor sor:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

Ha (u,v) kicsi, az elsőrendű közelítés jó

$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x,y) + \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

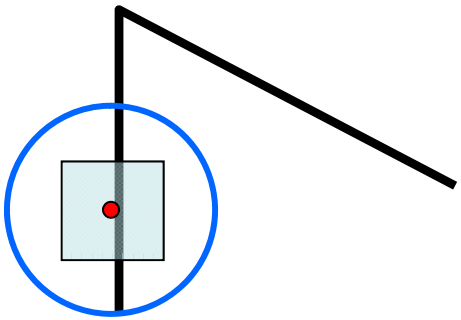
Behelyettesítve az *E(u,v)* kifejezésbe

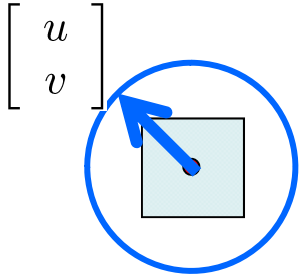
Elsőrendű közelítés kis mozgásokra

$$
\begin{aligned}
E(u,v) &= \sum_{(x,y)\in W} [I(x+u,y+v) - I(x,y)]^2 \\
&\approx \sum_{(x,y)\in W} [I(x,y) + [I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)]^2 \\
&\approx \sum_{(x,y)\in W} \left[[I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix} \right]^2
\end{aligned}
$$

Elsőrendű közelítés kis mozgásokra

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





Az ablakot a körön belül bármely irányba mozgathatom

- Melyik irány fogja a legnagyobb illetve legkisebb E értéket adni?
- H sajátértékei adják meg a választ

(„kis matematika”: sajátérték, sajátvektor)

$$Ax = \lambda x$$

A mátrix **x** sajátvektorai kielégítik az alábbi egyenletet

$$det(A - \lambda I) = 0$$

Ahol λ **x** sajátvektor skalár sajátértéke

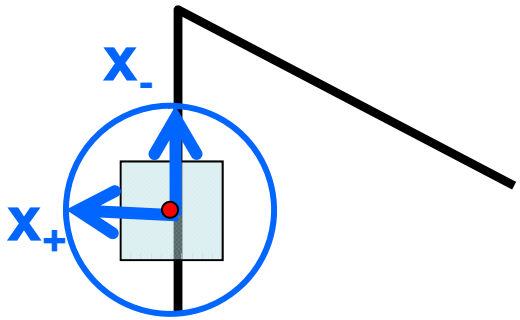
Esetünkben ***A*** *(=* ***H****)* egy 2x2 mátrix

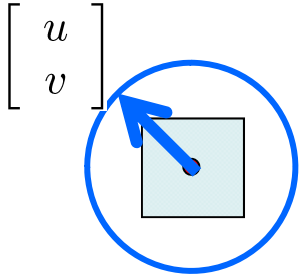
$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

$$\lambda_{\pm} = \frac{1}{2} \left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2} \right]$$

(„kis matematika”: sajátérték, sajátvektor)

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

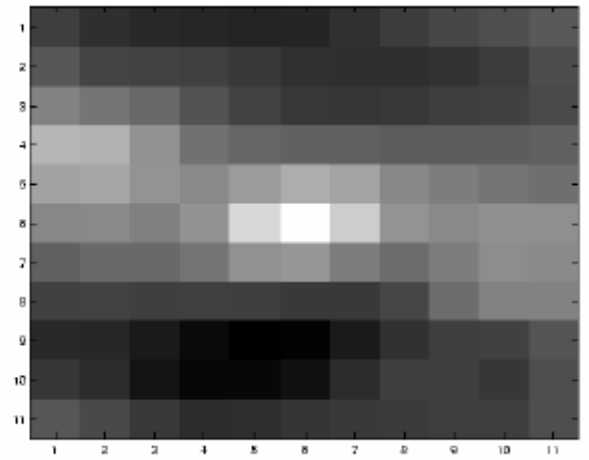


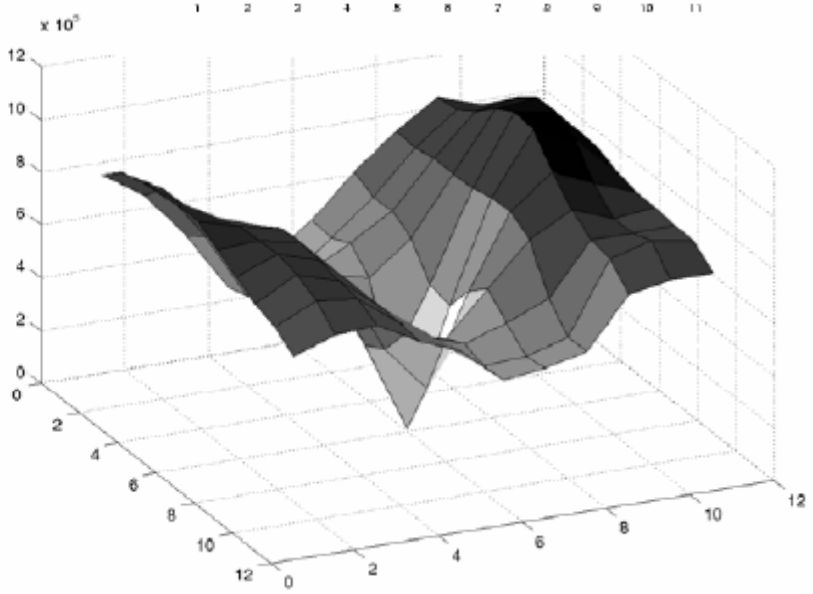


H sajátértékei szerint vizsgálva:

- x_+ = **legnagyobb** E növekményt okozó mozgásirány
- λ_+ = a növekmény x_+ irányban
- x_- = **legkisebb** E növekményt okozó mozgásirány
- λ_- = a növekmény x_- irányban

Gradiensek a texturált részleten

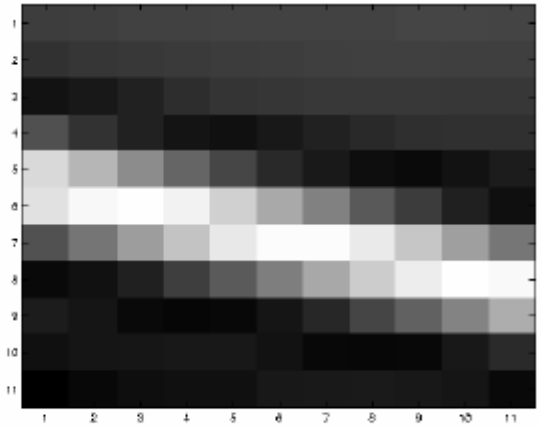


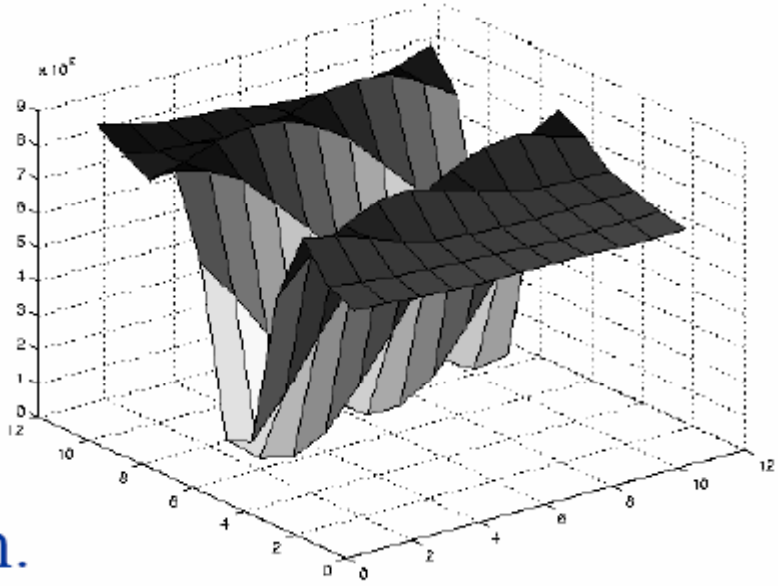


$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix}$$

Gradients in x and y.

Gradiensek az éldús részleten

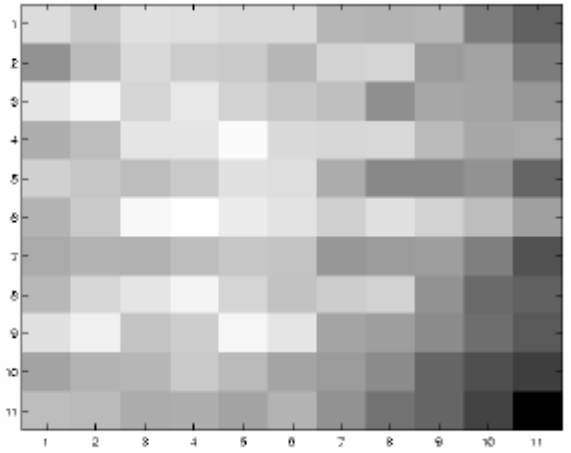


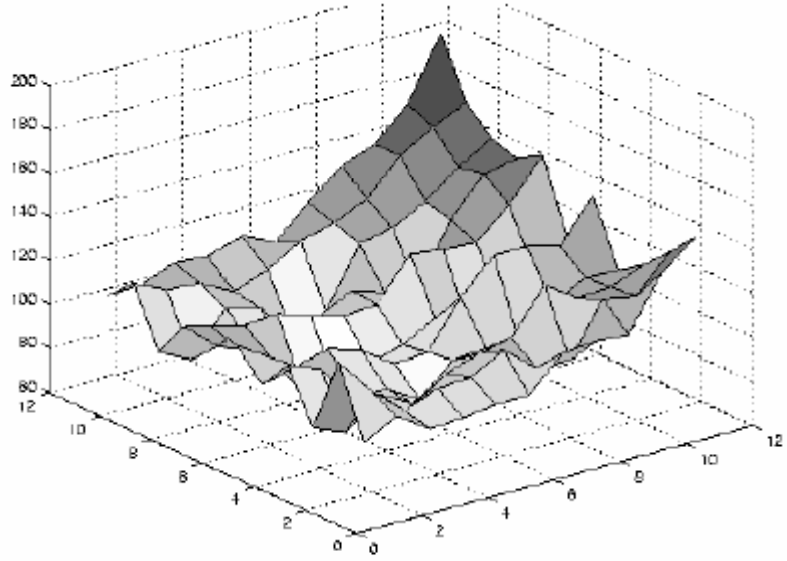


$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix}$$

Gradients oriented in one direction.

Gradiensek a homogén részleten





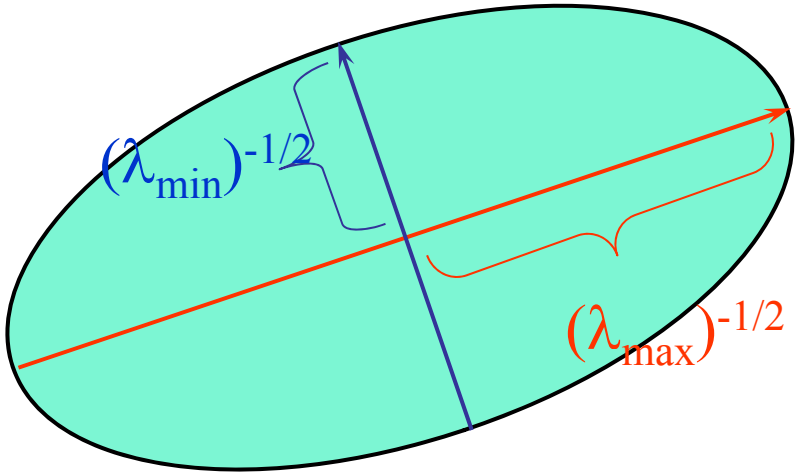
$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix}$$

Weak gradients everywhere.

Mi lehet a „sarokság” kritériuma?

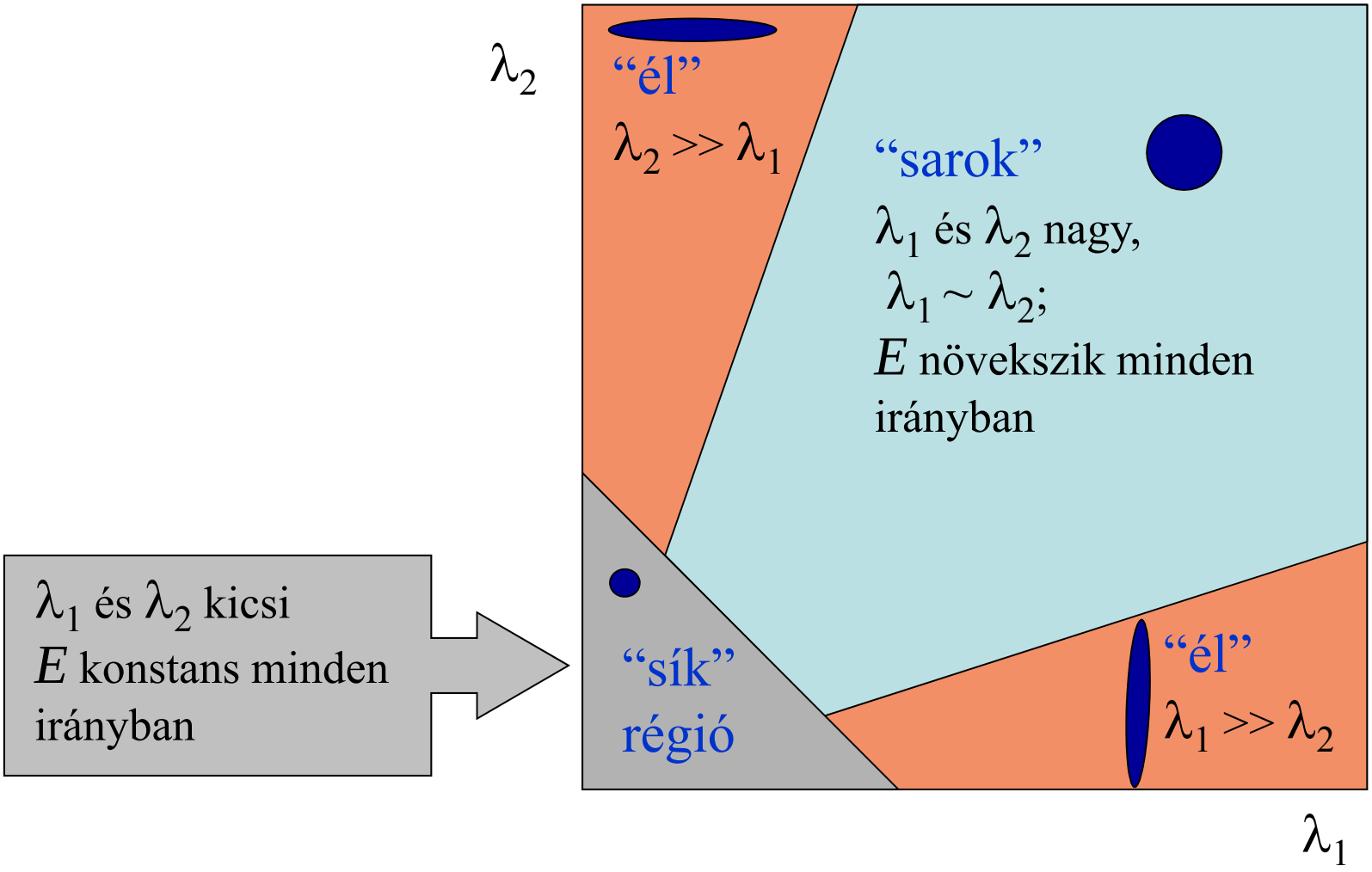
Minden irányban próbálkozva:
A minimális intenzitás változás λ_-
A maximális intenzitásváltozás λ_+

egy ellipszis **nagysága** és **iránya**
jellemző a pont „sarokság” mértékére



Mi lehet a „sarokság” kritériuma?

Képpontok **osztályozhatók** a sajátértékek szerint:

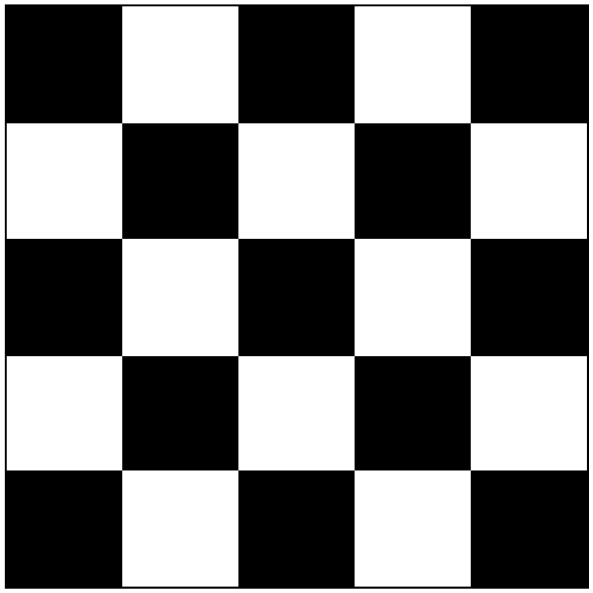


Mi lehet a „sarokság” kritériuma?

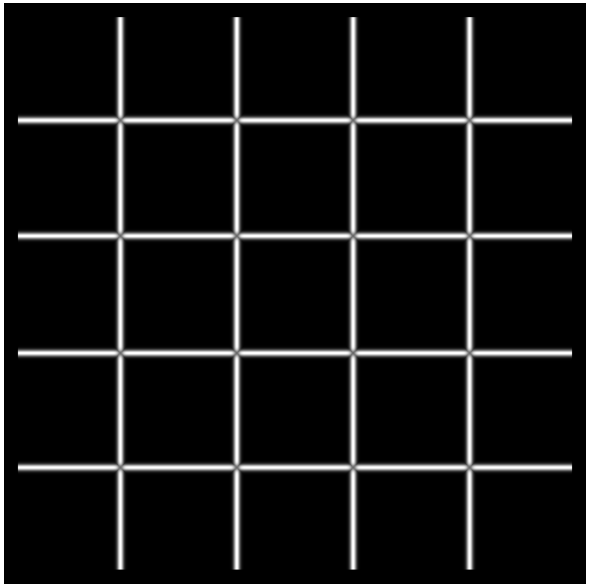
λ_+, $\mathbf{x}_+$, λ_-, $\mathbf{x}_-$ alapján mi legyen a kritérium?

E(u,v) legyen nagy kis elmozdulásokra, minden irányban

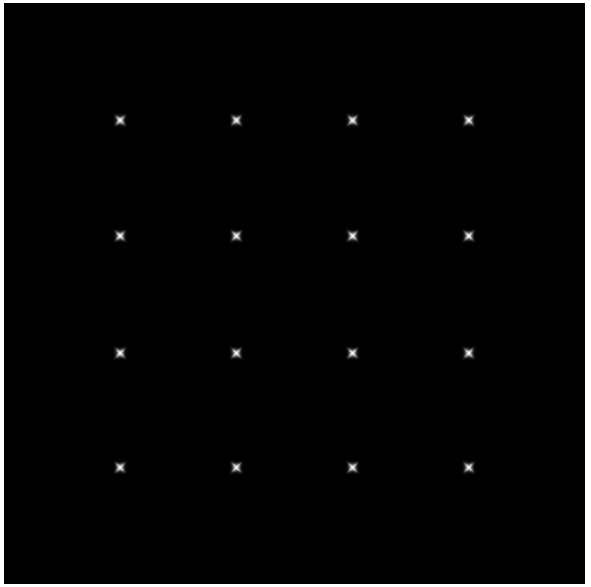
- A minimum értéket ***H*** kisebbik (λ_-) sajátértéke határozza meg



I



λ_+

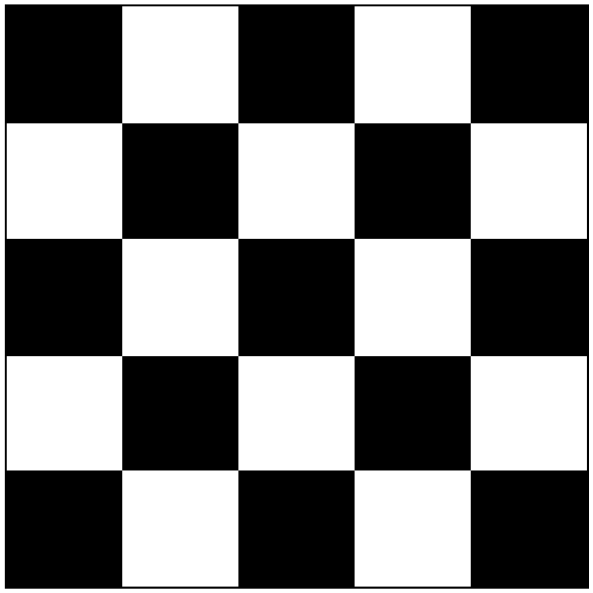


λ_-

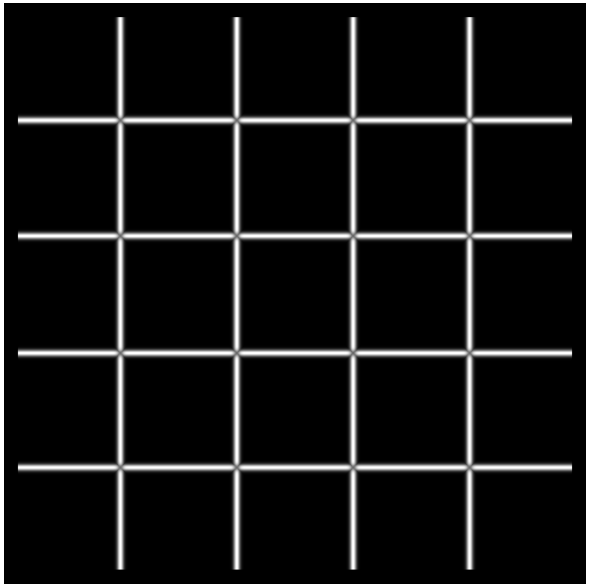
Mi lehet a „sarokság” kritériuma?

