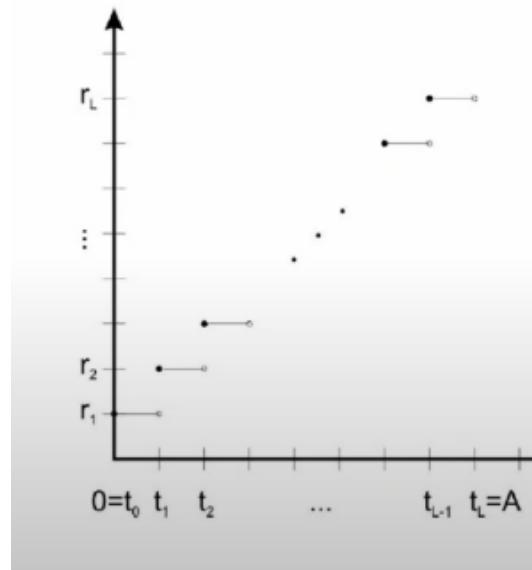


DIGI VIZSGA

HA MOST SE LESZ MEG SOSE LESZ MEG!

Kvantálás: Bizonyos intenzitás tartományokhoz más diszkrét intenzitás tartozik. Lépcsős függvény.

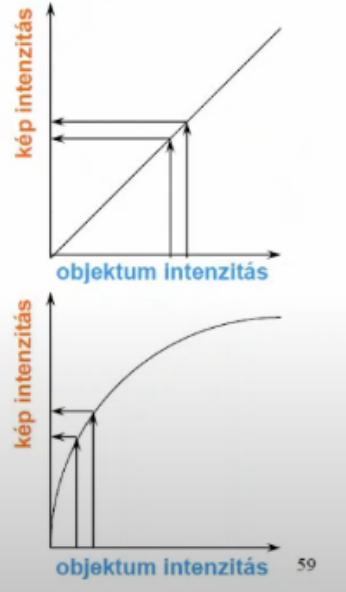


Uniform/lineáris kvantálás: Ha a kvantálás függvénye egyforma lépcsőfokokat tartalmaz.

Nemlineáris: Ha változik az intenzitás lépték, pl logaritmikus, a sötéteknél nagyobb az intenzitás lépték.

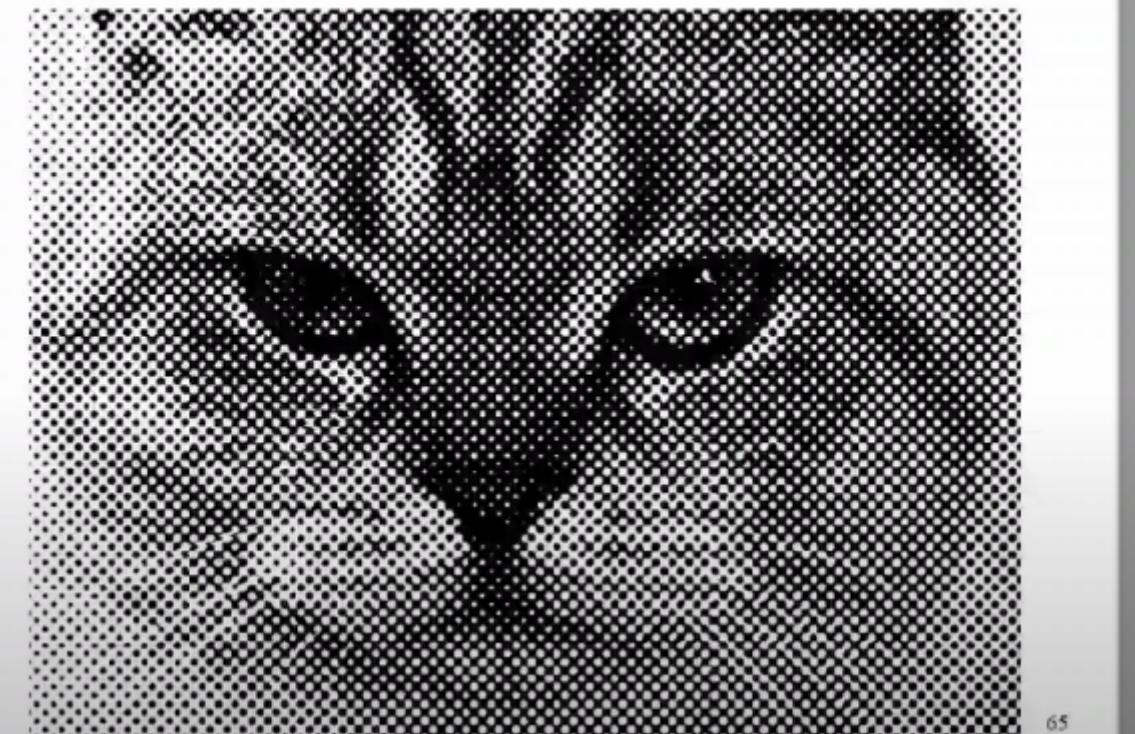
Kvantálási módok

- **Uniform/lineáris:**
a folytonos kép intenzitásait
lineárisan képezzük le
- **Nemlineáris (pl. logaritmikus):**
nagyobb intenzitásbeli lépték
a sötét területeken



Halftoning: Kvantálás bináris képekre, festékpöttyökből összeált kép

Halftone kép

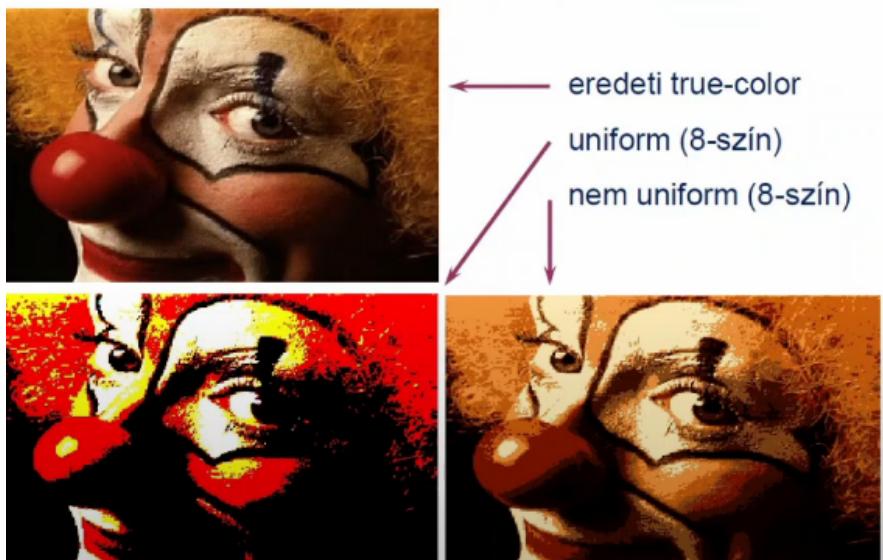


Színes képek kvantálása: Egy adott régióba tartozó színeknek ugyan azt a színt társítjuk, így a colormap csökken.

Uniform színes kvantálás: Egyforma méretű szeletekre bontjuk az eredeti színteret.

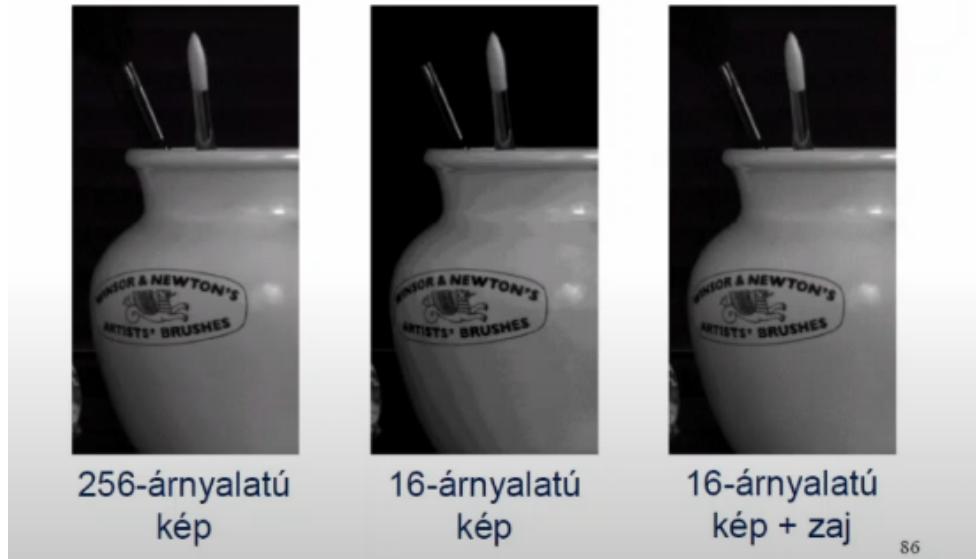
Nem uniform: Az adott képen szereplő színek eloszlásától függ. Amelyik színből több van, az több árnyalatot kap a palettán.