Algoritmus összegzése:

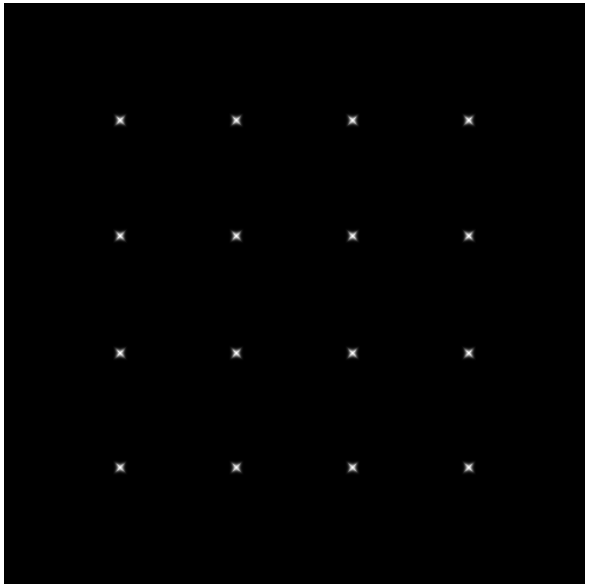
- Minden pontban kiszámoljuk a gradienseket
- A gradiens értékekből számoljuk ***H*** mátrixokat
- Kiszámoljuk a sajátértékeket
- Megkeressük a (λ_- > küszöb) válaszú képpontokat
- Sarokpontnak választjuk a helyi maximum λ_- értéket produkáló pontokat



I

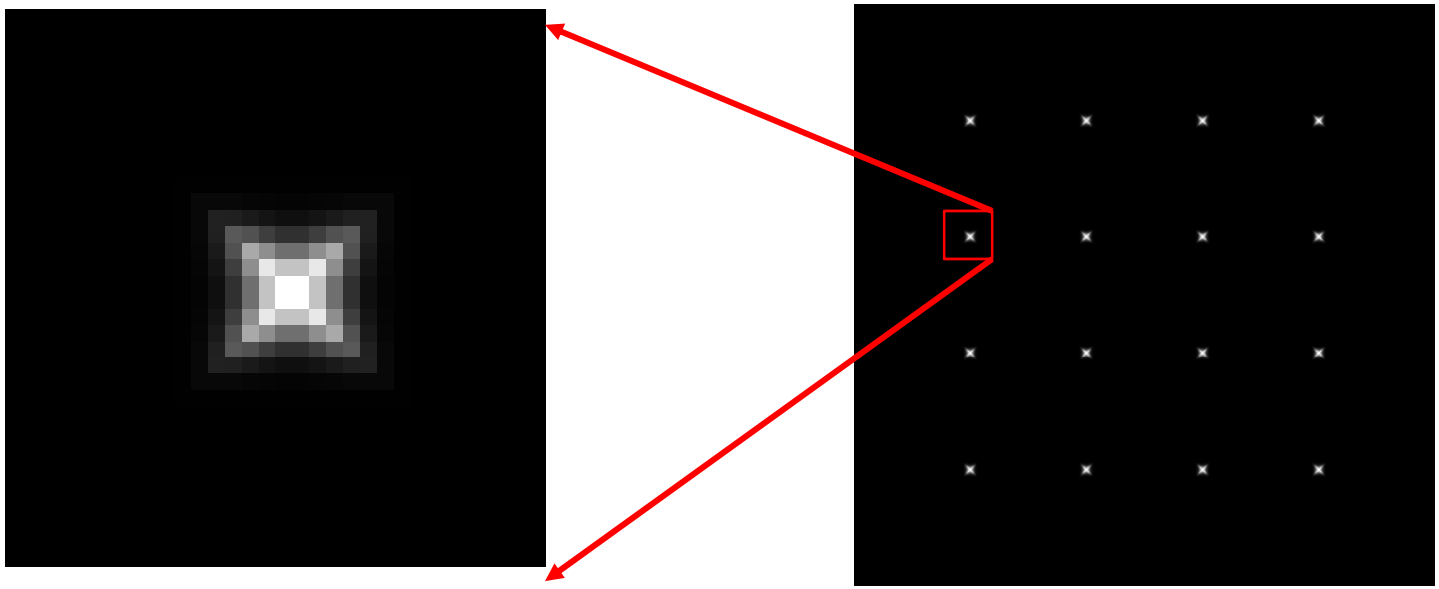


λ_+



λ_-

Mi lehet a „sarokság” kritériuma?



$\lambda_{\_}$ kinagyitva

$\lambda_{\_}$

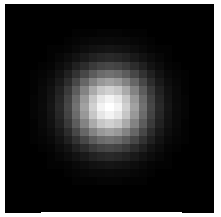
A gradiensek súlyozása

Szimpla W a gyakorlatban nem optimális

A központi pixeltől való távolság szerint súlyozhatunk

$$H = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

$$H = \sum_{(x,y)\in W} w_{x,y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



$w_{x,y}$

Harris operátor - a definició

R is a „sarokságra” jellemző mérték, ahol:

$$R = \det H - k(\text{trace } H)^2$$

trace a diagonálisok összege trace(H) = $h_{11} + h_{22}$

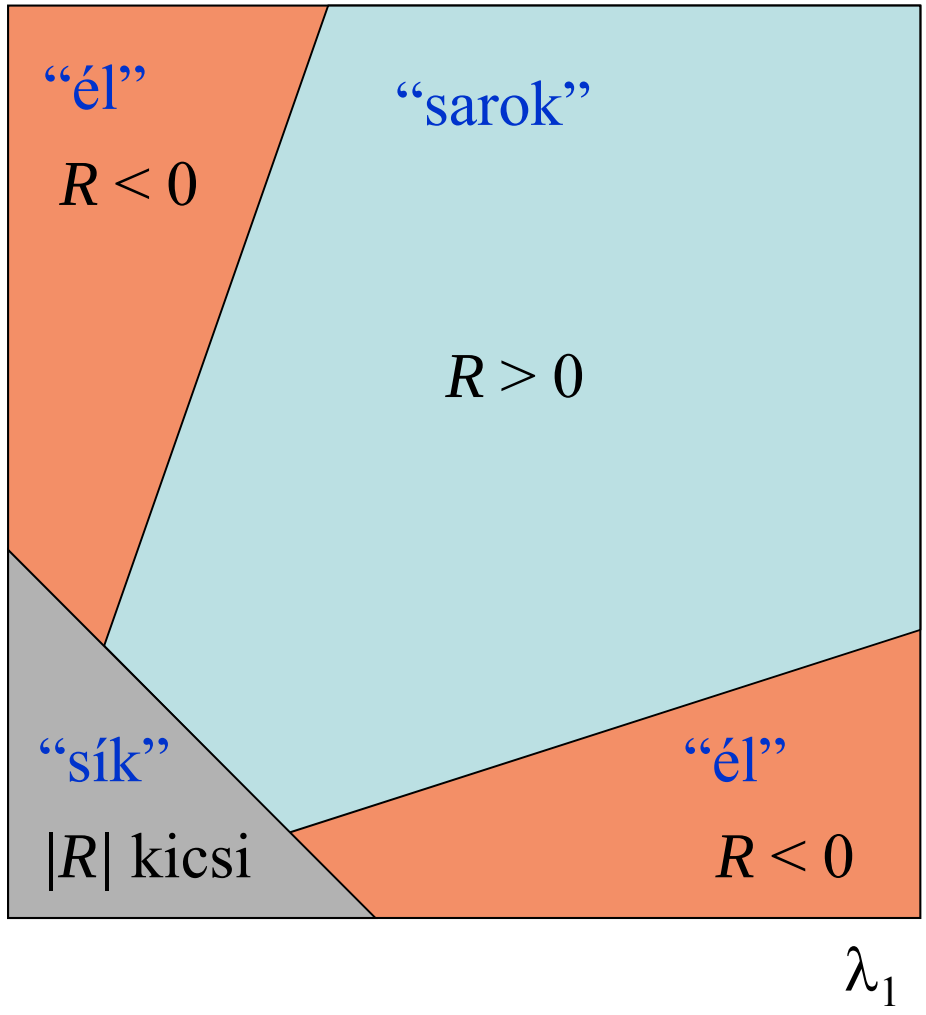
R hasonló eredményt ad, csak sokkal könnyebb kiszámolni

(k egy empírikus konstans = 0.04-0.06)

Harris operátor - a sajátértékek szerint

Képpontok **osztályozhatók** R szerint is:

1. R is csak H sajátértékeitől függ
2. R nagy érték sarok esetében
3. R negatív él esetében
4. |R| kicsi homogén rész esetében