Uniform ↔ nem uniform



Dithering: Mesterséges zaj hozzáadása a képhez, így a kvantálási hiba csökkenthető.

Kvantálási hiba csökkentése



86

Lineáris transzformáció: Odafele kitaláljuk, milyen alapanyagokat tartalmaz a koktélünk és mennyit az egyes elemekből. Vissza fele pedig kikeverjük az eredeti koktélpontos mását. 2D is ugyan ugy működik csak $M \times N$ -es mátrixokkal dolgozunk.

$$\rightarrow F(X) = \sum_{x=0}^{M-1} f(x) \cdot w(x, X)$$

súlyok

bázis-függvények

$$\leftarrow f(x) = \sum_{X=0}^{M-1} F(X) \cdot w'(x, X)$$

Fourier-transzformáció: Lineáris transzformációt használ. A bázis függvények ekkor m-edik egységgökök, azaz küllők. Ha egységgököket hatványozunk egységgököket kapunk.

1D Fourier-transzformáció

$$\xrightarrow{\text{DFT}} F(X) = \sum_{x=0}^{M-1} f(x) \cdot w(x, X) \quad (X = 0, 1, \dots, M-1)$$

$$w(x, X) = \frac{1}{M} e^{-2\pi i \frac{xX}{M}}$$

$$\xleftarrow{\text{IDFT}} f(x) = \sum_{X=0}^{M-1} F(X) \cdot w'(x, X) \quad (x = 0, 1, \dots, M-1)$$

$$w'(x, X) = e^{2\pi i \frac{xX}{M}}$$

Fourier-transzformáció mátrix vektor szorzásként is felírható: Műveletigénye M^2 .

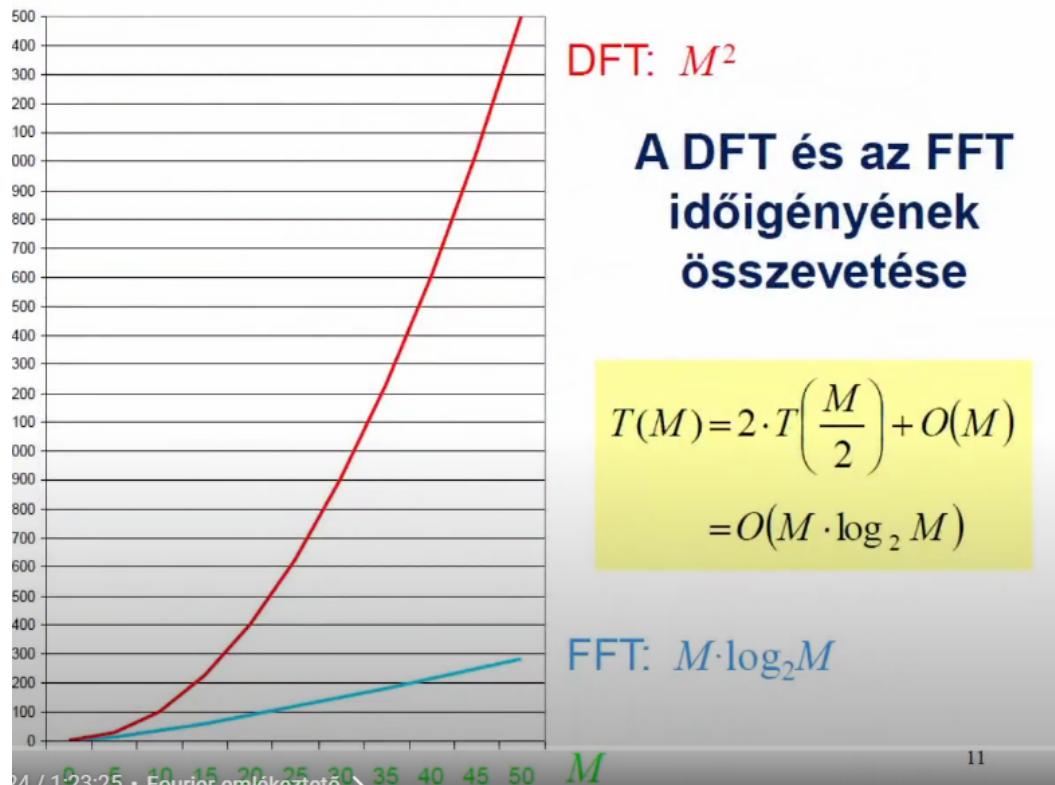
Az 1D DFT műveletigénye

$$\frac{1}{M} \begin{bmatrix} \omega_M^0 & \omega_M^0 & \omega_M^0 & \cdots & \omega_M^0 \\ \omega_M^0 & \omega_M^1 & \omega_M^2 & \cdots & \omega_M^{M-1} \\ \omega_M^0 & \omega_M^2 & \omega_M^4 & \cdots & \omega_M^{2(M-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \omega_M^0 & \omega_M^{M-1} & \omega_M^{2(M-1)} & \cdots & \omega_M^{(M-1)(M-1)} \end{bmatrix} \cdot \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ f(M-1) \end{bmatrix} = \begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ \vdots \\ F(M-1) \end{bmatrix}$$

$M \times M \qquad \qquad \qquad M \times 1 \qquad \qquad \qquad M \times 1$

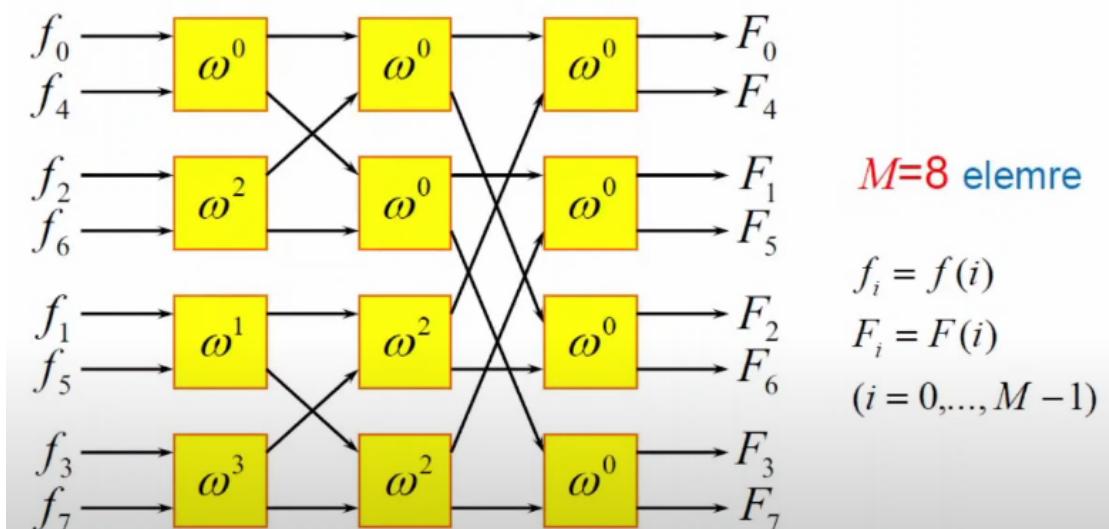
$M \cdot M = M^2$ szorzás

FT/fast fourier trasform: Gyorsabb fourier transzformáció. A fésűrendezés elvén működik, kihasználja hogy a fourier transzformáció mátrixa egy MxM-es mátrix és csak M féle elemet tartalmaz. Időigény $O(M \cdot \log_2 M)$



Párhuzamos FFT: Gyorsabb mint a sima FFT, időigénye $O(\log_2 M)$.

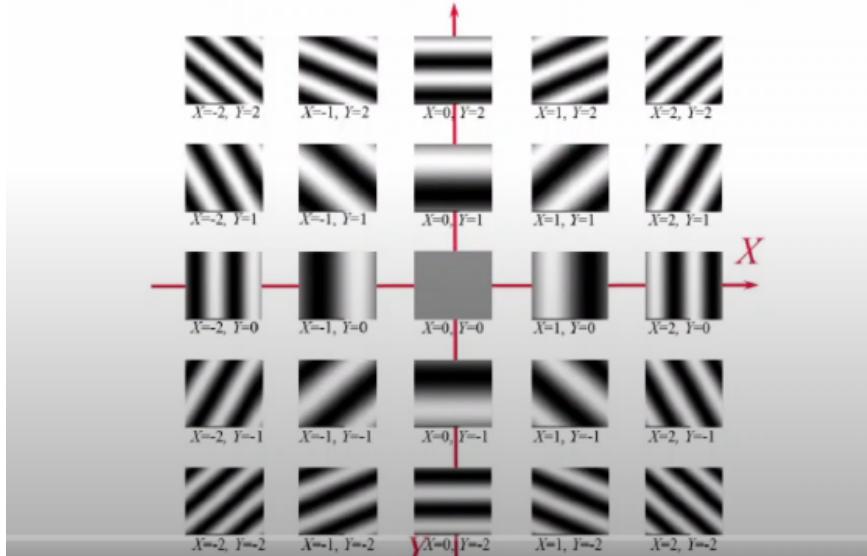
Párhuzamos FFT hálózat



Bázis kép: Összes eleme egy komplex szám. Ha a bázisképeket origóra rendezzük akkor inál távolabb vagyunk az origótól annál nagyobb a frekvencia. Bármilyen kép kikeverhető bázisképek és fourier transzformációk segítségével.