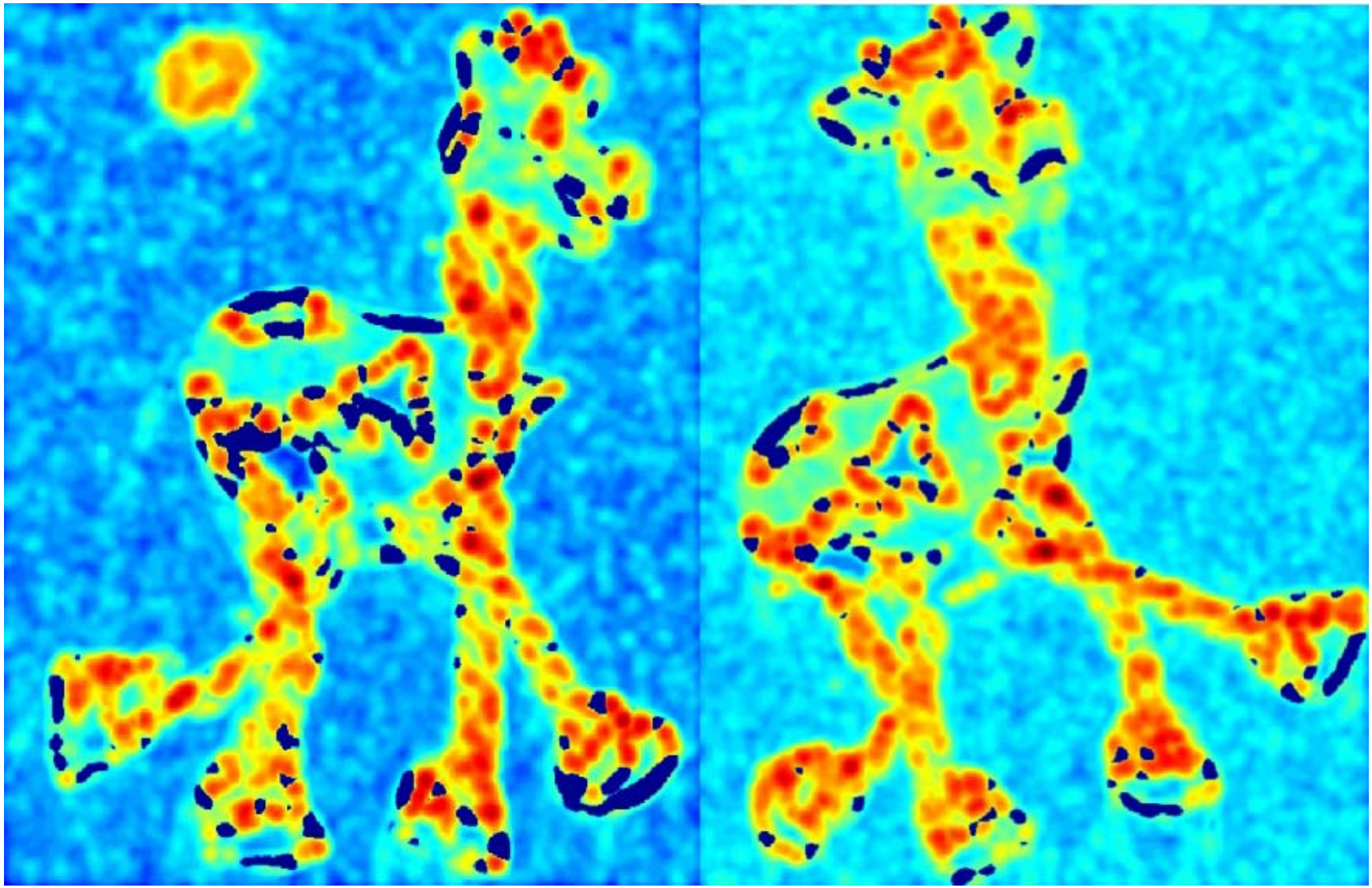


Harris operátor – a workflow

0. lépés: a kiindulási

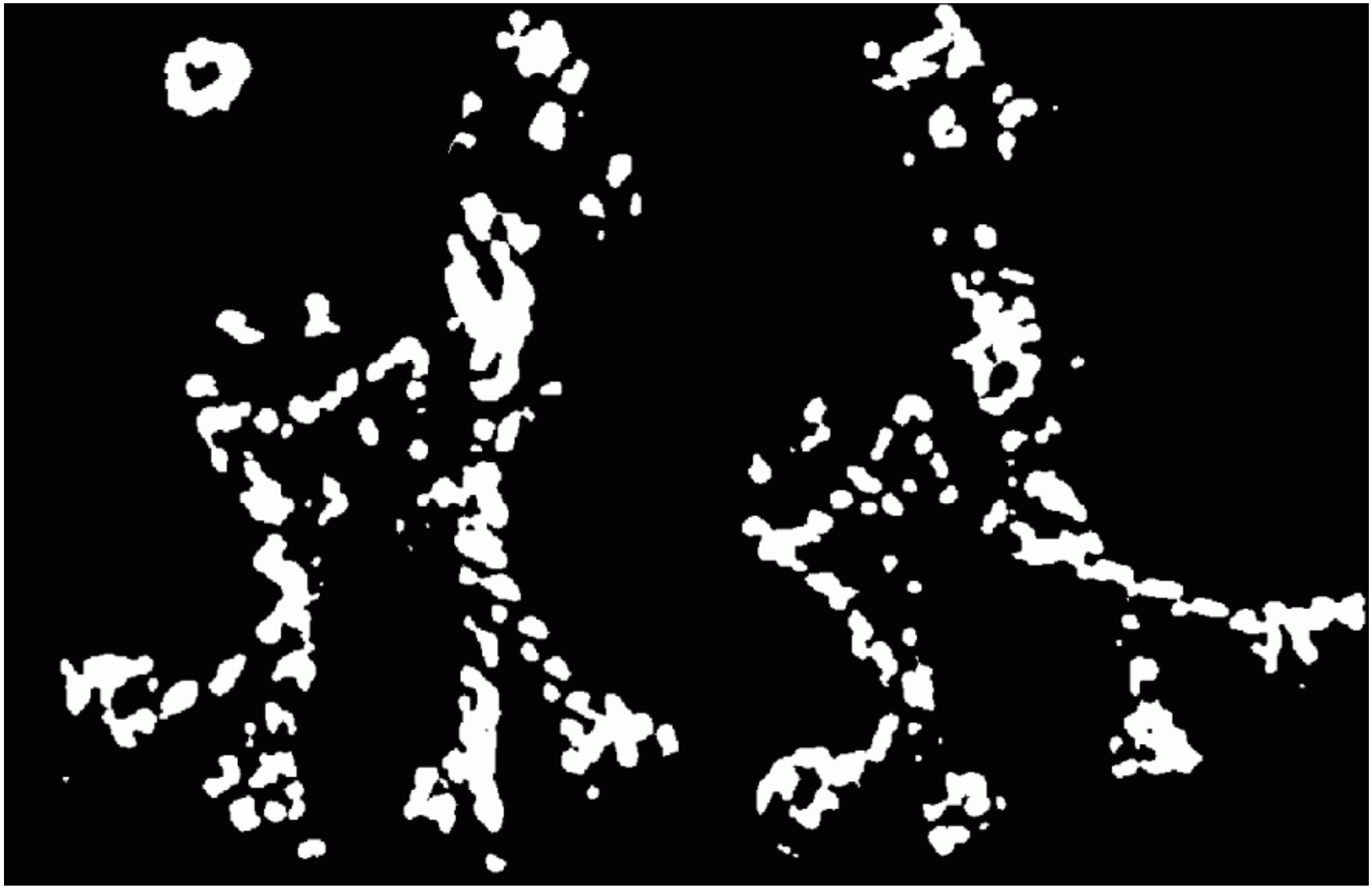
Harris operátor– a workflow

1. lépés: Kiszámolni az R értékeket



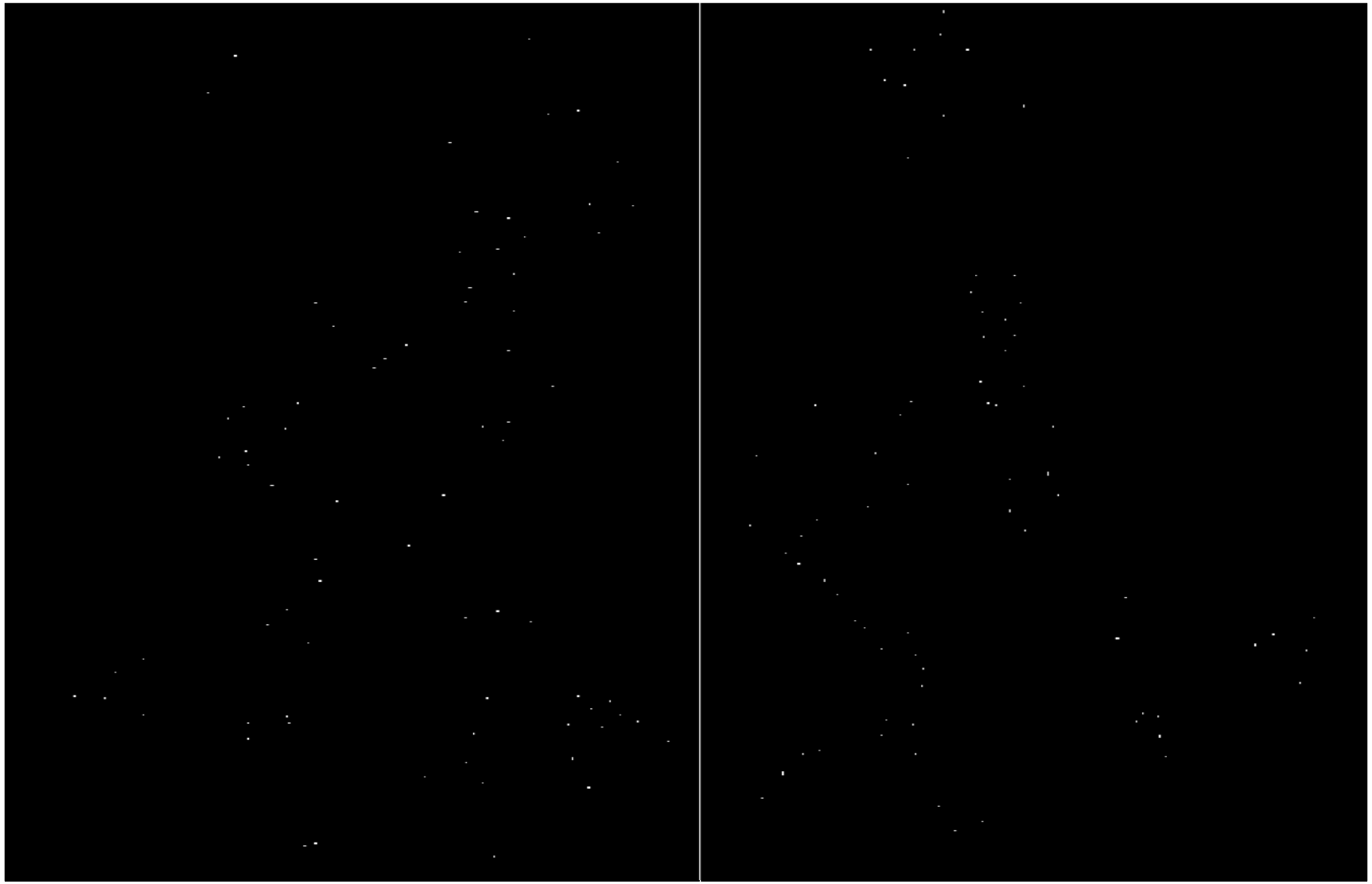
Harris operátor – a workflow

2. lépés: *Megtartani, ahol R>küszöb*



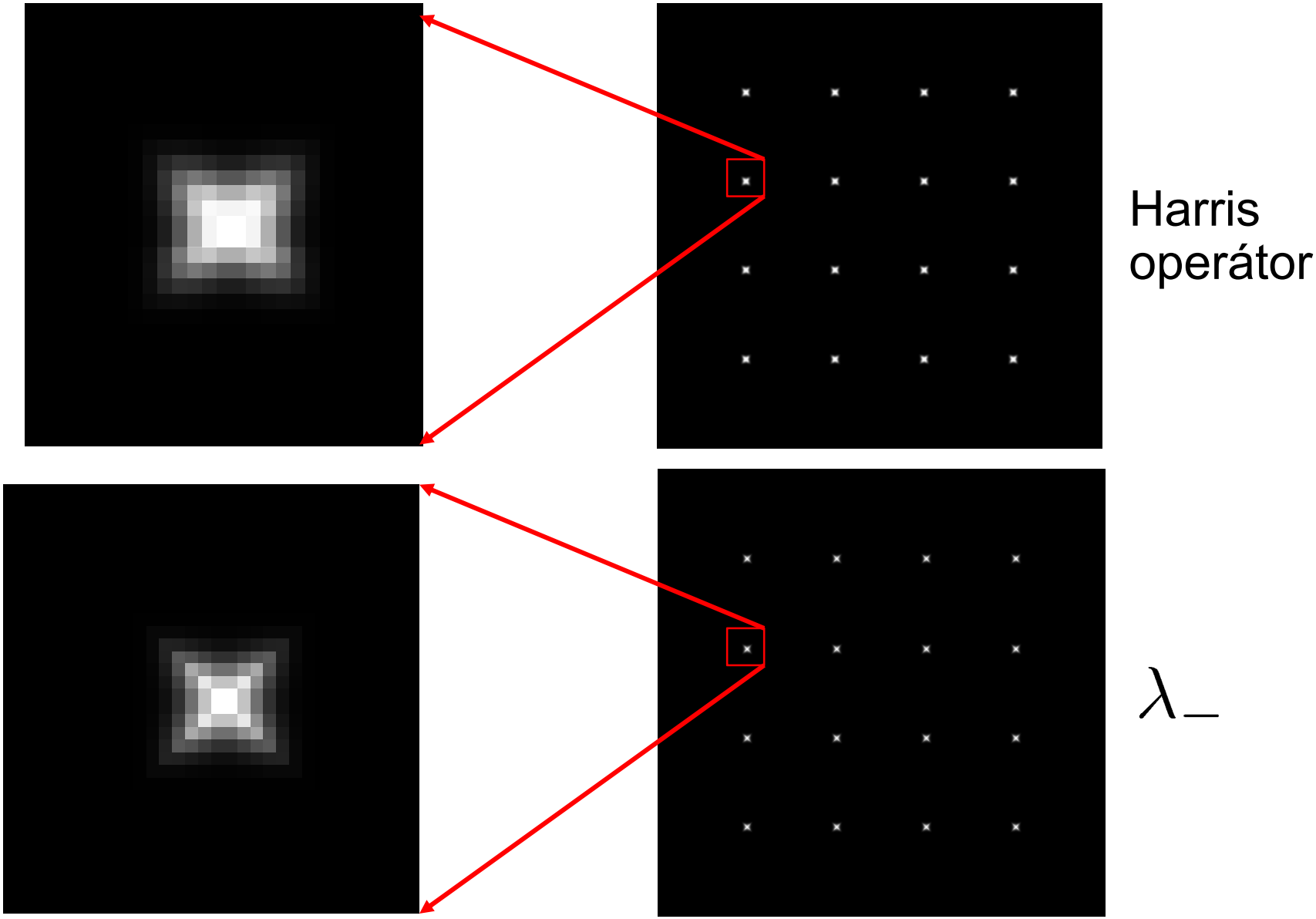
Harris operátor – a workflow

3. lépés: Csak a lokális R maximumokat megtartani

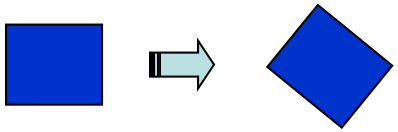
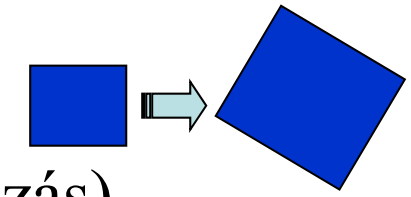
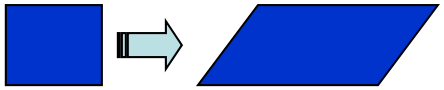
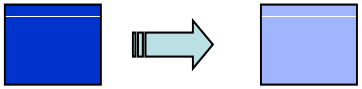


Harris operátor – a workflow

A Harris operátor vs. közvetlenül a sajátértékek szerinti sarokpont keresés összehasonlítása

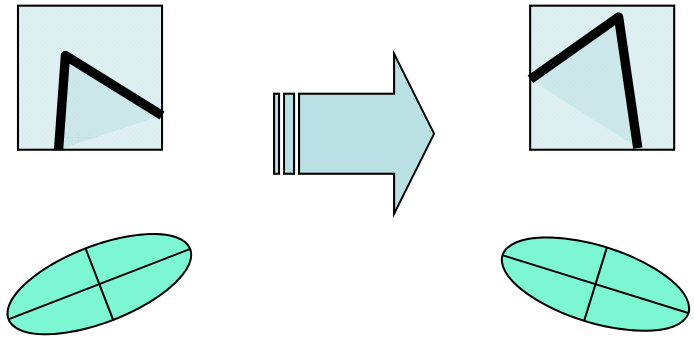


Képtorzítás alaptípusok

- Elforgatás 
- Hasonlóság (elforgatás+ uniform skálázás) 
- Affin (irányfüggő) skálázás 
- Affin intenzitás változás ($I \rightarrow a\ I + b$) 

Harris operátor robusztussága

- Elforgatás: invariancia



Az ellipszis együtt „forog”, de az alakja (sajátértékek) nem változnak

R sarokság mérték invariáns az elforgatásra!

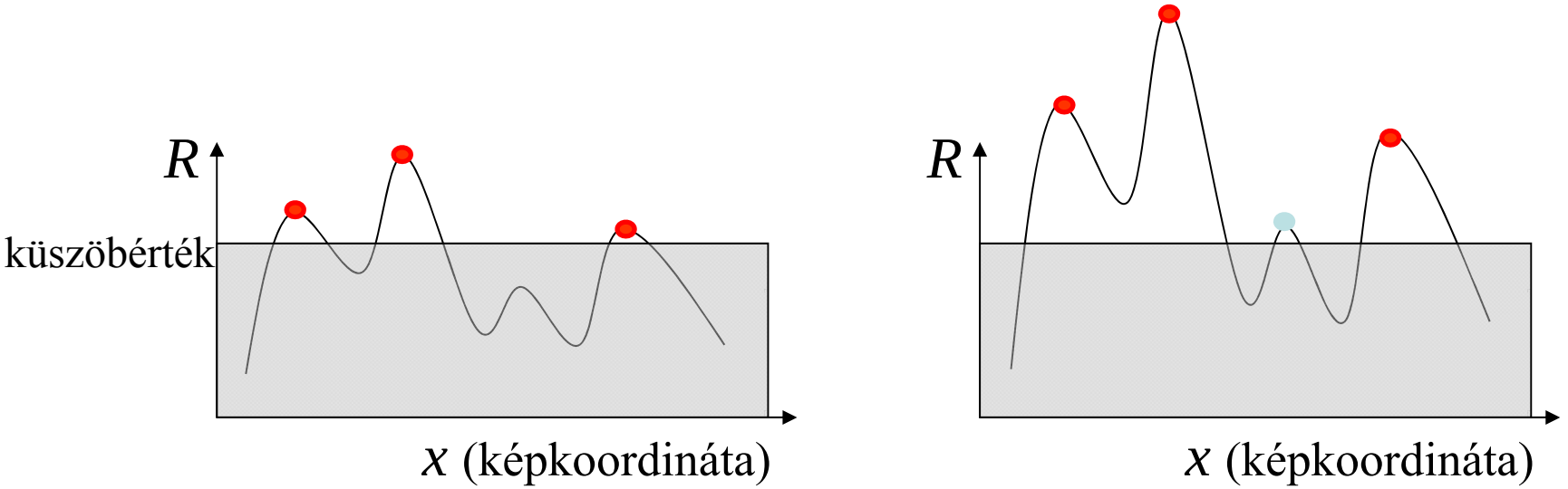
Harris operátor robusztussága

- Intenzitásváltozás: részleges invariancia

mivel intenzitás deriváltakkal dolgozunk =>

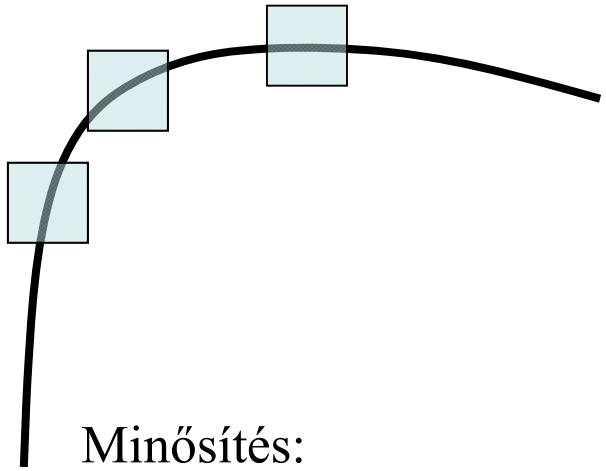
✓Invariancia: $I \rightarrow I + b$ esetén (eltolás)

✓Invariancia: $I \rightarrow a\ I$ esetén (skálázás)



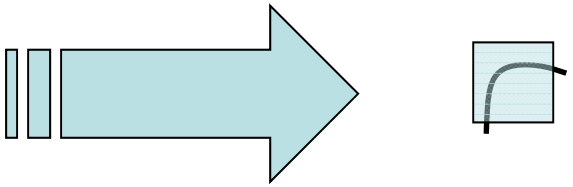
Harris operátor robusztusága

- de: NEM invariáns a képskálázásra!



Minősítés:

ÉLPONTOK

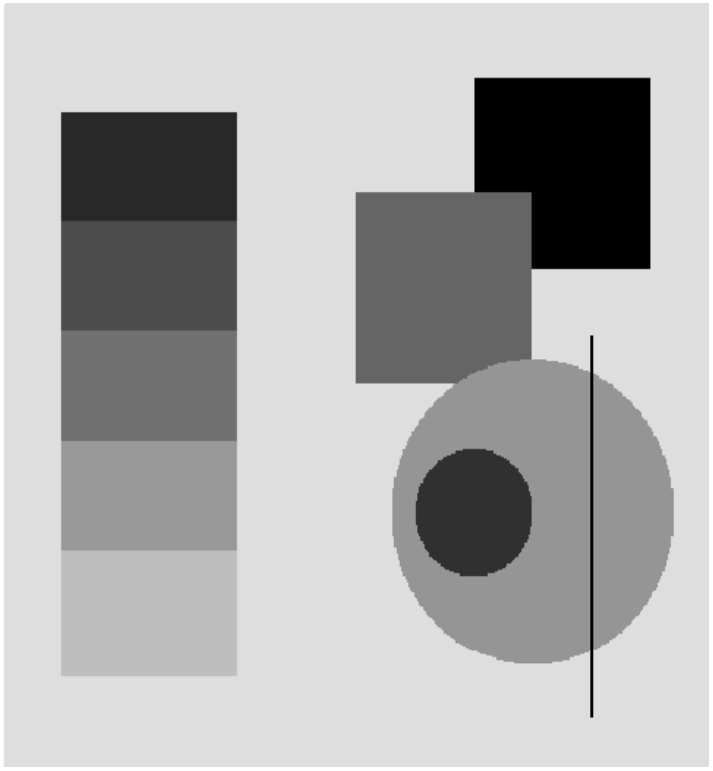


Minősítés:

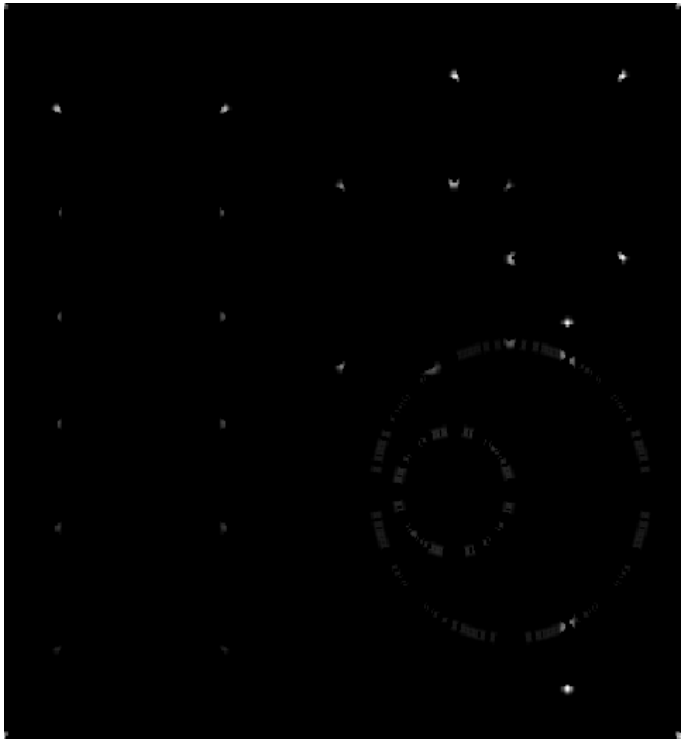
SAROKPONT

Harris operátor robusztusága

A Harris operátor - érzékeny a kontrasztra



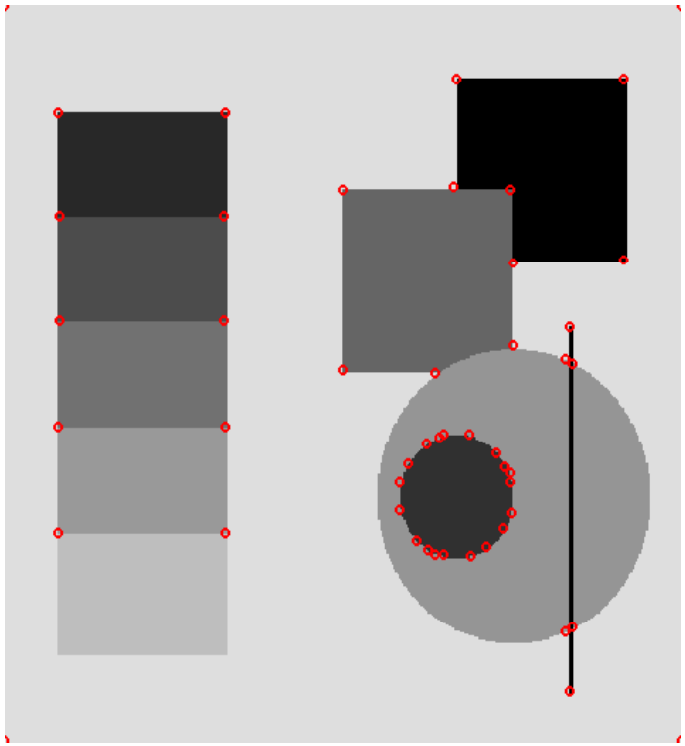
Tesztkép



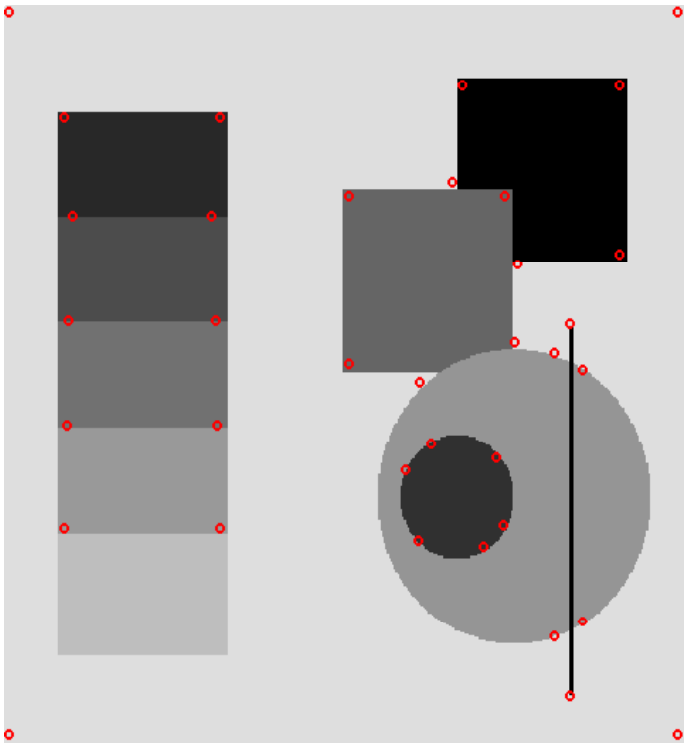
Harris „sarokság erősség” ábrázolása
egy nemlineáris skálán
(probléma kvalitatív illusztrálása)