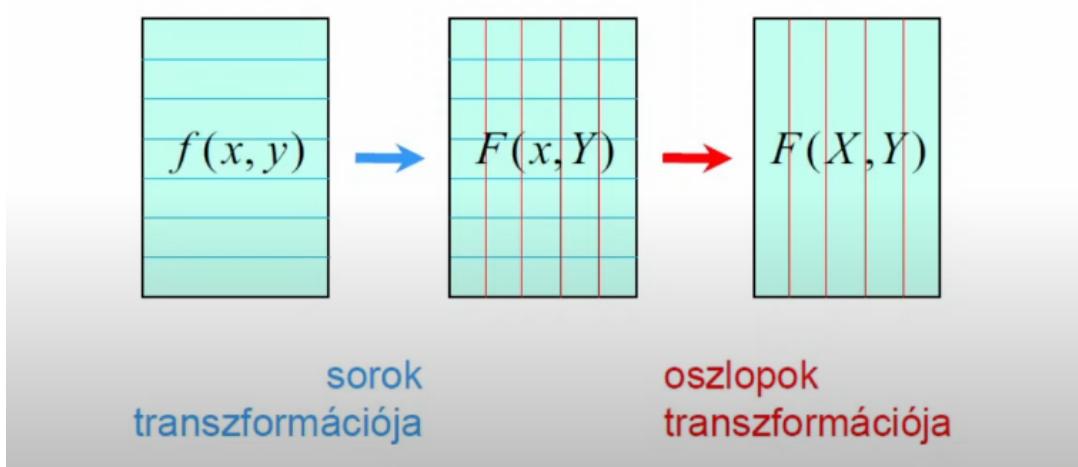
2D IDFT bázisképei

$$\operatorname{Re}\left(e^{2\pi j \left(\frac{xX}{M} + \frac{yY}{N}\right)}\right) = \cos\left(\frac{2\pi}{M}xX + \frac{2\pi}{N}yY\right)$$



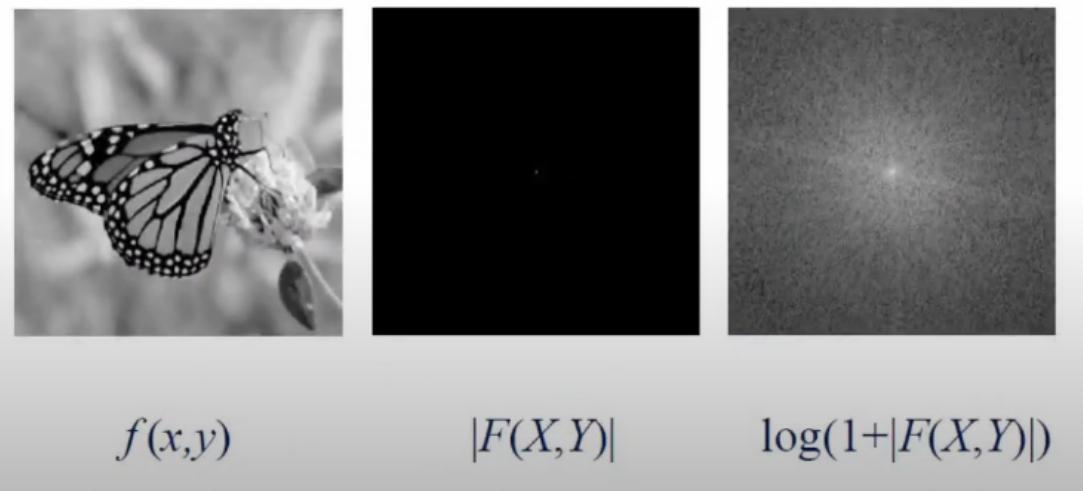
2D diszkrét furier transzformáció: A fourier transzformáció szeparálható, tehát elvégezzük először sorfolytonosan, majd oszlopfolytonosan a transzformációt, és megkapjuk a fourier transzformáltat. Ezért akár gyorsíthatjuk is az fft módszerrel.

A 2D DFT szeparálható

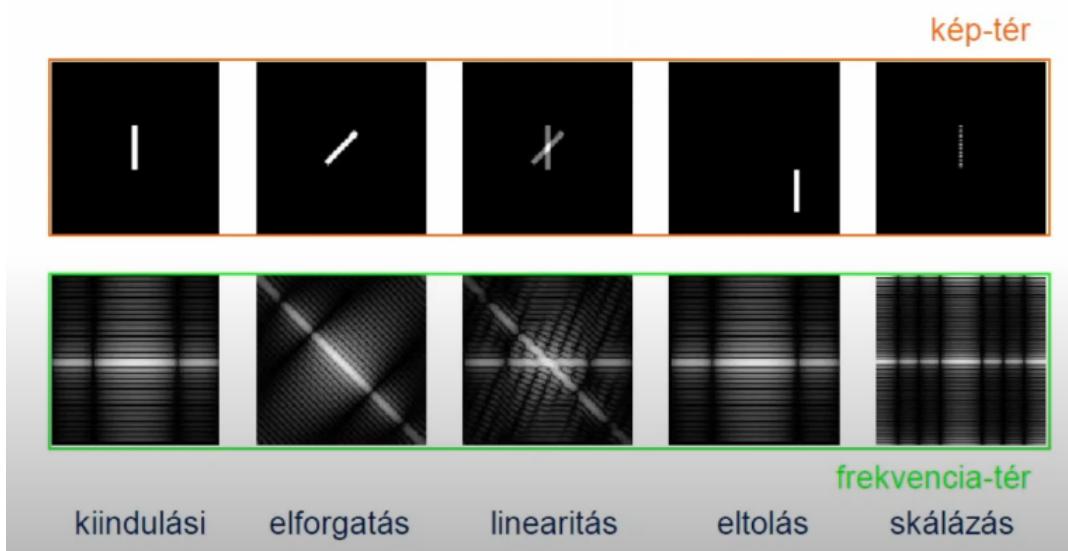


Frekvenciatér: Ebben ábrázolódik a képek fourier transzformált párja. Ezt ha megjelenítjük nem mutat sokat, csak egy fehér folt egy fekete háttéren, ez mutatja a kép átlag szürkeségét. De mivel a szemünk logaritmikusan érzékeny a fény erősségrére, ezért ha $\log(1+|F(X,Y)|)$ átalakítást végezünk, majd ezt jelenítjük meg akkor sokkal látványosabb az eredmény.

A frekvenciakép megjelenítése



Tulajdonságok



Konvolúció: Két jelet vagy képet kap inputként. Olyan, mint a szembepacsizó játékosok. Lényegében ez egy polinom szorzás. Felírható mátrix vektor szorzásként. Művelet igénye M^2 .

Példa konvolúcióra

$$M=N=3 \quad h = f^* g$$

$$\begin{array}{c} \leftarrow \quad g(0) \quad g(1) \quad g(2) \\ f(2) \quad f(1) \quad f(0) \quad \rightarrow \end{array}$$

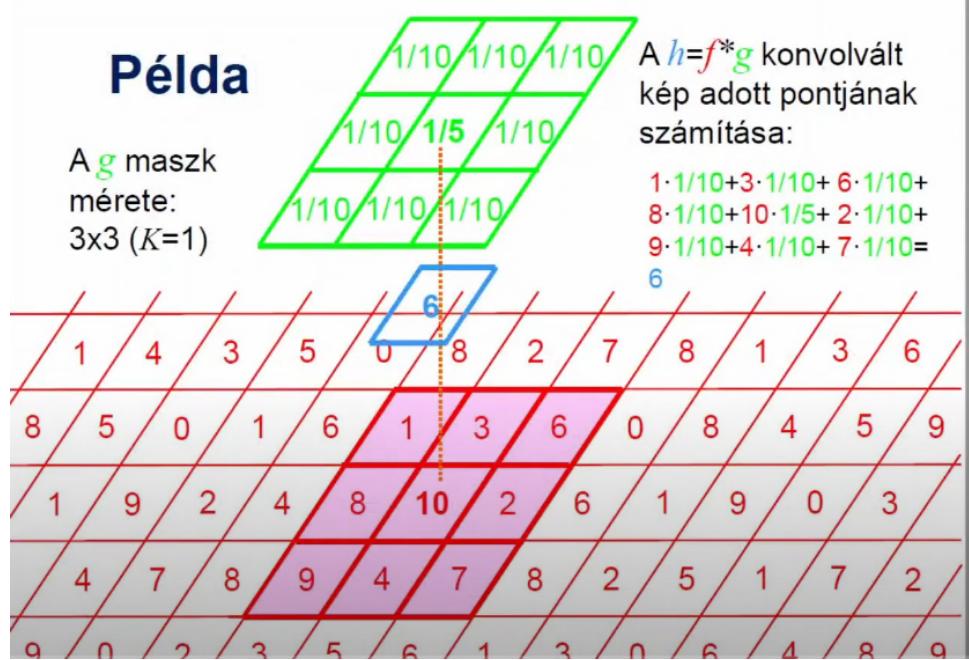
kiindulási helyzet

1D konvolúció – mátrix-vektor szorzás

$$h = f^* g \quad (M = 4, N = 3)$$

$$\begin{bmatrix} f(3) & f(2) & f(1) & f(0) & 0 & 0 \\ 0 & f(3) & f(2) & f(1) & f(0) & 0 \\ 0 & 0 & f(3) & f(2) & f(1) & f(0) \\ 0 & 0 & 0 & f(3) & f(2) & f(1) \\ 0 & 0 & 0 & 0 & f(3) & f(2) \\ 0 & 0 & 0 & 0 & 0 & f(3) \end{bmatrix}_{(M+N-1) \times (M+N-1)} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ g(0) \\ g(1) \\ g(2) \\ g(3) \end{bmatrix}_{(M+N-1) \times 1} = \begin{bmatrix} h(0) \\ h(1) \\ h(2) \\ h(3) \\ h(4) \\ h(5) \end{bmatrix}_{(M+N-1) \times 1}$$

Konvolúció 2D-ben: Egy MxN-es kép az input és egy konvolúciós maszk.



Korreláció:

$$f(x) \otimes g(x) = \overline{f}(-x) * g(x)$$

↑
komplex konjugált

Konvolúciós tételek: A konvolúciót úgy is elvégezhetjük, hogy vesszük a két kép fourier transzformáltját, majd azokat pontonként szorzással összeszorozzuk, majd visszatérünk fourier transzformációval a képtérbe. Ez gyorsabb, mint ha képtérbe konvolválnánk. Időigénye $O(M^2 \log_2 M)$

$$F(f * g) = F(f) \cdot F(g)$$

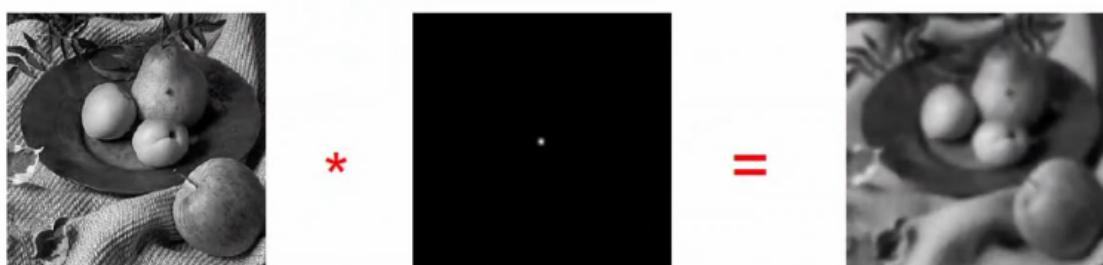
$$F(f \cdot g) = F(f) * F(g)$$

$$f * g = F^{-1}(F(f) \cdot F(g))$$

$$f \cdot g = F^{-1}(F(f) * F(g))$$

Példa a konvolúciós tételere

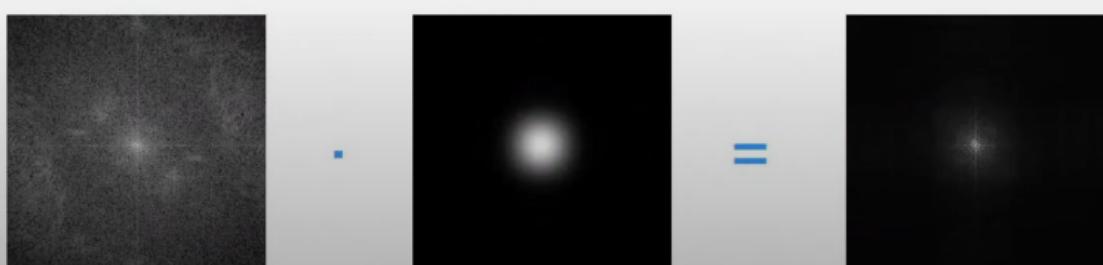
konvolúció



*

=

Fourier tr.



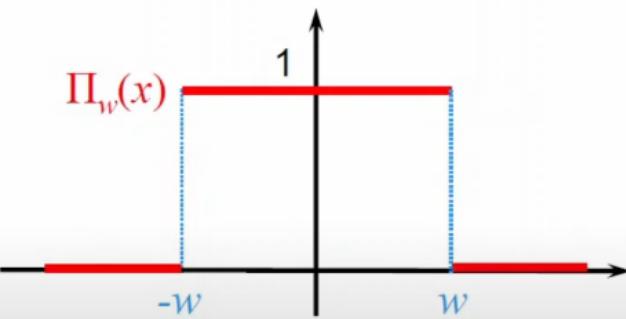
.

=

pontonkénti szorzás

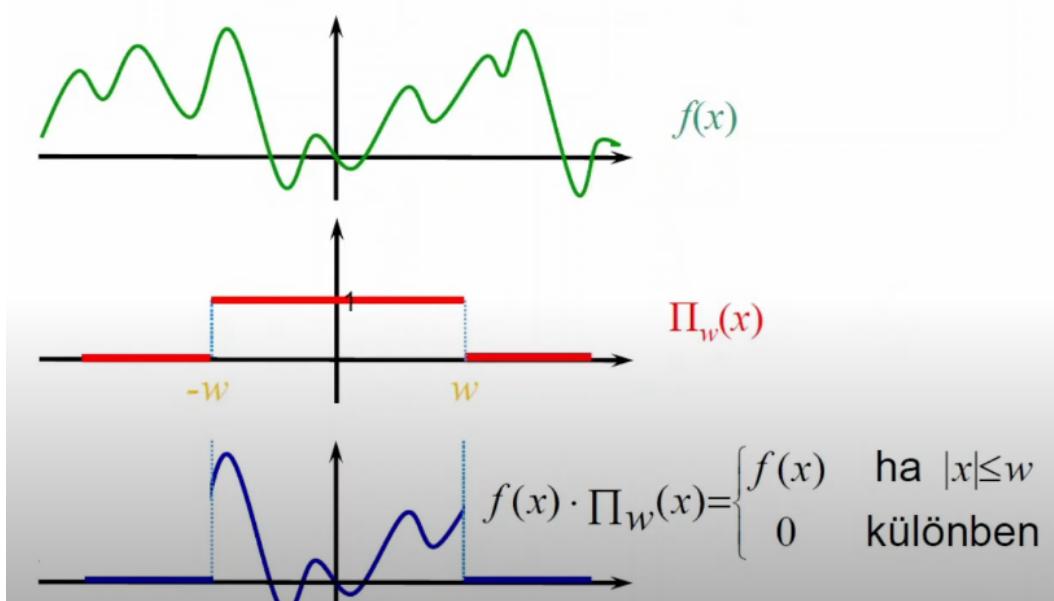
Doboz függvény: Egy intervallumon belül a függvény értéke egy amúgymindenhol 0, szimmetrikus. Ezzel a fügvénnyel ha pontonként rászorzunk egy másik függvényre akkor kihasíthatunk belőle egy részt.