Harris operátor robusztusága

A Harris operátor érzékeny a skálázásra



1x

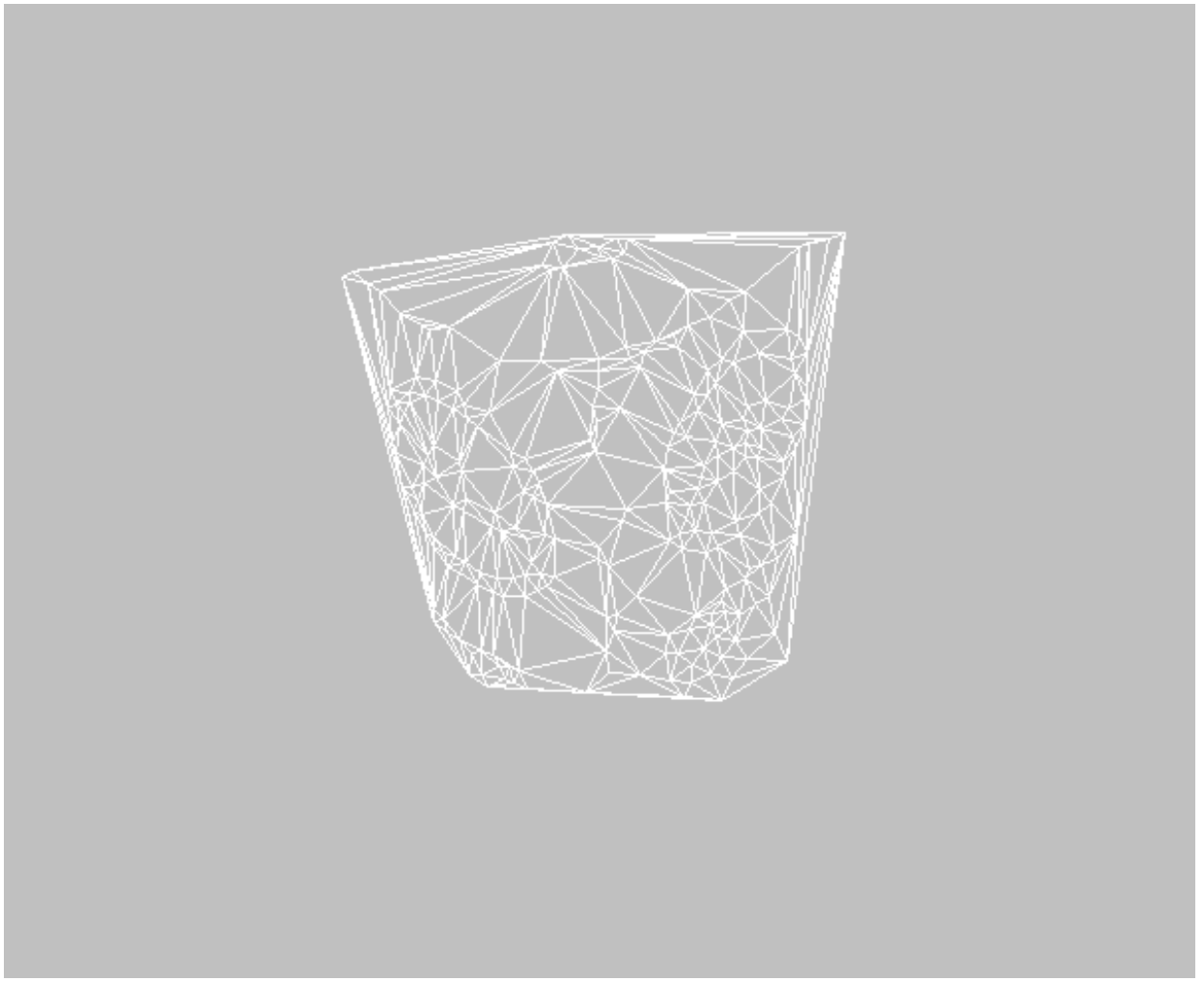


7x

Harris illusztráció: Mozgáskövetés

(az Oktatási Portálon található pdf verzióban sajnos nem látszik a mozgás robusztus követése)

Lehetséges alkalmazás: 3D rekonstrukció



Minta detektálás

Texturális jellemzők követése

Több (együttálló) intenzitást (pixelt) is próbálhatunk követni szimultán

egyszerű, kevésbé robusztus

intenzitás

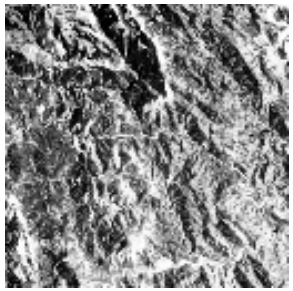
szín

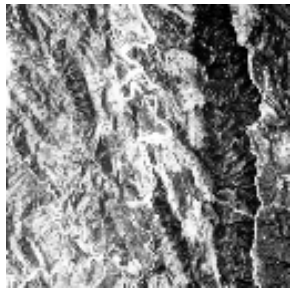
élek

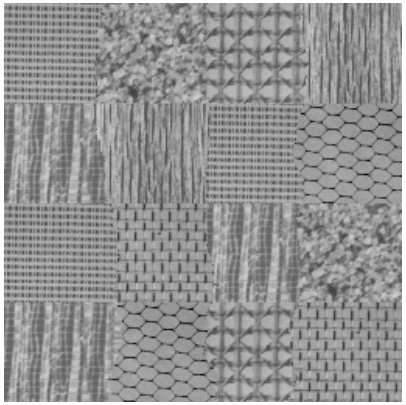
textura (folt/minta)

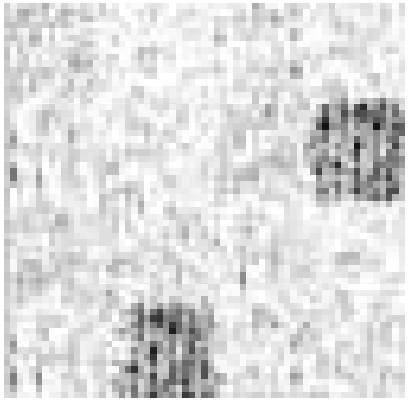
...

bonyolultabb, robusztusabb









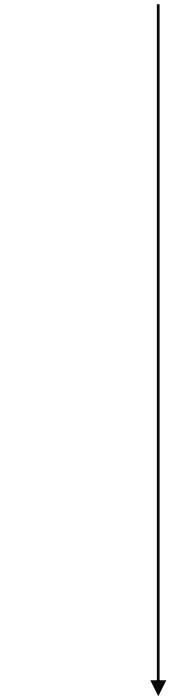
A követendő textúra (folt) akár lehetne egy felismerhető arcrészlet (minta) is...

Didaktikailag nincs jelentősége, de akár: gyalogos, sofőr mozgáskövetése (pl. hova figyel,….)

Texturális jellemzők követése

Több (együttálló) intenzitást (pixelt) is próbálhatunk követni szimultán

egyszerű, kevésbé robusztus



intenzitás

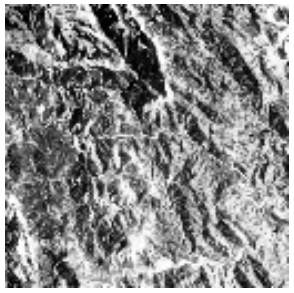
szín

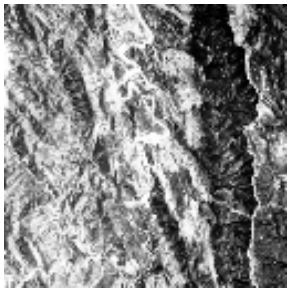
élek

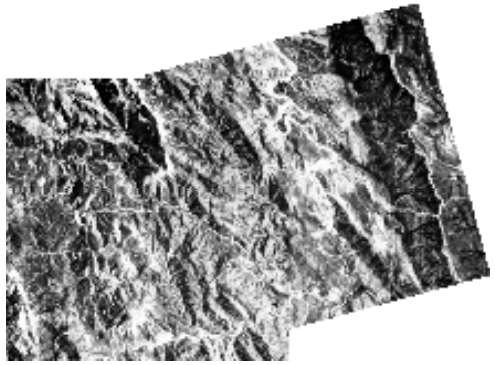
textura (folt/minta)

...

bonyolultabb, robusztusabb



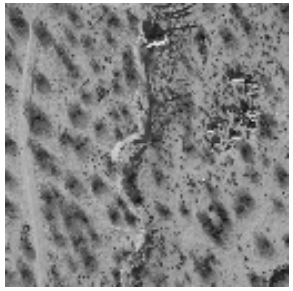
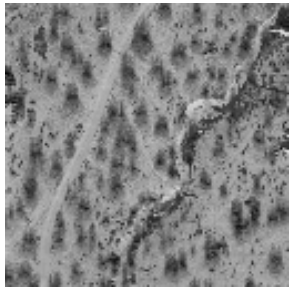
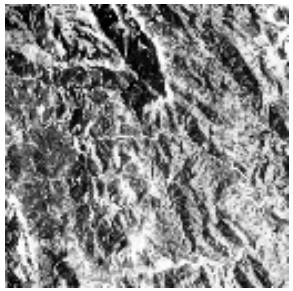
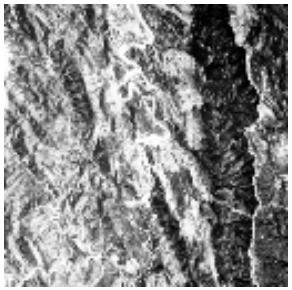




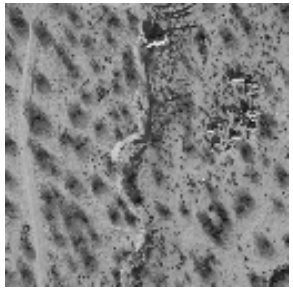
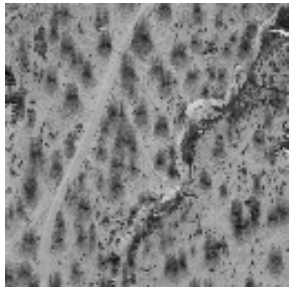
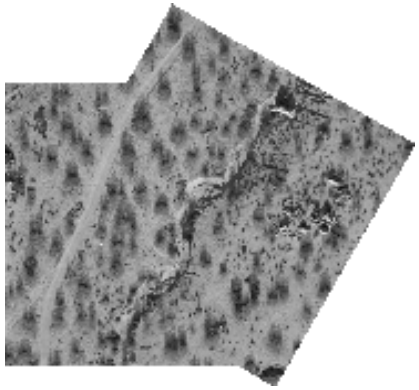
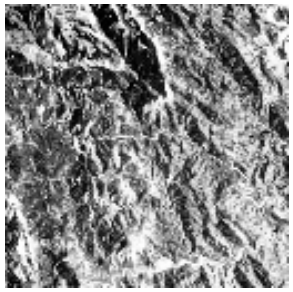
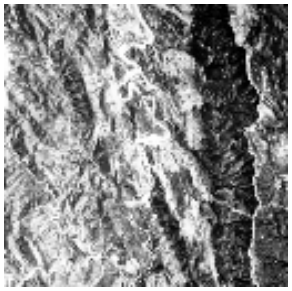
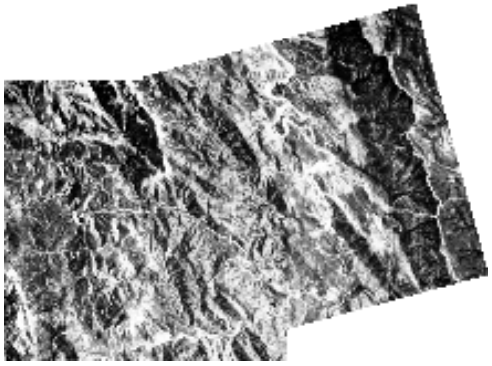
A követendő textúra (folt) akár lehetne egy felismerhető arcrészlet (minta) is...

Didaktikailag nincs jelentősége, de akár: gyalogos, sofőr mozgáskövetése (pl. hova figyel,….)

Textura-, képmozaikok illesztése

Textura-, képmozaikok illesztése

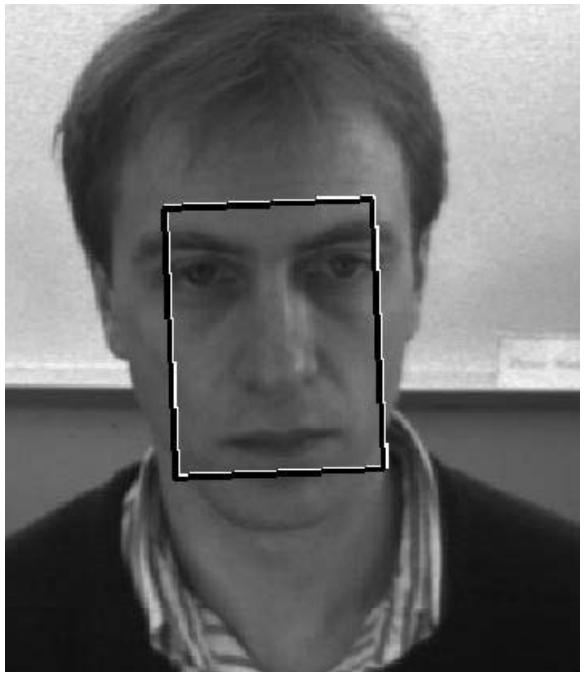
Textura követés

Örökérvényű „jó tanács”:

“If brute-force doesn’t working, you’re not using enough… .”

mintakép

A kezdő képkockán megtalálva



Mit keressünk? & Hogy keressük?

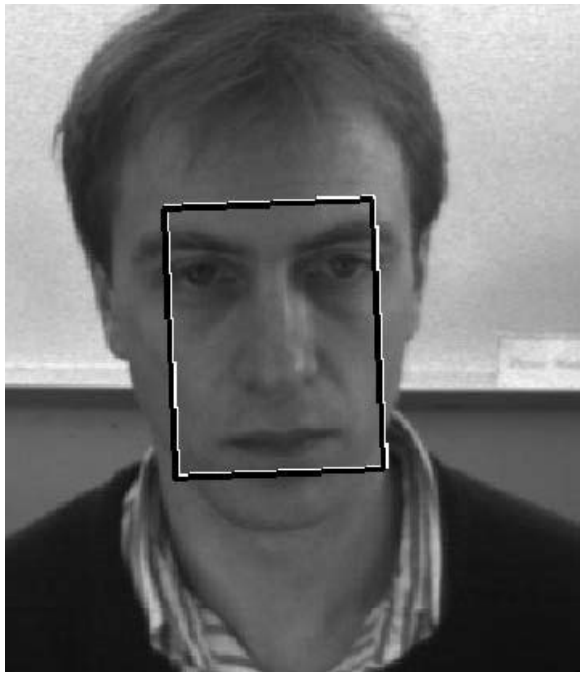
Textura követés

SSD (Sum-of-Squared-Differences) hiba minimalizálással keressük meg a legjobb illeszkedést

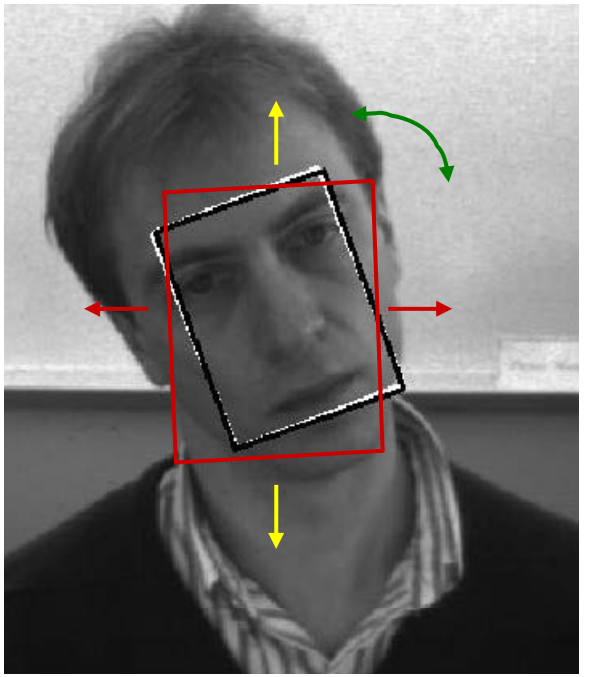
---

mintakép

A kezdő képkockán megtalálva



Piros keret: a kiinduló helyzet a következő képen



Esetünkben feltétlen valósidejűség kell: és a keresési idő mennyi?

SSD követési algoritmus

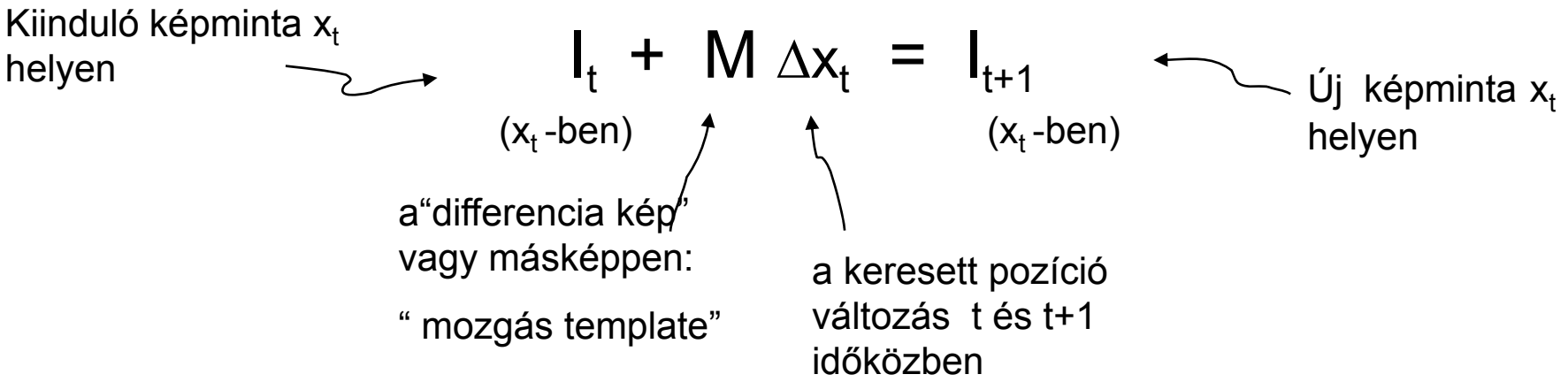
Közelítő feltételezés: a képváltozás **lineáris** függvénye az elmozdulásnak translation. (Természetesen minden közelítő modell „rossz” valamennyire)

$$I_t + M\,\Delta x_t = I_{t+1}$$

(x_t -ben) (x_t -ben)

SSD követési algoritmus

Közelítő feltételezés: a képváltozás **lineáris** függvénye az elmozdulásnak translation. (Természetesen minden közelítő modell „rossz” valamennyire)



A közelítő feltételezés mellett $\Delta x_t = ?$

SSD követési algoritmus

Közelítő feltételezés: a képváltozás **lineáris** függvénye az elmozdulásnak translation. (Természetesen minden közelítő modell „rossz” valamennyire)

Kiinduló képminta x_t helyen

$$I_t + M\,\Delta x_t = I_{t+1}$$

(x_t) (x_t)

Új képminta x_t helyen

a“differencia kép” vagy másképpen:

“ mozgás template”

a keresett pozíció változás t és t+1 időközben

A közelítő feltételezés mellett $\Delta x_t = ?$

n x 1 mátrix

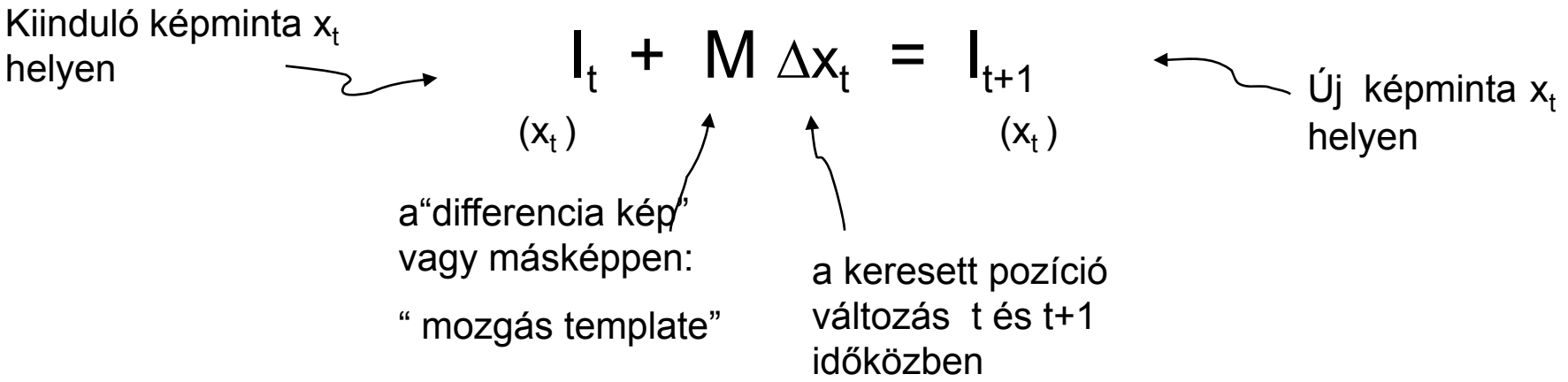
$$M\,\Delta x_t = (I_{t+1} - I_t)$$

n x 1 mátrix

(x_t-ben)

SSD követési algoritmus

Közelítő feltételezés: a képváltozás **lineáris** függvénye az elmozdulásnak translation. (Természetesen minden közelítő modell „rossz” valamennyire)



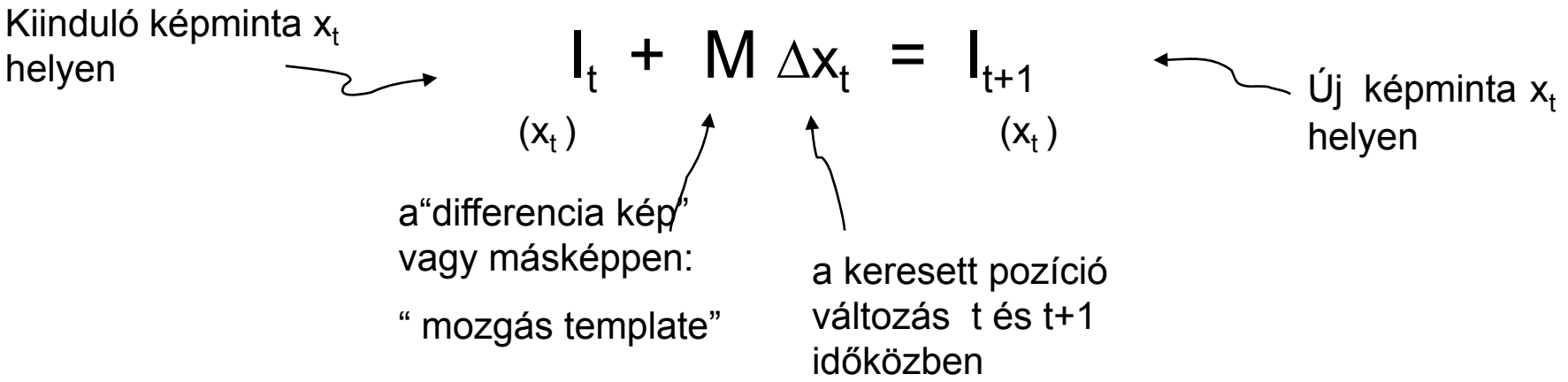
A közelítő feltételezés mellett $\Delta x_t = ?$

$$M\,\Delta x_t = (I_{t+1} - I_t)$$

$$M^T M\,\Delta x_t = M^T(I_{t+1} - I_t)$$

SSD követési algoritmus

Közelítő feltételezés: a képváltozás **lineáris** függvénye az elmozdulásnak translation. (Természetesen minden közelítő modell „rossz” valamennyire)



A közelítő feltételezés mellett $\Delta x_t = ?$

$$M \Delta x_t = (I_{t+1} - I_t)$$

$$M^T M \Delta x_t = M^T (I_{t+1} - I_t)$$

$$\Delta x_t = (M^T M)^{-1} M^T (I_{t+1} - I_t)$$

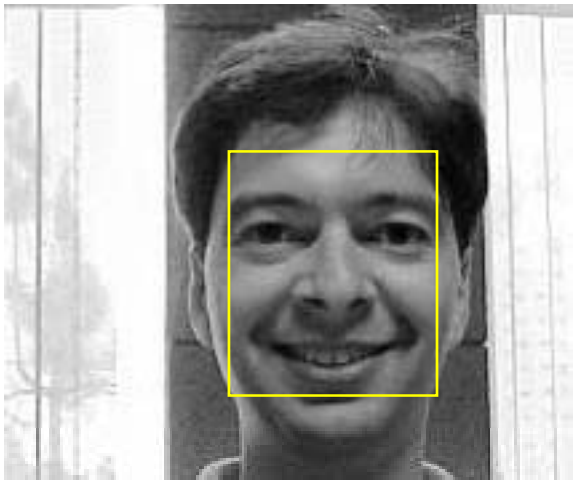
Ez maga a követési algoritmus...

SSD követési algoritmus

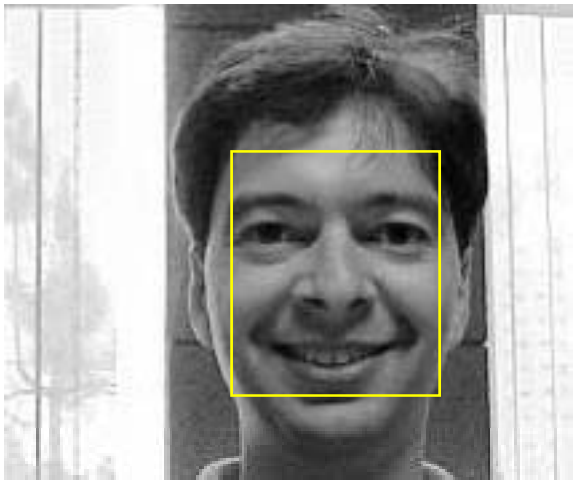
Mi történik a képen?

$$I_t + M\,\Delta x_t = I_{t+1}$$

(x_t–ben) (x_t-ben)



+ ? =

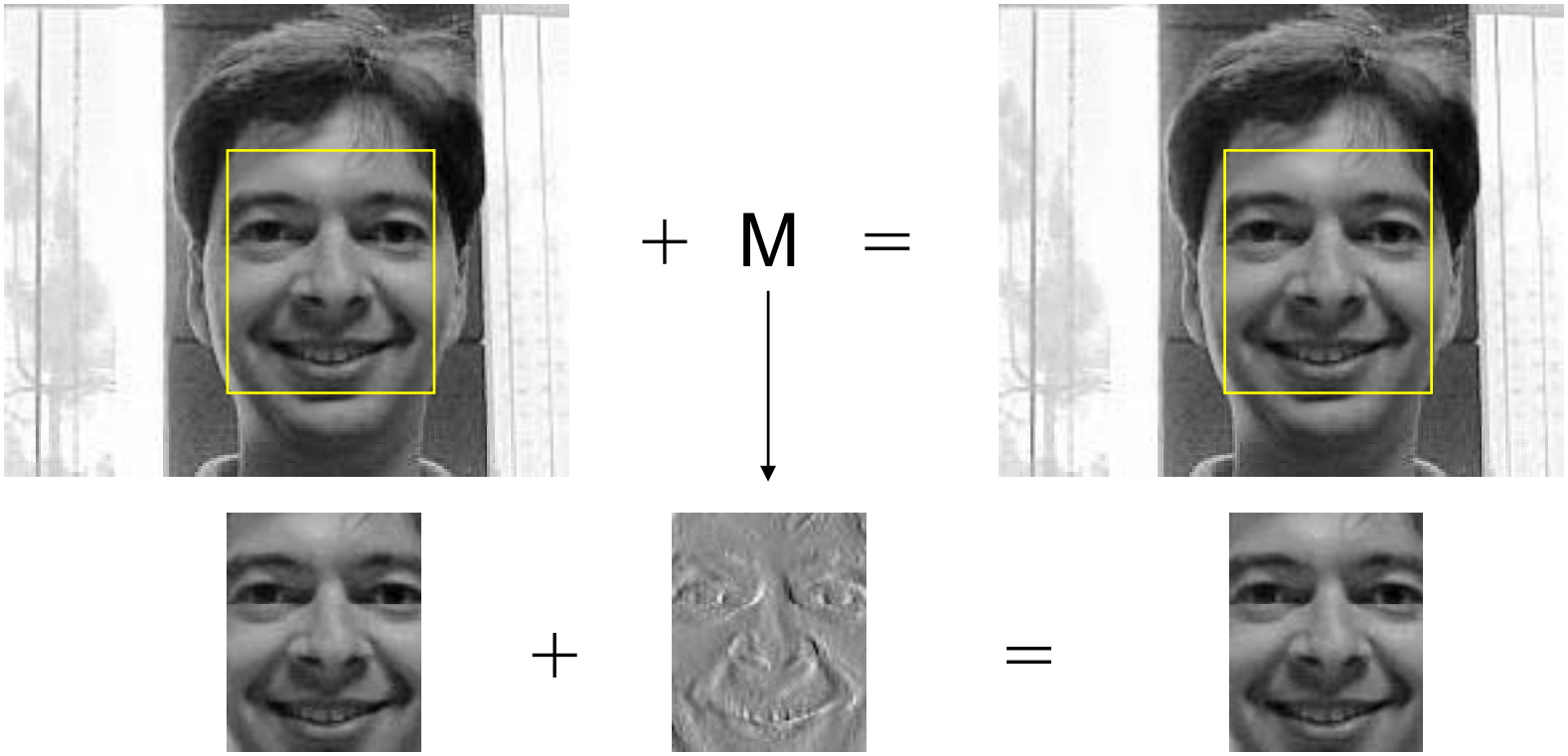


+ ? =

SSD követési algoritmus

M a képminta és elmozdult megfelelője közötti különbség

$$I_t(x_t) + M\,\Delta x_t = I_{t+1}(x_t)$$

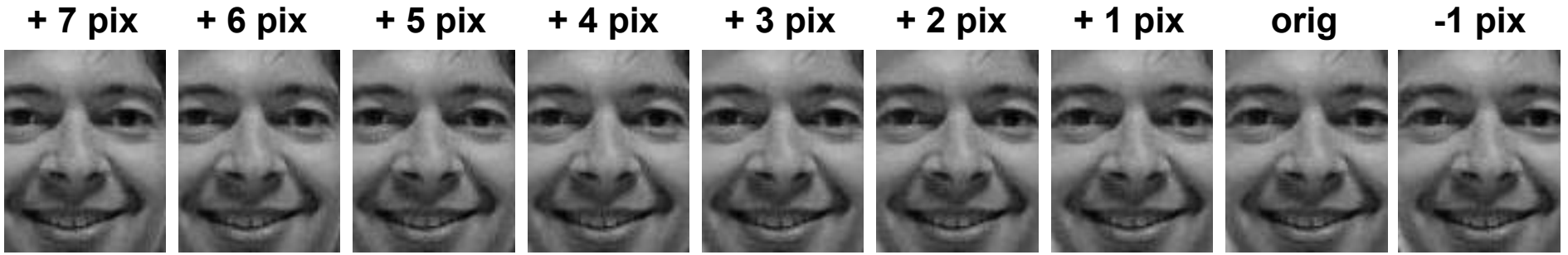


SSD követési algoritmus

A lineáris közelítés milyen nagy elmozdulásig igaz?

$$M\,\Delta x_t = I_{t+1} - I_t$$

Az eltolás mértéke ...



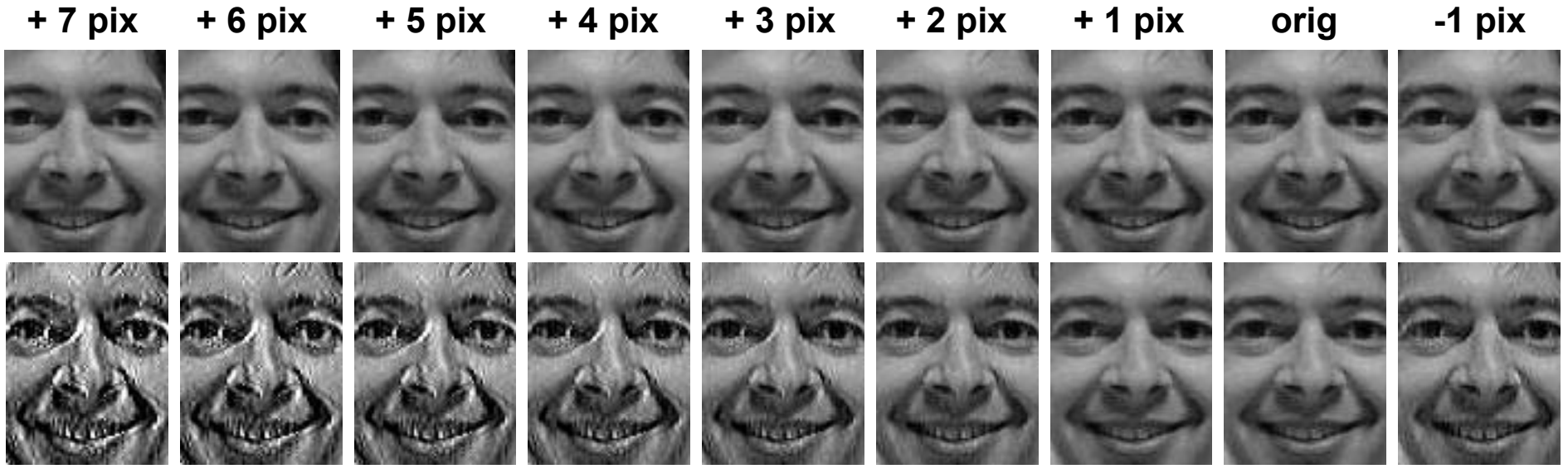
Eredeti kép + a template többszöröse...

SSD követési algoritmus

A lineáris közelítés milyen nagy elmozdulásig igaz?

$$M\,\Delta x_t = I_{t+1} - I_t$$

Az eltolás mértéke ...



Eredeti kép + a jobboldali template többszöröse...

Az eredeti kép felismerhető, ugyanakkor az elmozdulás korrigálva (vagyis követve)!

Követés transzlációk (etc.)

Több állapotváltozó is kezelhető egyszerre:

$$M \begin{bmatrix} \Delta x_t \\ \Delta y_t \end{bmatrix} = I_{t+1} - I_t$$

- x transzláció
- y transzláció

n x 2 matrix

eredeti képminta | x translation template | y translation template | eredeti képminta | x translation template | y translation template

M természetesen függ a követendő képmintától, de időben nem változik

Követés transzlációk

További paraméterekre is jó:

$$M \begin{pmatrix} \Delta x_t \\ \Delta y_t \\ \Delta r_t \\ \Delta s_t \end{pmatrix} = I_{t+1} - I_t$$

- x transzláció
- y transzláció
- rotáció
- skálázás

M n x 4 mátrix

eredeti | x | y | elforgatás | skálázás

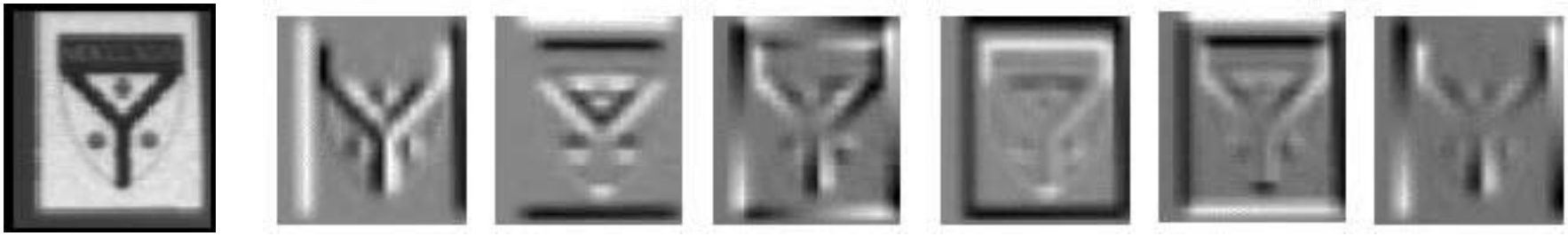
Követés transzlációk

Vagy elvben bármilyen további transzformáció figyelembevehető...

$$M \begin{pmatrix} \Delta x_t \\ \Delta y_t \\ \Delta r_t \\ \Delta s_t \\ \Delta a_t \\ \Delta h_t \end{pmatrix} = I_{t+1} - I_t$$

- x transzláció
- y transzláció
- rotáció
- skálázás
- x-y képarány
- nyírás (shear)

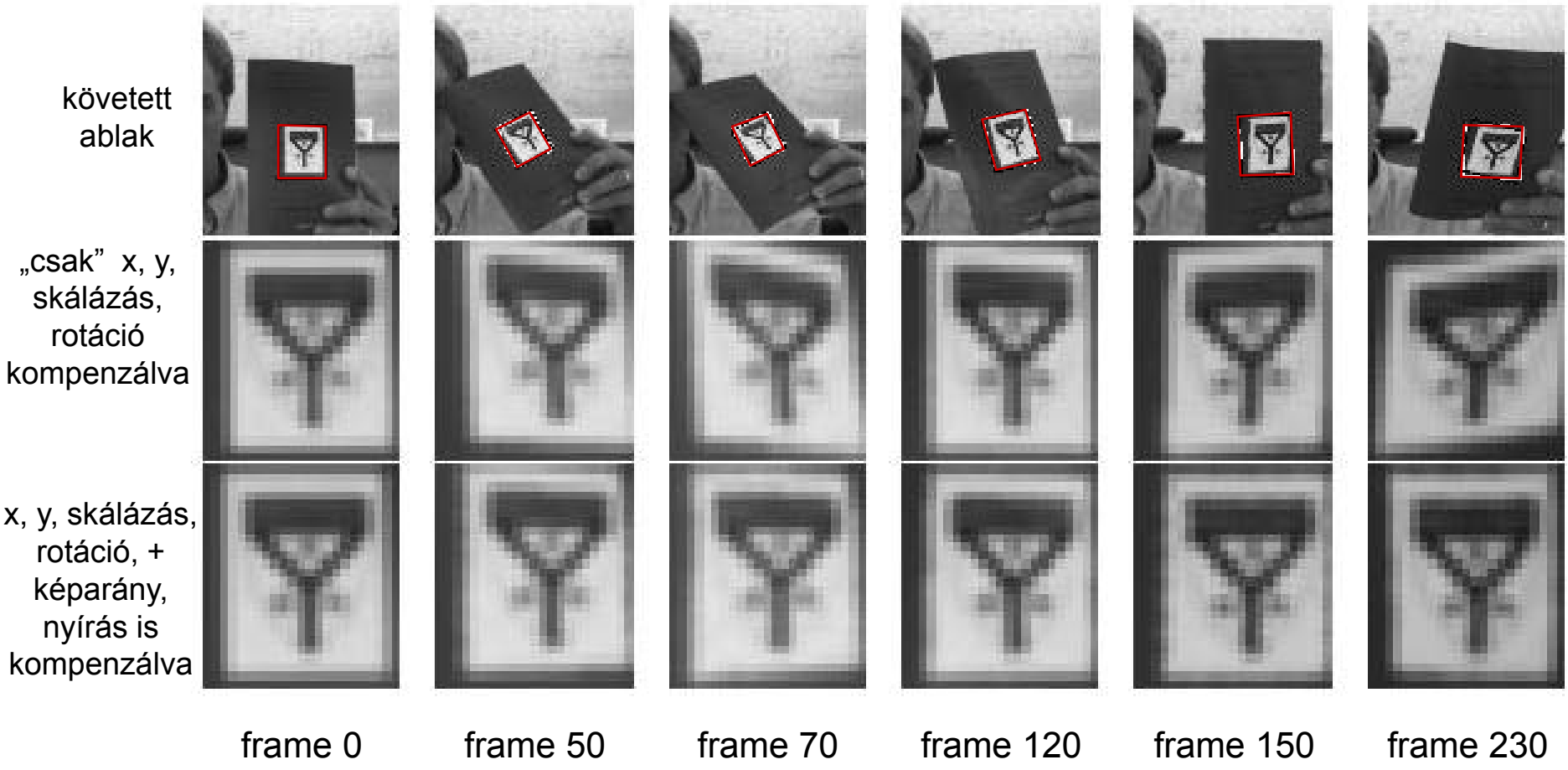
n x 6 mátrix



eredeti | x | y | elforgatás | skálázás | összenyomás | nyírás

Követés transzlációk

Az irodalomból átvett demo forrás futási eredményei szerint az alsó szekvencia már szépen kompenzál (másképpen fogalmazva: optimális illesztés esetén a mozgás paraméterek meghatározhatók)