Doboz függvény



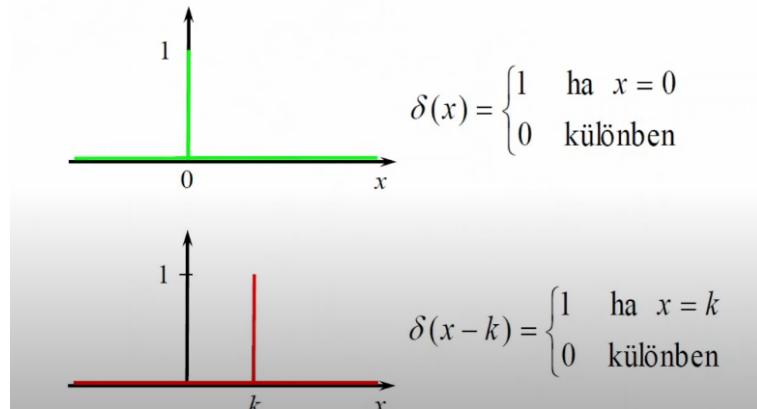
$$\Pi_w(x) = \begin{cases} 1 & \text{ha } |x| \leq w \\ 0 & \text{különben} \end{cases}$$

Pontonkénti szorzás doboz fügvénnyel

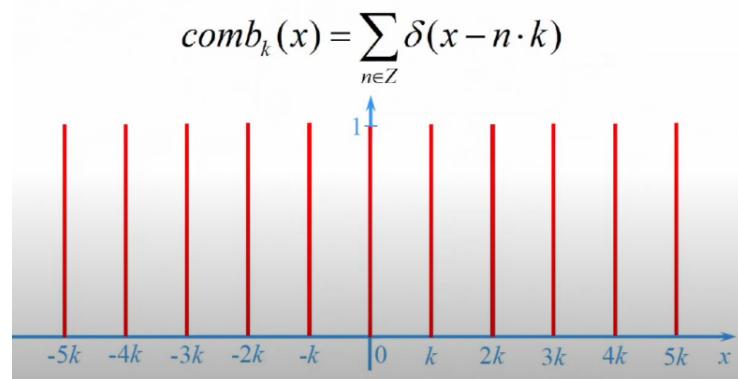


Delta függvény: Egy szálka ami 0-ban 1 amúgy mindenhol 0. Ezt lehet eltolni is. Ebből kialakítható a fésűs függvény (Dirac fésű). Ha a fésűs függvénnel pontonként szorzunk akkor mintavételezünk.

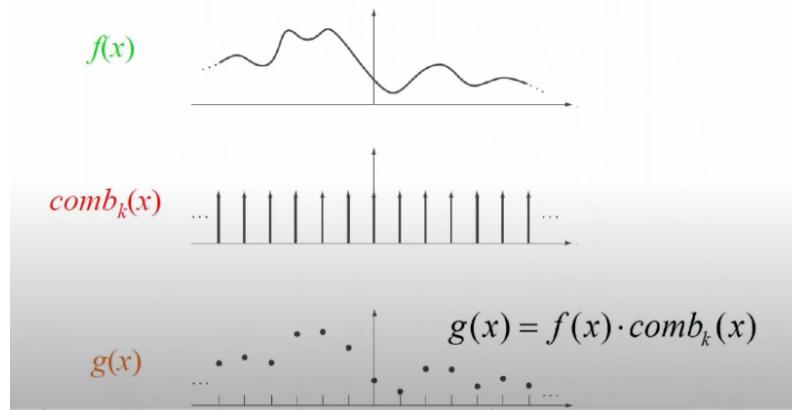
A δ -függvény és eltoltja



Dirac fésű (*Shah function*)

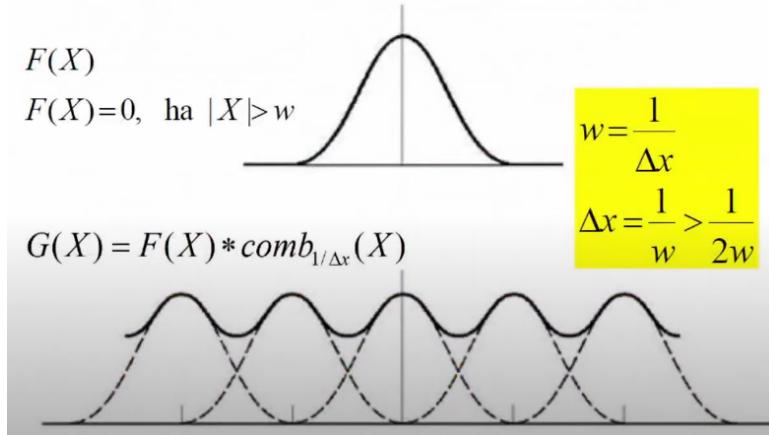


Pontonkénti szorzás a Dirac fésűvel – mintavételezés

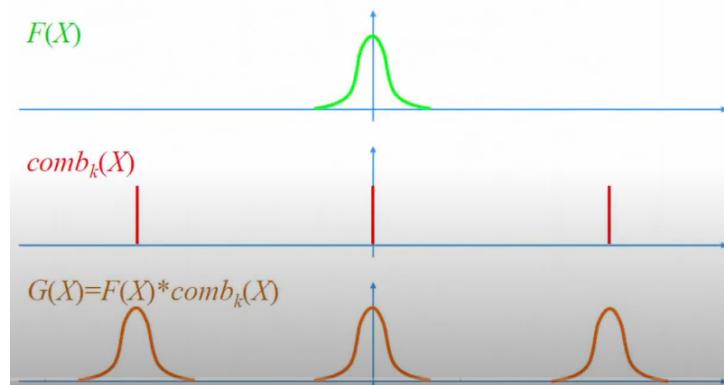


Fésűs függvénytel való konvolúció: Ekkor másolás történik. Ha túl sűrűn vannak a fésű fogai akkor a másolt tagok fedik egymást, ezt **aliasing**nak nevezzük azaz átfedésnek, álcázásnak. Ha nincs átfedés akkor a dobozfüggvénytel a periodikusan másolt függvényből kihasíthatjuk az eredeti függvényt, ha átfedés van akkor nem kapjuk vissza az eredetit.

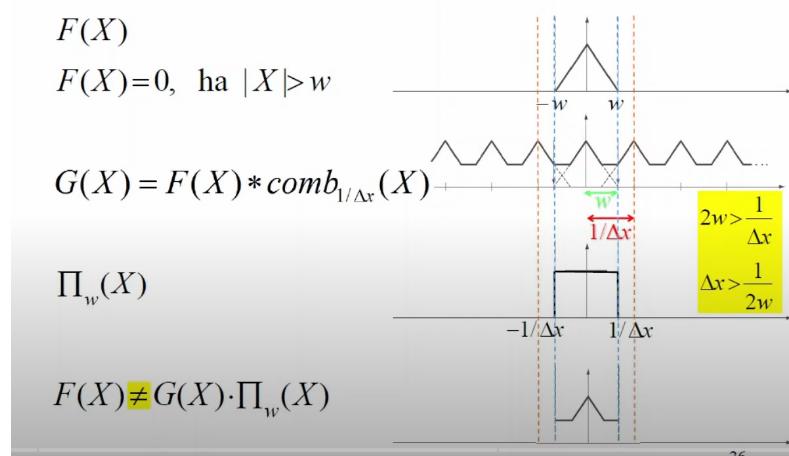
Átfedés, álcázás (aliasing)



Konvolúció a Dirac fésűvel – periodikus másolás

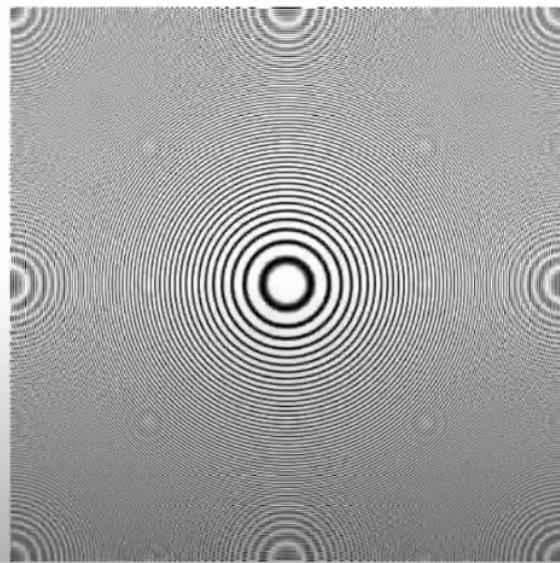


Ha van átfedés



Alulmintavételezés: Az előbbi pár pontból fakadóan a képen hullámok jelenhetnek meg.

Átfedés, álcázás (*aliasing*)



a képen
hamis/fantom
minták jelennek
meg

Sávhatárolt: Az a függvény amely fourier transzformációja egy idő után lecseng, tehát w -nél nagyobb frekvenciákat már nem tartalmaz. Az az egy valamilyen sugarú körön kívül lecseng. A w a Nyquist frekvencia.

Sávhatárolt (*bandlimited*) függvények

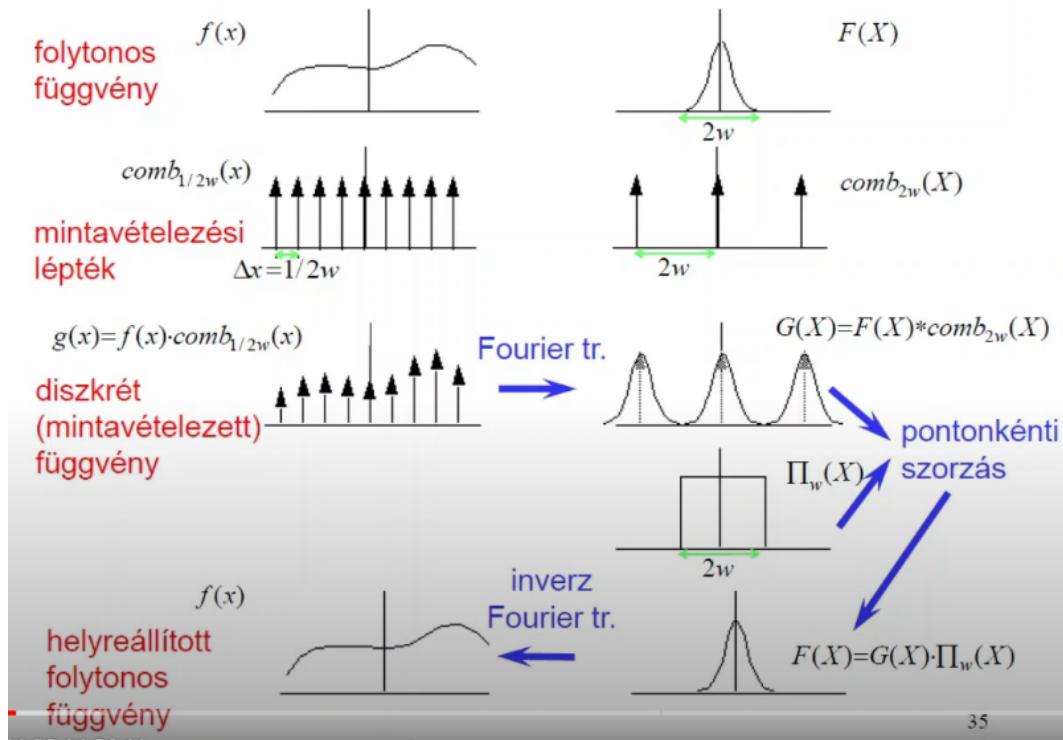
Az $f(x)$ függvény **sávhatárolt**, ha létezik olyan w , hogy $F(X)$ (vagyis az $f(x)$ Fourier transzformáltja) nem tartalmaz w -nél nagyobb frekvenciákat, azaz:

$$F(X)=0, \text{ ha } |X|>w .$$

w -t **Nyquist-frekvenciának** nevezzük, ha

$$F(X)=0, \text{ ha } |X|>w, \text{ ugyanakkor } F(-w) \neq 0 \text{ vagy } F(w) \neq 0 .$$

Mintavételezési téTEL: Arra ad választ, hogy hogyan mintavételezzük úgy, hogy a mintavételezett kép ugyan azt az információt hordozza, mint az eredeti. Tudjuk, hogy egy n-ed fokú polinom tetszőleges $n+1$ pontja alapján visszarajzoható. Az eredeti függvényen végezünk egy mintavételezést, tehát egy fésűs függvény segítségével pontonként szorzást akkor diszkrét függvényt kapunk. Majd ha annak vesszük a fourier transzformáltját, akkor kapunk egy periodikusan ismétlődő függvényt, amire ha rászorunk pontonkénti szorzással, egy dobozfüggvénnyel akkor kihasíthatjuk a közepét, amit ha vissza transzformálunk inverz fourierrel, megkapjuk az eredeti képünket. Tehát egy diszkrét mintavételezett függvényből a frekvenciatérben visszaállíthatjuk az eredeti függvényt.



Whittaker-Shannon mintavételezési téTEL: Ha a függvény w -vel sávhatárolt, akkor ha a mintavételezési lépték az $\Delta x \leq 1/2w$ akkor visszaállítható a mintavételezett kép. 2D-ben is ugyan így megállja a helyét a tétel.

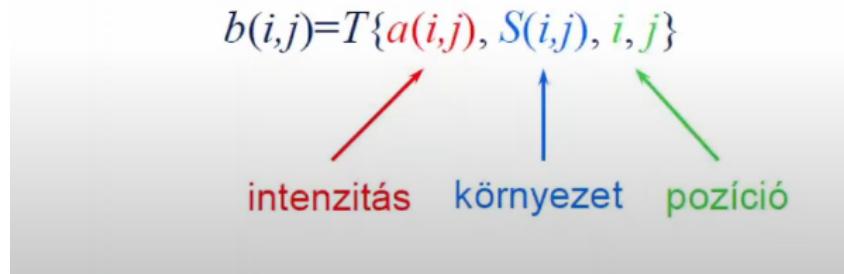
Whittaker-Shannon mintavételezési tétele

Az 1D folytonos, w -nél nagyobb frekvenciákat nem tartalmazó (sávhatárolt) jel akkor és csak akkor állítható vissza a mintavételezettjéből, ha a Δx mintavételezési léptékre teljesül az alábbi feltétel:

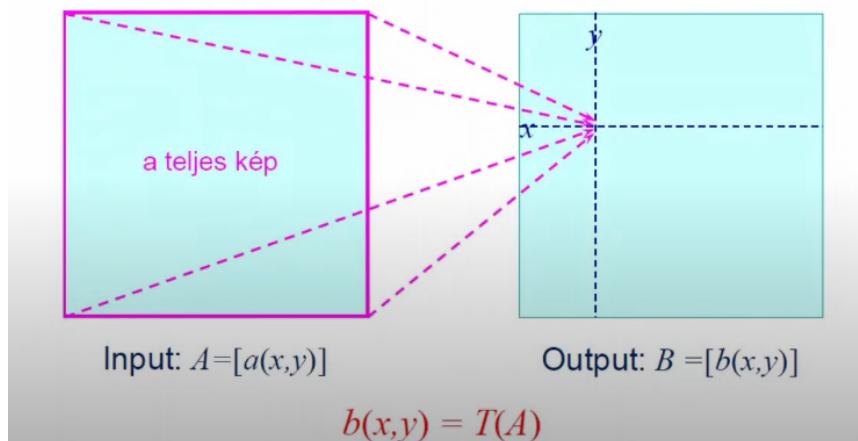
$$\Delta x \leq \frac{1}{2w} .$$

Egy képes képművelet: Egy input képből egy output képet csinál. Az, hogy egy adott pixel a képen hogyan változik meg függhet a pixel intenzitásától, a környezetében lévő pixelek intenzitásától, és a pixel pozíójától a képen.

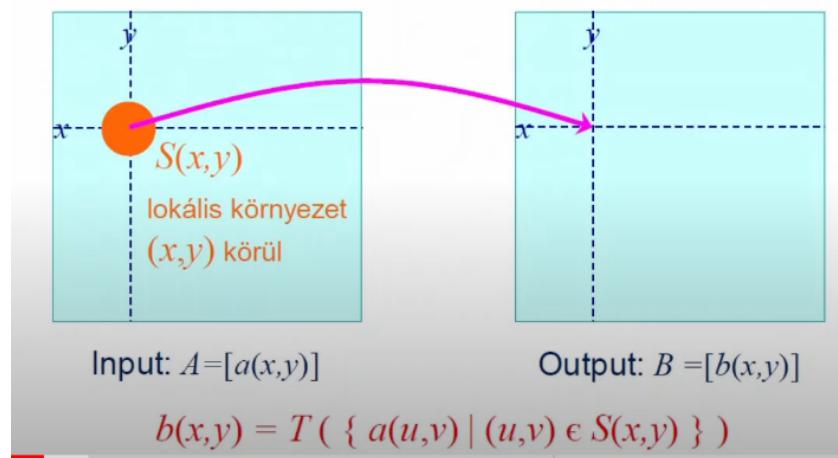
$$T: A = [a(i,j)] \rightarrow B = [b(i,j)]$$