Megvilágitás ingadozás kezelése

Akár (az egyre nagyobb dimenziójú) M-be zsúfolhatunk minden újabb kompenzált hatást:

$$M \, \vec{\Delta x}_t = I_{t+1} - I_t$$

n x p mátrix

p parameters

n	= minta pixelek száma
p	= paraméterek száma

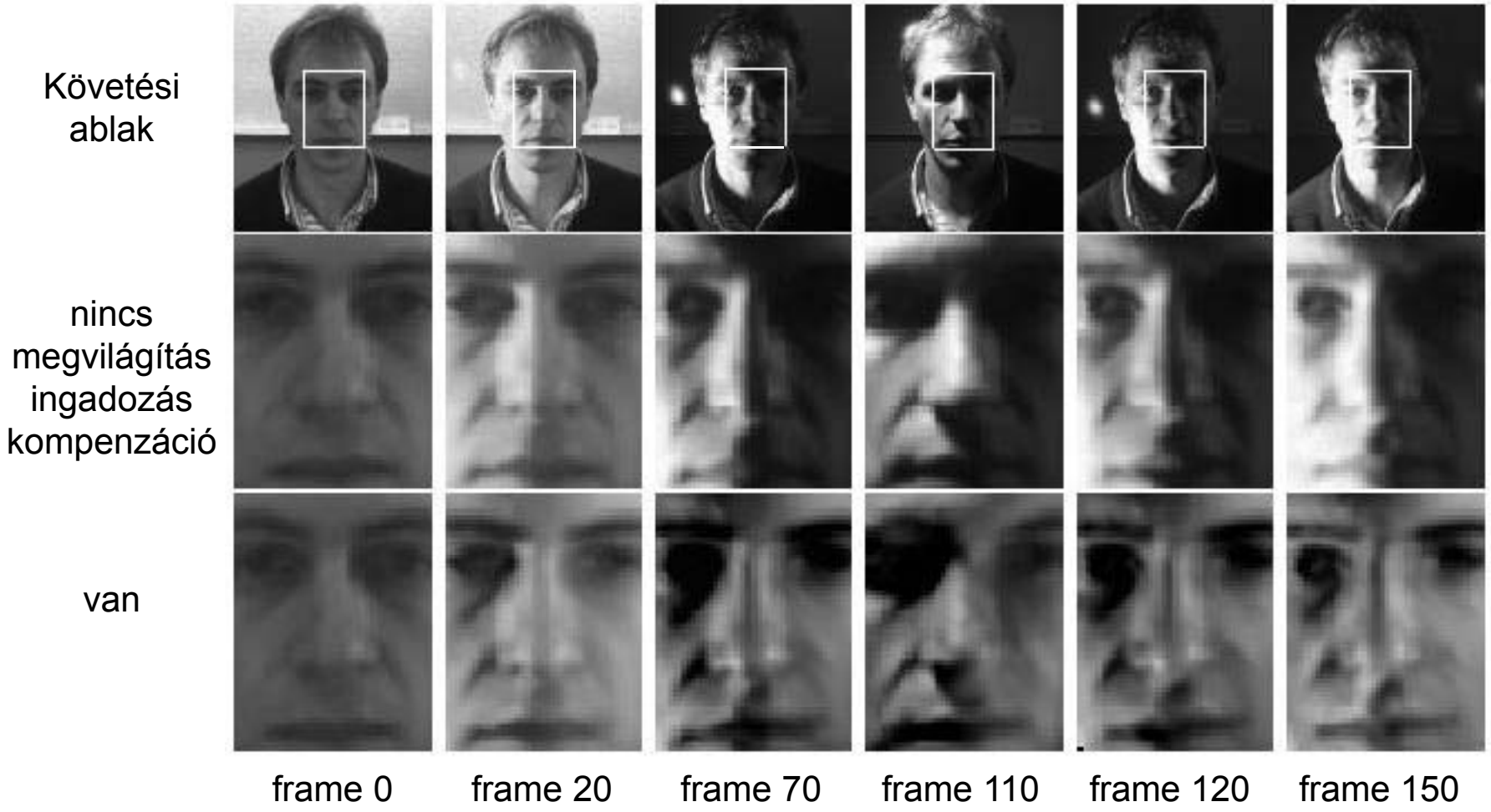
- x transzláció
- y transzláció
- rotáció
- skálázás
- x-y képarány
- nyírás
- megvilágítás változás

Például 6 további “mozgás” template is megadható:

fényerő kontraszt ← különböző megvilágítási irányok hatása →

Megvilágitás ingadozás kezelése

Ismét a demo futási eredményeket megjelenítve:



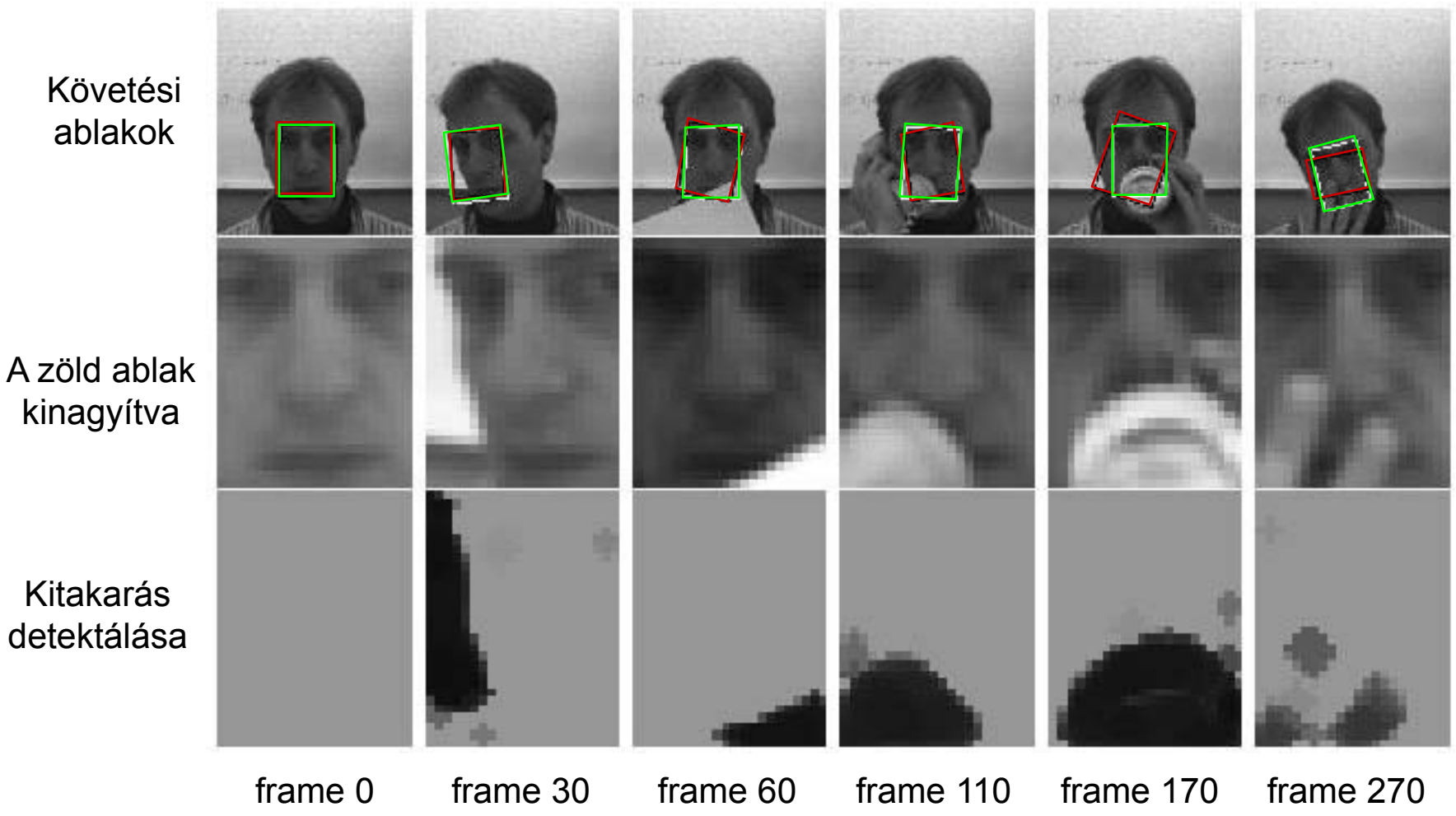
Részben takart objektumok követése

A megoldás kulcsa: nem minden pixelt veszünk (a követendő ablakon belül) figyelembe

- ha az összes „paraméter template” alkalmazásával is - bizonyos részeken - az eltérés mindig nagy maradna: ezeket a részeket hagyjuk figyelmen kívül (mert feltehetőleg kitakart részekről lehet szó).
- csak a maradék minta ablakot próbáljuk meg illeszteni (így egyben követni…)

Részben takart objektumok követése

Demo: a zöld ablak „figyeli” a kitakarást (a piros ablak nem)



Bináris képfeldolgozás
BLOB jellemzők

•Ha lehet, vezessük vissza a problémát bináris képek halmazának feldolgozására

Bináris képjellemzők meghatározása

valósidejű implementációja relatíve könnyű lesz

aztán érdemes lesz - amit csak lehet - erre visszavezetni (kiterjesztés gradált, színes képekre)

Képreprezentáció

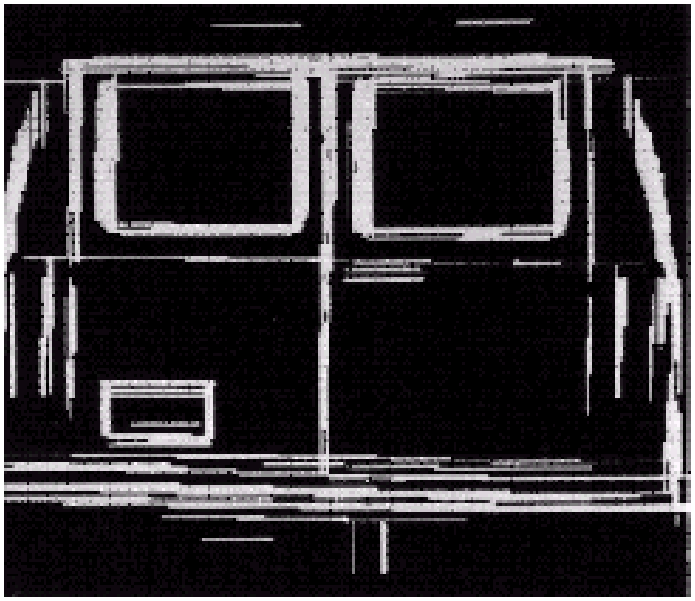
- „Belső” reprezentáció: régió-alapú
  - Amikor az intenzitás, szín, textúra, stb. a fontos
- „Külső” reprezentáció: régió-határ alapú
  - Amikor az alak, él, stb. a fontos

Gyakran mindkét típusú reprezentációt célszerű együtt alkalmazni

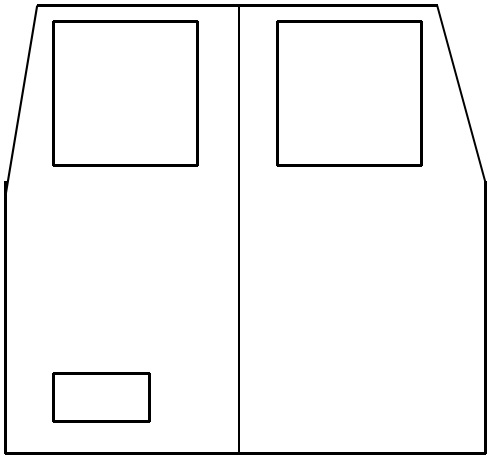
Képjellemzők „kívánatos” tulajdonságai

- Hatékony implementáció (valósidejűség!)
- Hatékony osztályozás (nagy a szétválasztó erő: mérhető különbség egyes objektum osztályokra)
- Robusztusság:
  - Méret invariancia
  - Transzláció invariancia
  - Rotáció invariancia
- …

Példa: „külső” reprezentáció jobb



Élpontok: ha zaj is van - „túl sok”



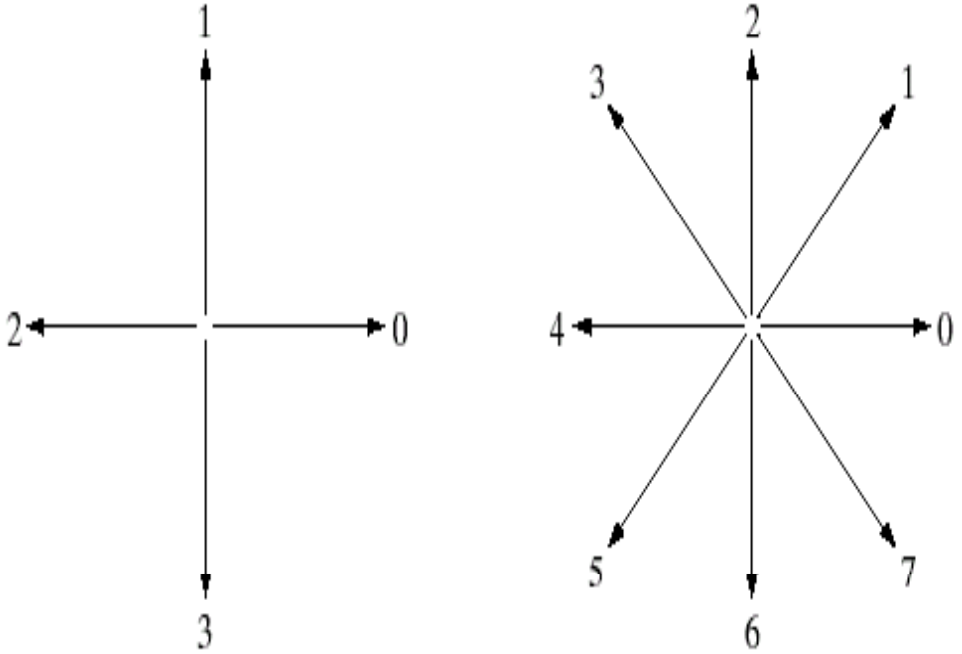
Kontúrok: „éppen elég lenne”

Lánckód

- Külső reprezentáció – a régió határokat írja le
- Közelítés egyenes szegmensekkel
- 4 vagy 8 szomszédos képmodell

a b

FIGURE 11.1 Direction numbers for (a) 4-directional chain code, and (b) 8-directional chain code.



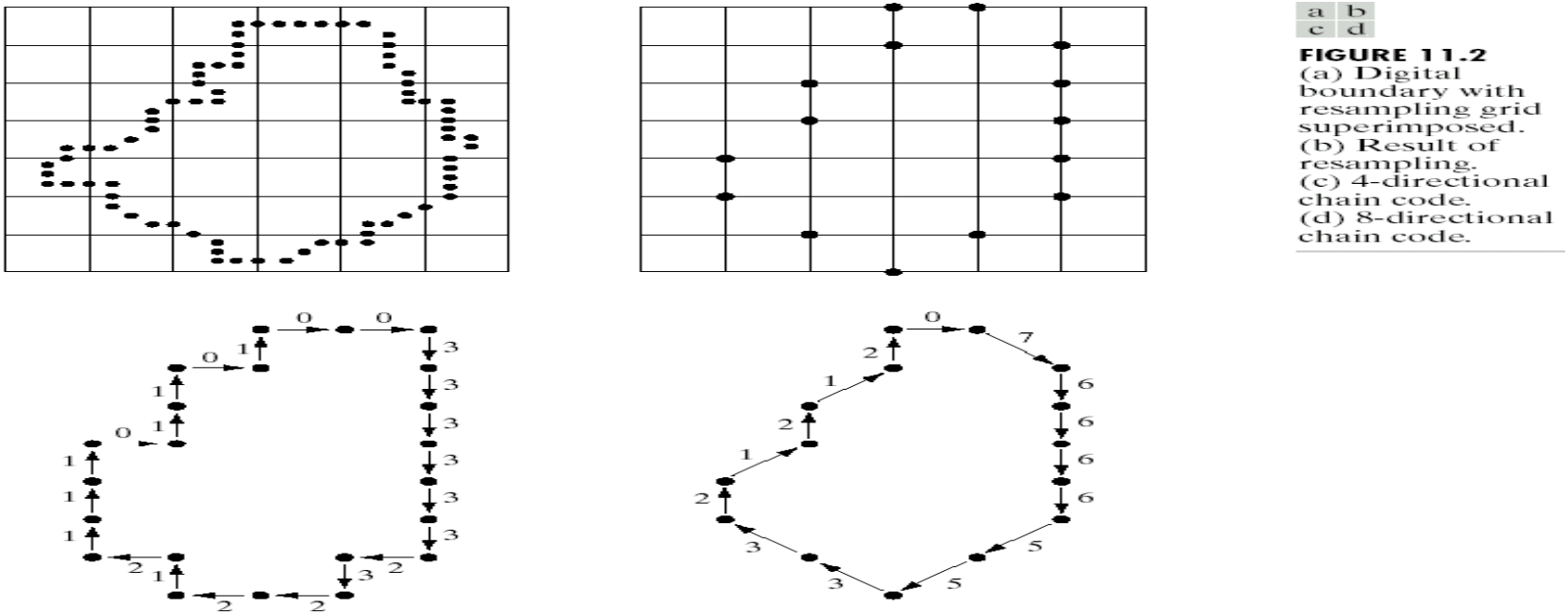
Lánckód – implementációk

Általában

- Kezdőpont megkeresése
- Óramutató járásával egyező vagy ellentétes irányban elfordulva a következő szomszédos pont megkeresése és összekötésük
  - Túl hosszú a lánckód
  - Már az élek kis változása (zaj!) eltérő lánckódhoz vezet

Gyorsabb implementáció

- Éldetektált kép durvább raszterű újra-mintavételezése
- Amelyik rasztert érinti az eredeti régió határ – megjelölve, összekötés



a b
c d

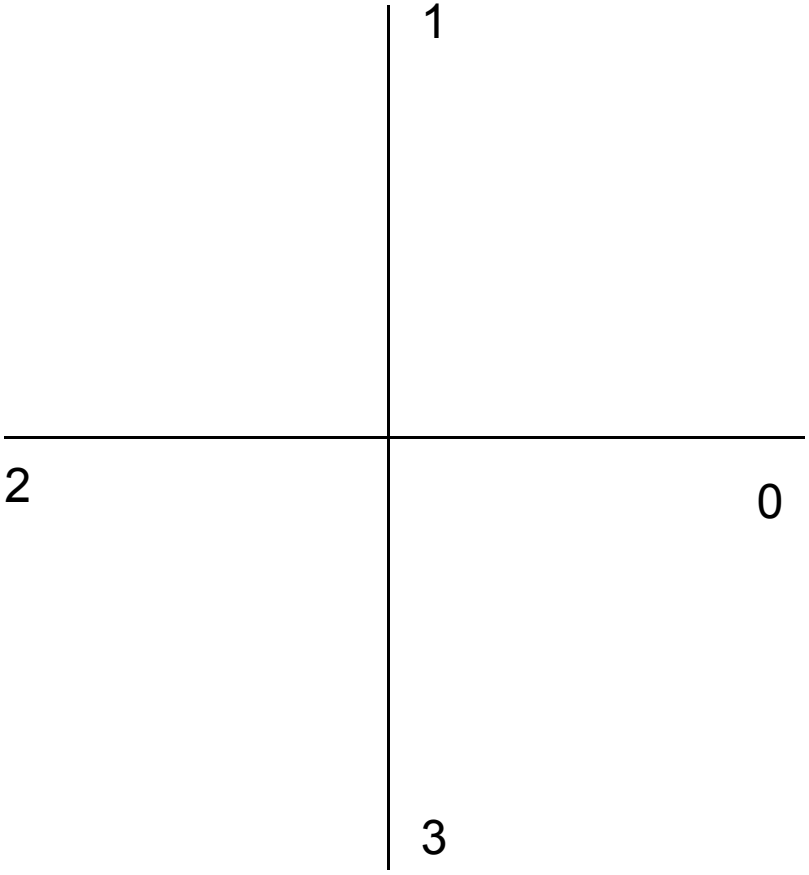
FIGURE 11.2
(a) Digital boundary with resampling grid superimposed. (b) Result of resampling. (c) 4-directional chain code. (d) 8-directional chain code.