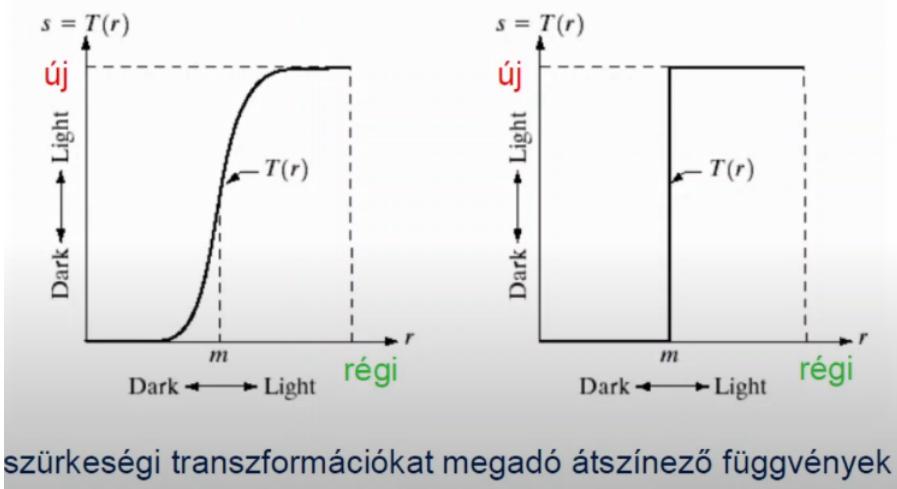
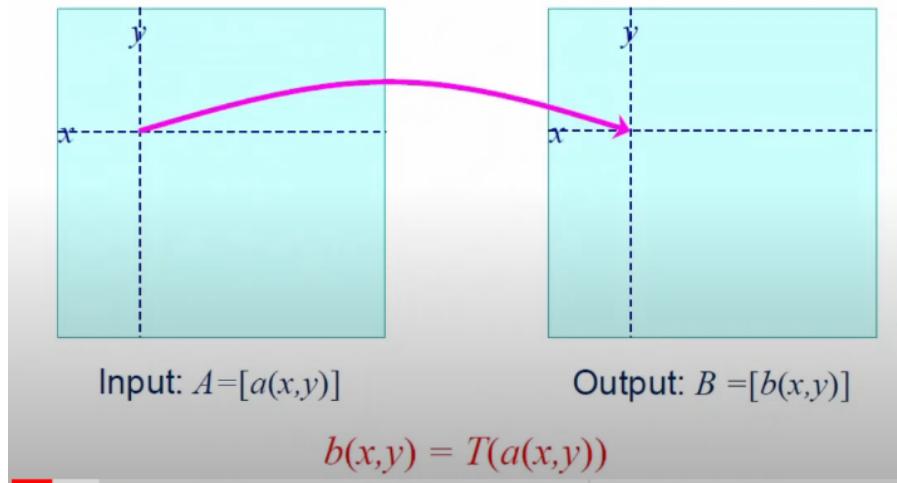
Globális képművelet: Az input kép minden pixele befolyásolja az output kép egy egy pixelét. Pl Fourier.



Lokális képművelet: Az outputképen lévő egy egy pixel intenzitását befolyásolja, az inputképen az adott pixel környezete is.



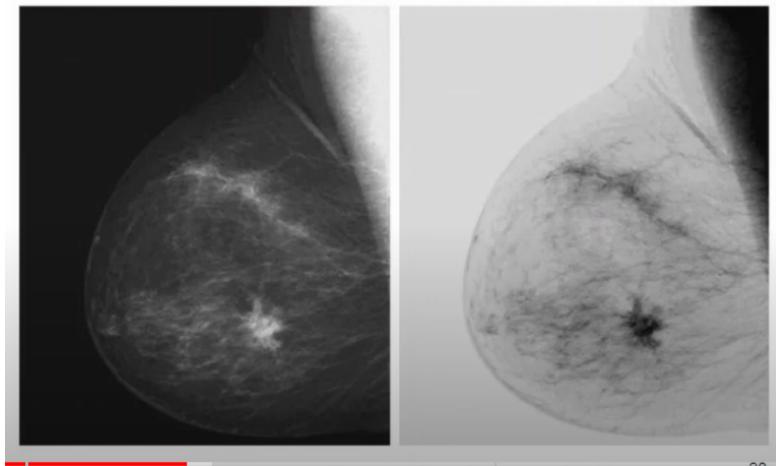
Pont-operáció: Az outputképen szereplő pixel intenzitása csak attól függ, hogy mi volt az input képen lévő pixel intenzitás. pl régi szín új szín transzformáció. Megadható függvényel illetve átszínező/keresőtáblázattal (LUT-ok/look up table).



index	érték
0	255
1	254
2	253
3	252
4	251
5	250
...	...
254	1
255	0

átszínező- vagy keresőtábla
(*LUT: look-up-table*)

Negálás: Pontoperáció.



Álszínezés: Pontoperáció. Más intenzitású szürkéhez más színeket adunk.



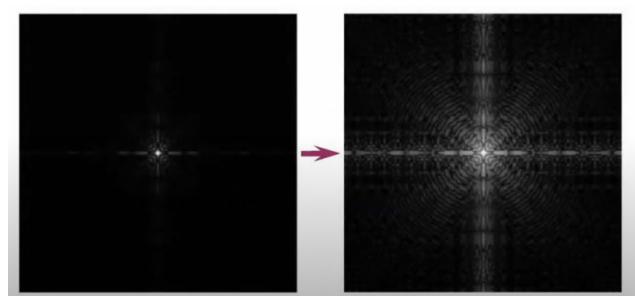
szürkeárnyalatos \rightarrow RGB

$(T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow)$

Logaritmikus transzformáció: Pontoperáció. Fourier transzformációnál nem kivehető az eredmény, ezért ezzel a transzformációval jobban láthatjuk az eredményt. A szemünk logaritmikusan érzékeny a fényre.

$$s = c \cdot \log(1+r)$$

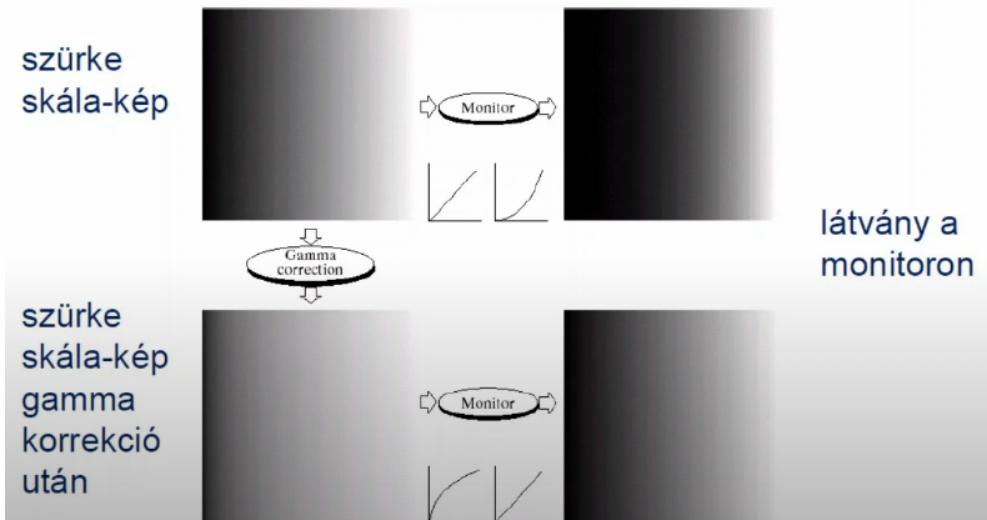
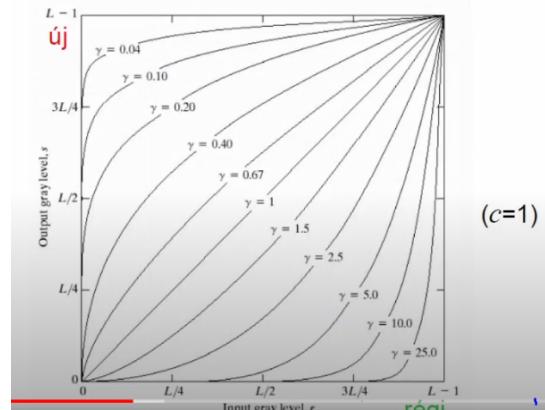
- r : régi intenzitás
- s : új intenzitás
- c : konstans ($c > 0$)



Gamma korrekció: A régi intenzitást a gammaadik hatványra emeljük. Arra jó hogy egyenletes szürke skálát kapunk. Ahhoz hogy szemünk a monitoron egyenletesnek lássa a szürke skálát érdemes gamma korrekciót végezni. Szűrkeárnyalatos képeket korrigálhatunk vele hogy jobban kivehetők legyenek a részletek.

$$s = c \cdot r^\gamma$$

- r : régi intenzitás
- s : új intenzitás
- c : konstans ($c > 0$)
- γ : konstans ($\gamma > 0$)

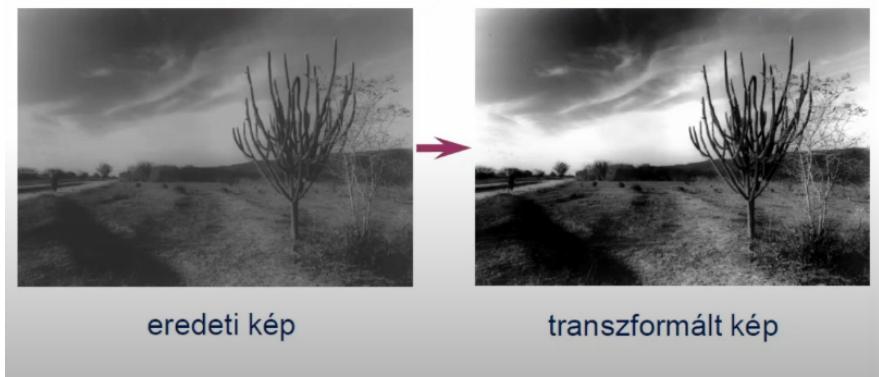


Példa gamma korrekcióra

eredeti légifotó	$\gamma=3$
$\gamma=4$	$\gamma=5$

($c=1$)

Kontraszt széthúzás: Pontoperáció.