Láncкód - tulajdonságok

- Érzékeny a kezdőpont választásra
  - Pl. minimális lánckód hosszúság szerinti kezdőpontválasztás
- Érzékeny rotációra, méretre egyaránt

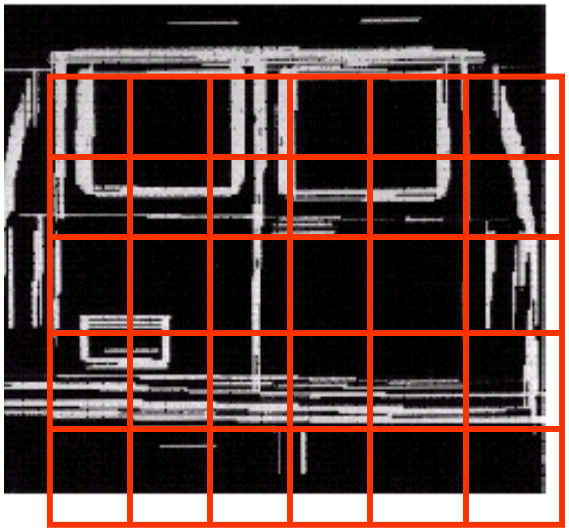
- Következő diákon látható egyszerű megoldások jók, feltéve, hogy: az éldetektáló algoritmus rotáció- és skálázás-független – legtöbbször nem ez az eset… akkor más megoldást kell keresnünk

Rotáció invariancia biztosítása

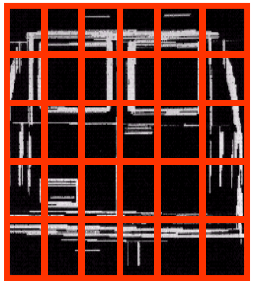


- Differencia képzés
- Például: eredeti 4-szomszédos lánckód: **10103322**
- Differencia képzés után: **3133030** (óramutató járásával ellentétes irány szerinti körbejárással)

Méret invariancia biztosítása



Nagyobb kép



Kisebb kép

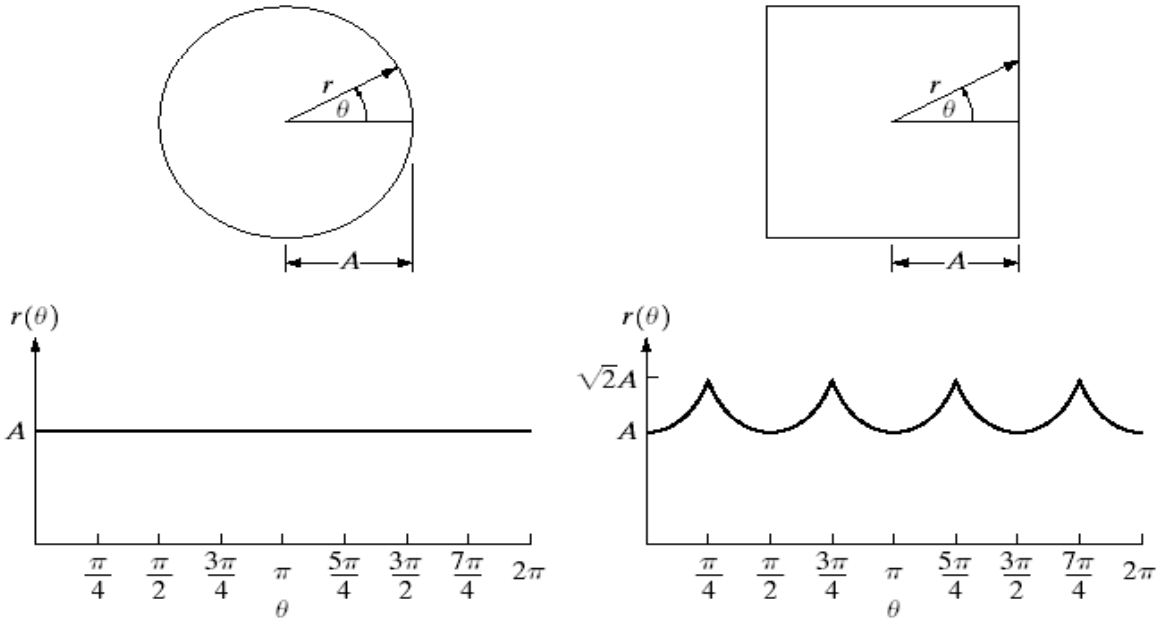
- Méret normálás: az újra-mintavételezés felbontását változtatjuk valamilyen normáló kritérium szerint

Lenyomat (angolul: signature)

- 1-dimenziós régió-határ mérték (sokfajta algoritmus létezik)
- Egy egyszerű implementáció: súlyponttól való távolság a szög függvényében:

a b

FIGURE 11.5 Distance-versus-angle signatures. In (a) $r(\theta)$ is constant. In (b), the signature consists of repetitions of the pattern $r(\theta) = A \sec\theta$ for $0 \le \theta \le \pi/4$ and $r(\theta) = A \csc\theta$ for $\pi/4 < \theta \le \pi/2$.



- Transzláció invariancia: alaphelyzetben is OK
- Rotáció invariancia: pl. legyen a súlyponttól legtávolabbi pont a kezdőpont
- Skálázás invariancia: pl. távolság normálás - a legnagyobb távolsággal osztva

További régió-határ mértékek

- **régió-határ hossz** (pl. lánckódból becsülhető: horizontális elemek száma + vertikális elemek száma + diagonális elemek száma x √2
- **átmérő**
  - Max. tengely (extrémum pontok távolsága , Feret átmérő)
  - Min. tengely (max. tengelyre merőleges irányban a max. szegmens hossz)
  - Excentricitás (max. / min tengelyek hányadosa)
- **orientáció**
- **görbület**
- **…**

Régió mértékek

- Régió **terület**
- Kompaktság mérték = **kerület^2/terület**
  - rotációra, skálázásra, transzlációra invariáns
- **Régió intenzitás** átlag / max. / min. érték
- Átlagérték alatti / feletti pixelek aránya
- …

Régió topológiai mértékek – Euler szám

A topológiai jellemzők a régiók szétszakítását / összekötését nem okozó deformációkra érzéketlenek

Legismertebb közülük az Euler szám (különböző definíciók /reprezentációk lehetségesek)

- V-Q+F=**C-H=E**
- V= sarkok száma
- Q= élek száma
- F= poligonok száma
- **C= egybefüggő régiók száma**
- **H= lyukak száma**

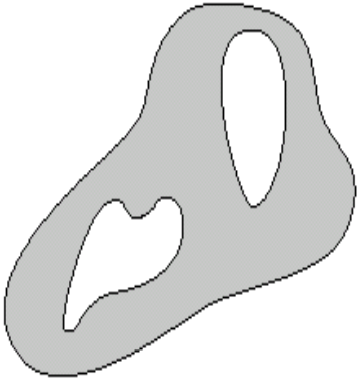


FIGURE 11.17 A region with two holes.

E=C-H=1-2=-1

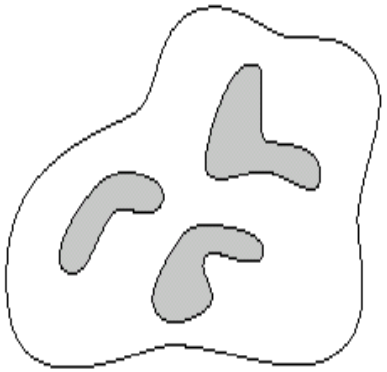
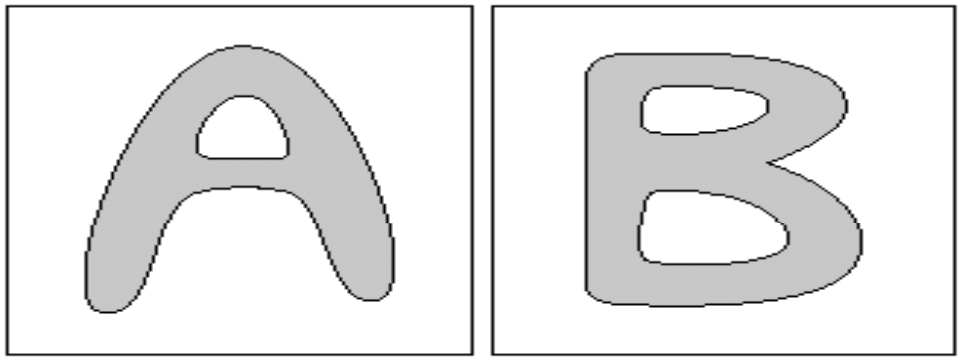


FIGURE 11.18 A region with three connected components.

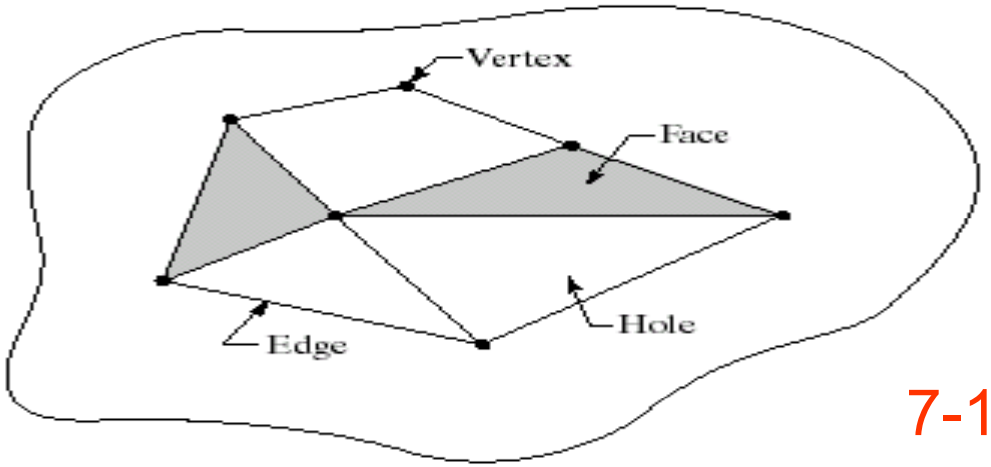
E=C-H=3-1=2

Régió topológiai mértékek – Euler szám



a b

FIGURE 11.19 Regions with Euler number equal to 0 and −1, respectively.



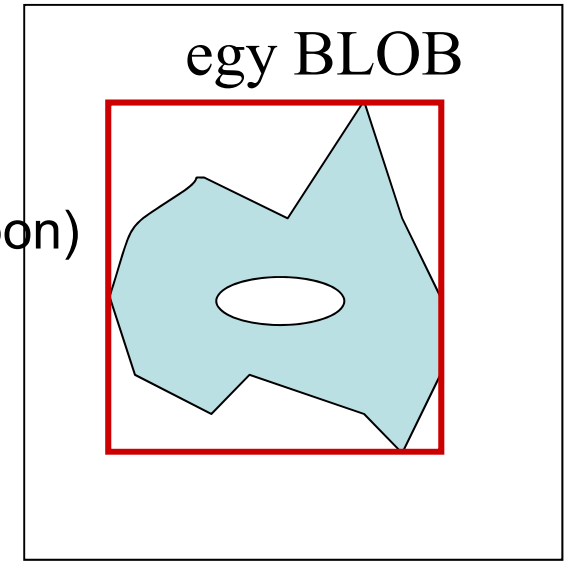
V-Q+F=C-H=E
V= sarkok száma
Q= élek száma
F= poligonok száma
C= egybefüggő régiók száma
H= lyukak száma
E= Euler szám

7-11+2=-2

FIGURE 11.20 A region containing a polygonal network.

BLOB jellemzőkre példák

- terület (pixelek száma)
  - Zajt el kell távolítani(pl. kicsi/nagy objektum alapon)
- Lyukak száma
- Objektum területe
- Lyukak területe
- Teljes terület (lyuk + objektum)
- Kerület = kontúr hossza
- Befoglaló keret
  - Bal felső sarok pozíciója
  - Befoglaló keret szélessége / magassága

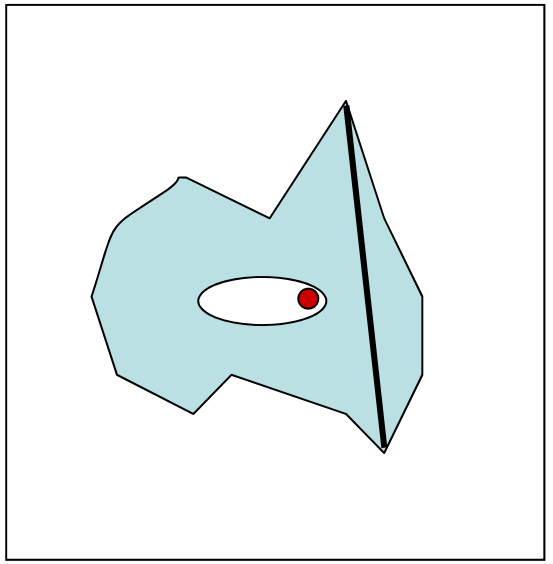


Szómagyarázat: BLOB = Binary Large OBject

BLOB jellemzőkre példák

- Súlypont (x_m,y_m)

$$x_m = \sum_{i \in object} x_i \qquad y_m = \sum_{i \in object} y_i$$



- „Kompaktság”
  - Diszkre minimális $\dfrac{Perimeter^2}{Area}$
  - Téglalapra minimális $\dfrac{Area}{Width \cdot Height}$
- körkörösség $\dfrac{Perimeter}{2\sqrt{\pi \cdot area}}$
- Legnagyobb távolság (Feret átmérő)
- Orientáció
  - Feret átmérő orientációja

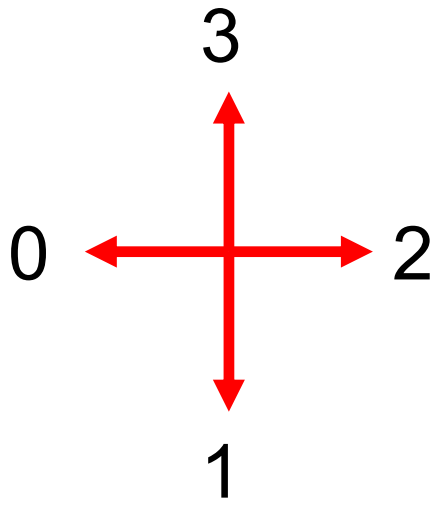
BLOB jellemzőkre példák

- Alakleírás
  - Befoglaló keret aránya: magasság/szélesség
    - A nyújtottságról mond valamit
  - Objektum illeszkedése befoglaló téglalapba
    - Köze van a terület / kerület arányhoz
  - Objektum illeszkedése befoglaló ellipszisbe
    - Köze van a terület / kerület arányhoz

Lehetünk kreatívak: egy csomó egyéb BLOB jellemző található még ki!
A BLOBOK HOGYAN HATÁROZHATÓK MEG EGYSZERŰEN / GYORSAN?

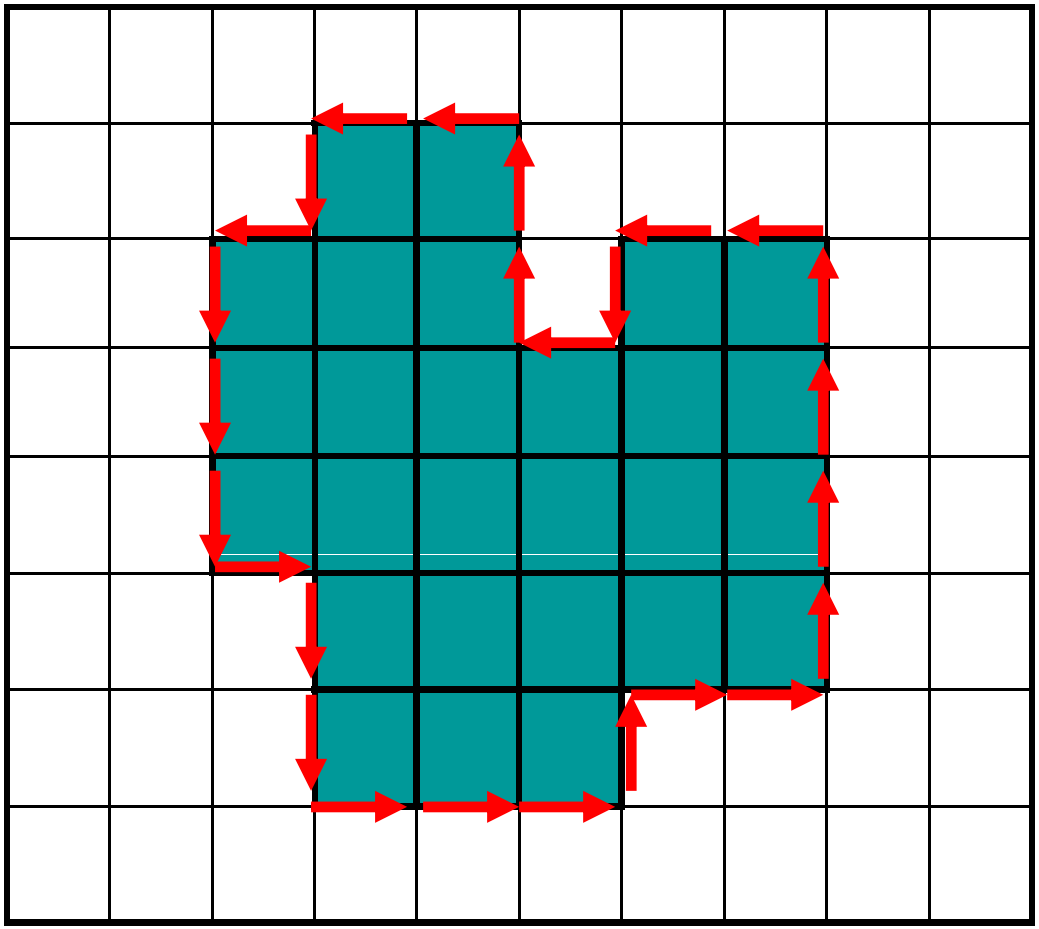
LÁNCKÓD számítás-hatékony implementációja ablakműveletekkel

LÁNCKÓD – 4 szomszéd reprezentációja

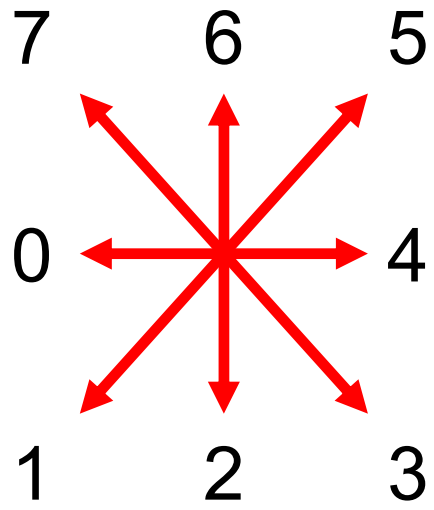


kontúr lánckódja:

10111211222322333300
103300

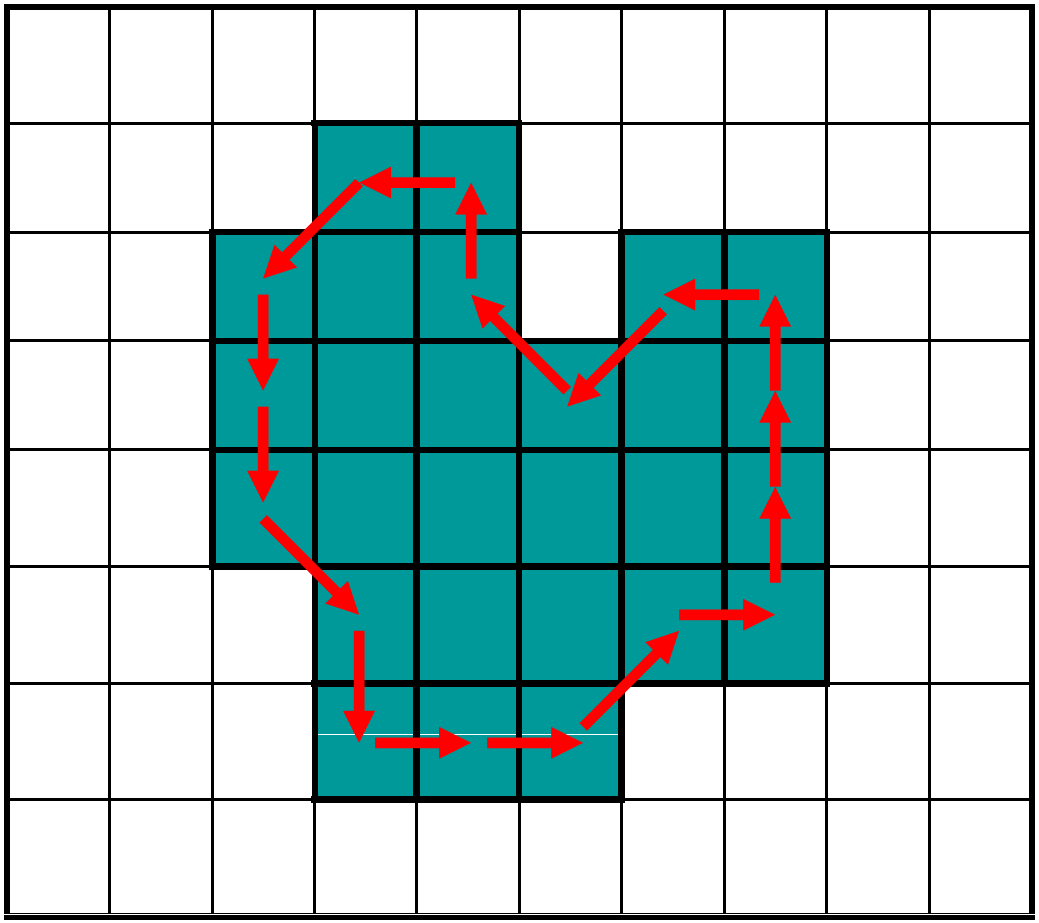


LÁNCKÓD – 8 szomszéd reprezentációja



kontúr lánckódja:

12232445466601760



4 sz. LÁNCKÓD implementáció 2x2 ablakművelettel

legyen **P** egy objektumpont

legyen **Q** egy 4-szomszédos háttérpont

Q és **P**, relatív pozíciójától függően címkézzük a 2x2 – es ablakon belül a maradék 2 pixelt: (**U** és **V)** az alábbi módon:

CODE 0

V	Q
U	P

CODE 1

Q	P
V	U

CODE 2

P	U
Q	V

CODE 3

U	V
P	Q

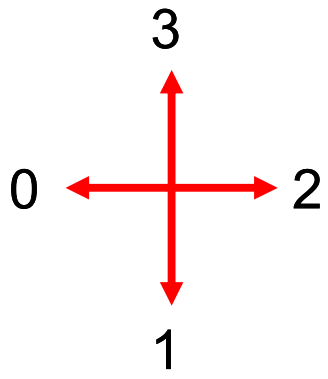
4 sz. LÁNCKÓD implementáció 2x2 ablakművelettel

- Legyen az objektumpontok értéke “1” , a háttérpontok értéke “0”
- **U** és **V** értéke határozza meg, hogy merre forduljon a következő lépésben az algoritmus:

U	**V**	**P’**	**Q’**	**fordulás**	**kód***
X	1	V	Q	jobbra	CODE-1
1	0	U	V	iránytartás	CODE
0	0	P	U	balra	CODE+1

*modulo 4 számlálóval implementálható

4-sz. LÁNCKÓD implementáció 2x2 ablakművelettel



CODE 0	
V	Q
U	P

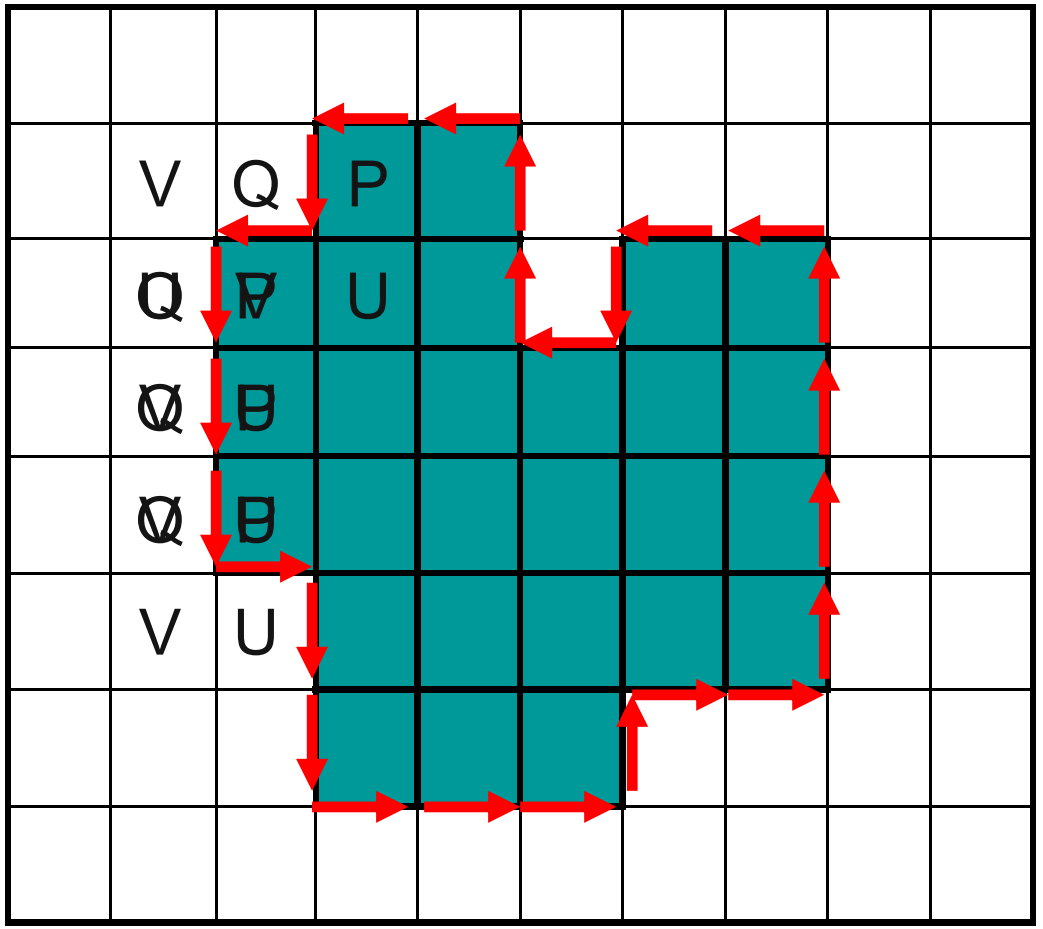
CODE 1	
Q	P
V	U

CODE 2	
P	U
Q	V

CODE 3	
U	V
P	Q

U	V	P'	Q'	TURN	CODE*
X	1	V	Q	RIGHT	CODE-1
1	0	U	V	NONE	CODE
0	0	P	U	LEFT	CODE+1

*Implement as a modulo 4 counter



8-sz. LÁNCKÓD implementáció 3x3 ablakművelettel

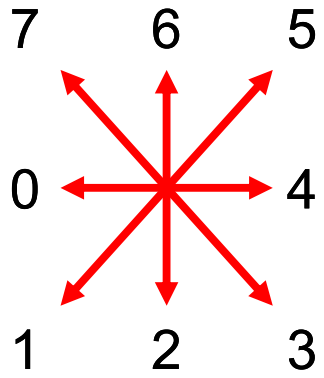
•Legyen **P** egy objektumpont és $\mathbf{R_0}$ egy 4-szomszédú háttérpont.

•Legyen az objektumpontokhoz “1”, a háttérpontokhoz “0” rendelve.

•Rendeljük **P** 8-szomszédos környezetéhez $\mathbf{R_0}$, $\mathbf{R_1}$, …, **R7** címkéket az alábbi sorrendben:

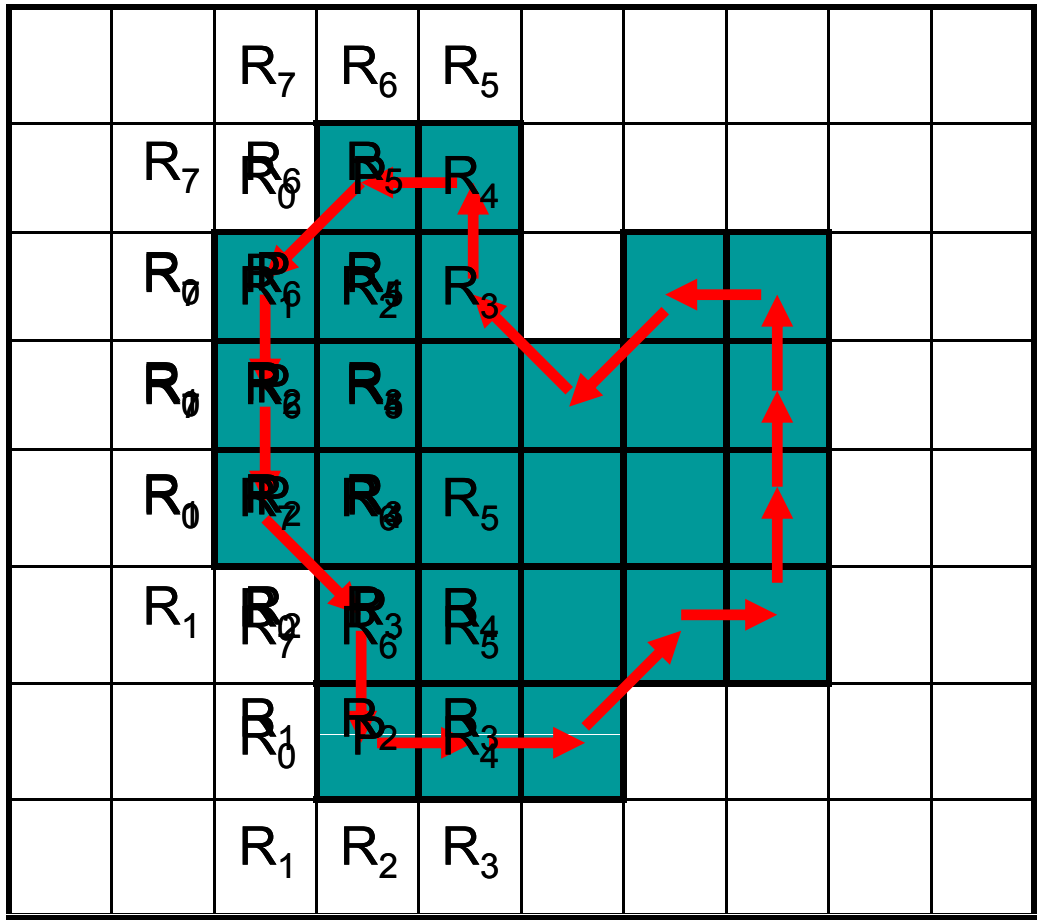
R_7	R_6	R_5
R_0	P	R_4
R_1	R_2	R_3

8-sz. LÁNCKÓD implementáció 3x3 ablakművelettel



ALGORITMUS:

i=0
WHILE (R_i==0) { i++ }
MOVE P to R_i
SET i=0
JUMP to next search



INVARIÁNS LÁNCKÓD számítási algoritmusok

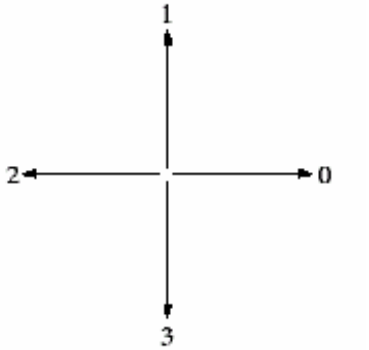
A LÁNCKÓD problémái (és a megoldások)

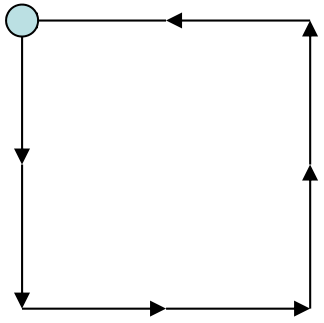
- Kezdőpont függő
- Orientáció függő

Alakfelismeréshez biztosítani kellene a lánckód invarianciáját

- Normalizált kódolás
- Differenciális kódolás

Normalizálás





33001122

33001122
30011223
00112233
01122330
11223300
12233001
22330011
23300112

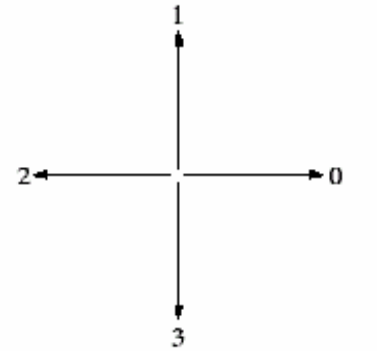
Sorba rendezve →

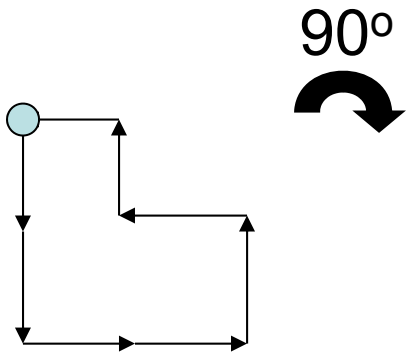
00112233
01122330
11223300
12233001
22330011
23300112
33001122
30011223

→ Az első sor lesz a normalizált kód

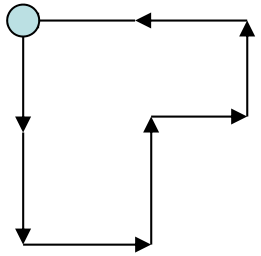
00112233

Differenciális lánckód





90°



33001212

↓normalizálás

00121233

33010122

↓normalizálás

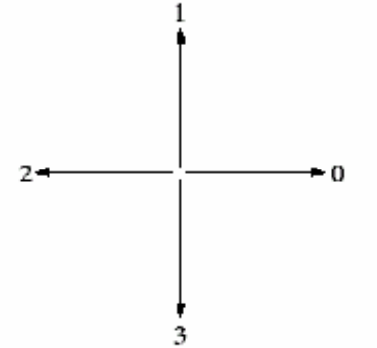
01012233

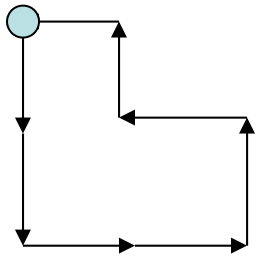
Differencális kódolás:

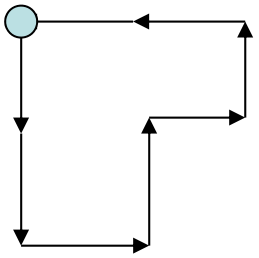
$d_k=c_k-c_{k-1}$ (modulo 4) 4-szomszédú lánckód esetén

$d_k=c_k-c_{k-1}$ (modulo 8) 8-szomszédú lánckód esetén

Invariáns alakleírás: Normalizált Differenciális Lánckód







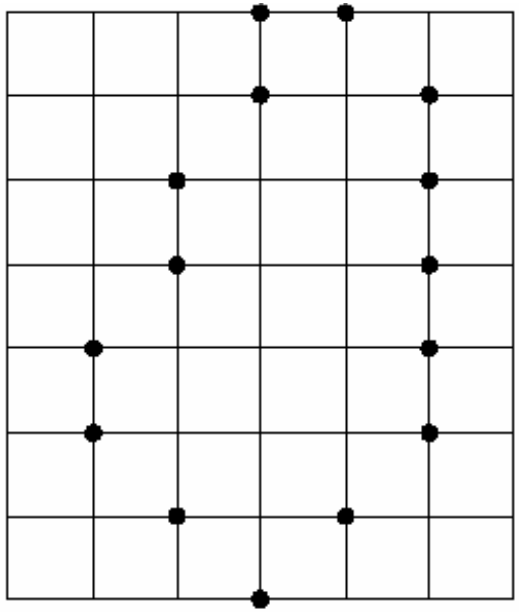
Differenciális kód:
$d_k = c_k - c_{k-1} \pmod 4$

33001212
↓ differenciálás
10101131
↓ normalizálás
01011311

33010122
↓ differenciálás
10113110
↓ normalizálás
01011311

A kezdőpont eltolás és a tényleges elforgatás (90°-al) dacára a két alakleírás ugyanazt a kódot adja

Egy másik stratégia az invariáns alakleírásra: Fourier leírók



{x(k),y(k)}, k=1,2,..,N

z(k)=x(k)+j y(k)

↓ DFT

$$a(n) = \frac{1}{N}\sum_{k=1}^{N} z(k)e^{-j2\pi nk/N}$$

↓ Fourier együtthatók

↓ IDFT

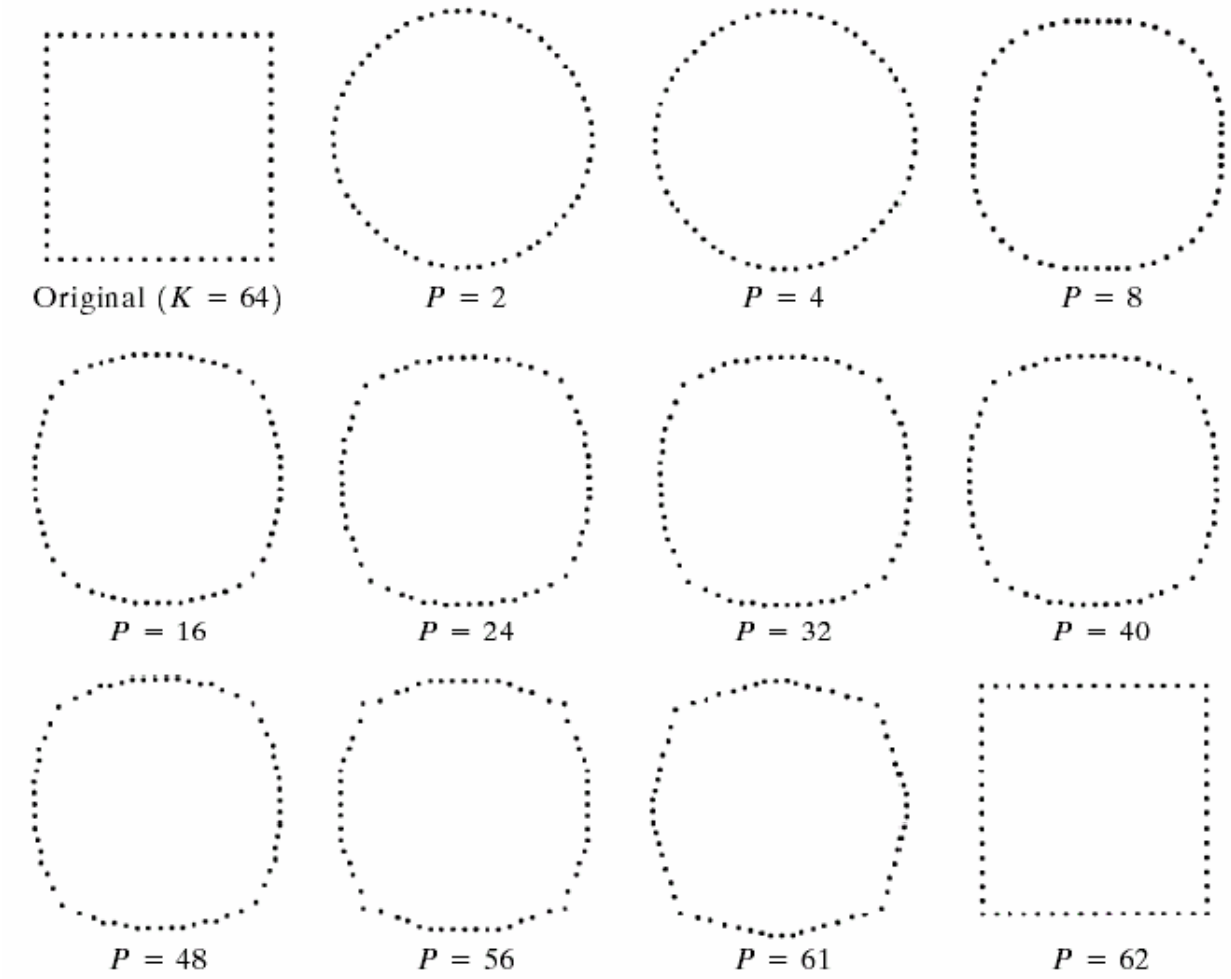
$$z(k) = \frac{1}{N}\sum_{n=1}^{N} a(n)e^{j2\pi nk/N}$$

Itt csak röviden utalni az előnyökre!

- Csak1 dimenziós (kontúr menti) Fourier transzformáció
- Együtthatók jelentése (1D és 2 D transzformáció esetében eltérő a szemléltetés)
- Eltolás, elforgatás (kezdőpont eltolás ill. tényleges elforgatás), skálázás hatása
- Fourier együtthatókból képezhetők ugyanúgy INVARIÁNS ALAKEGYÜTTHATÓK

Például:

$$\hat{z}(k) = \frac{1}{N}\sum_{n=1}^{P} a(n)e^{j2\pi mk/N}$$



P : a visszaállításhoz figyelembevett együtthatók száma