Küszbölés I.: Pontoperáció. Többszintű képből binárisát gyárt. Adott T intenzitás felett az adott pixel legyen hófehér, ha nem akkor pedig fekete. Ezt tudjuk fordítva is alkalmazni.

$$g(x,y) = \begin{cases} 1, & \text{ha } f(x,y) \geq T \\ 0, & \text{különben} \end{cases}$$

vagy

$$g(x,y) = \begin{cases} 1, & \text{ha } f(x,y) \leq T \\ 0, & \text{különben} \end{cases}$$



Küszöbölés II.: Pontoperáció. **Intervallumos küszöbölés.** Azok a pixelek lesznek fehérek, amelyek egy adott intenzitás intervallumba esnek, a többi nyilván fekete lesz.

$$g(x, y) = \begin{cases} 1, & \text{ha } f(x, y) \in D \\ 0, & \text{különben} \end{cases}$$

D : szürkeségi értékek halmaza
(pl. egy intervallum)



(a)



(b)



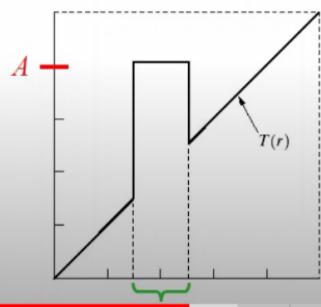
Intervallumos küszöbölés

- (a) eredeti kép,
- (b) $T=10$,
- (c) $T=20$,
- (d) $20 < T < 45$

43

Kitakaró küszöbölés: Egy bizonyos tartományon kívüli intenzitással rendelkező pixelek intenzitása maradjon meg, minden más legyen fekete vagy fehér. Ekkor Nem bináris kép lesz az eredmény.

$$g(x, y) = \begin{cases} f(x, y), & \text{ha } f(x, y) \notin D \\ A, & \text{különben} \end{cases}$$

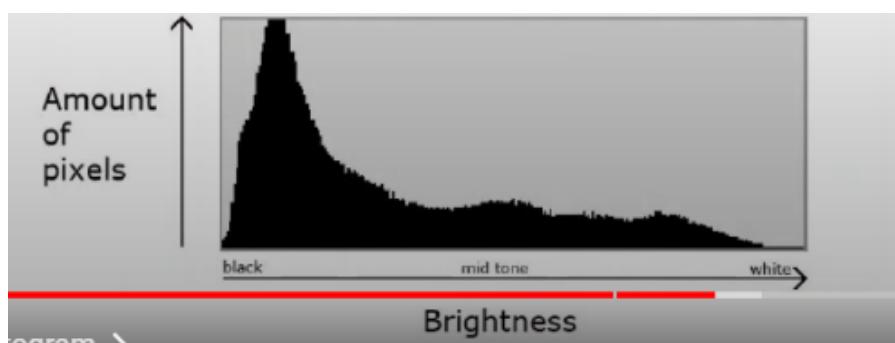


Bitsíkok kiemelése: Pontoperáció. Például egy 8 bites kép szétbontható nyolc darab bináris képre. A bitsíkok kiemelése küszöböléssel történik, ha 128al küszöbölnk, akkor 8. bitsíket tudjuk kiemelni, ha 64 el akkor a 7.-et stb... Tömöríthetünk is ezzel a módszerrel, ha csak az alsó pár bitsíket tartjuk meg eszre se fogjuk venni a különbséget.

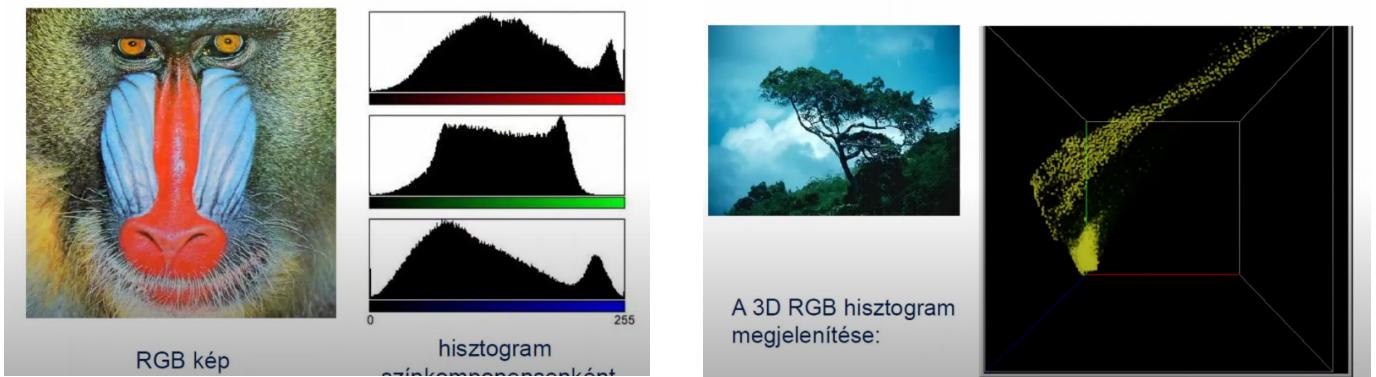
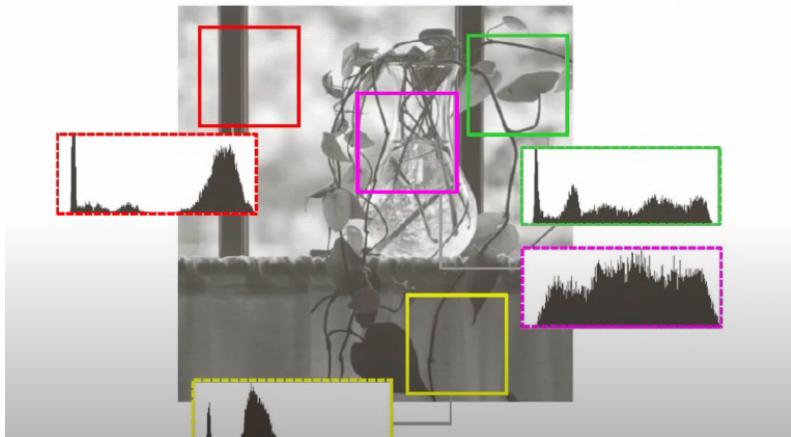


53

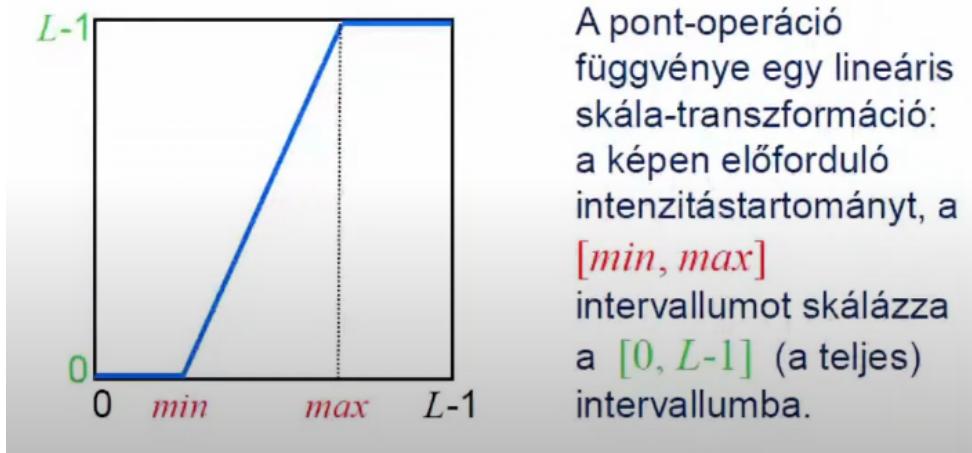
Hisztogram: Képek esetén annak a statisztikája, hogy milyen intenzitású képből mennyi van az adott képen. Van globális illetve lokális hisztogram is. A lokális hisztogram esetében a képe egy részletét kiemeljük és arra egyedi hisztogramokat készítünk. Színes kép hisztogramja, az egyes színek előfordulását mutatja az adott pixel esetében (3 db hisztogram RGB szinekre). Lehetséges a színes hisztogram egy x,y,z-s koordináta-rendszerben ábrázolt hisztogram is. Egy hisztogramhoz több kép is tartozhat.

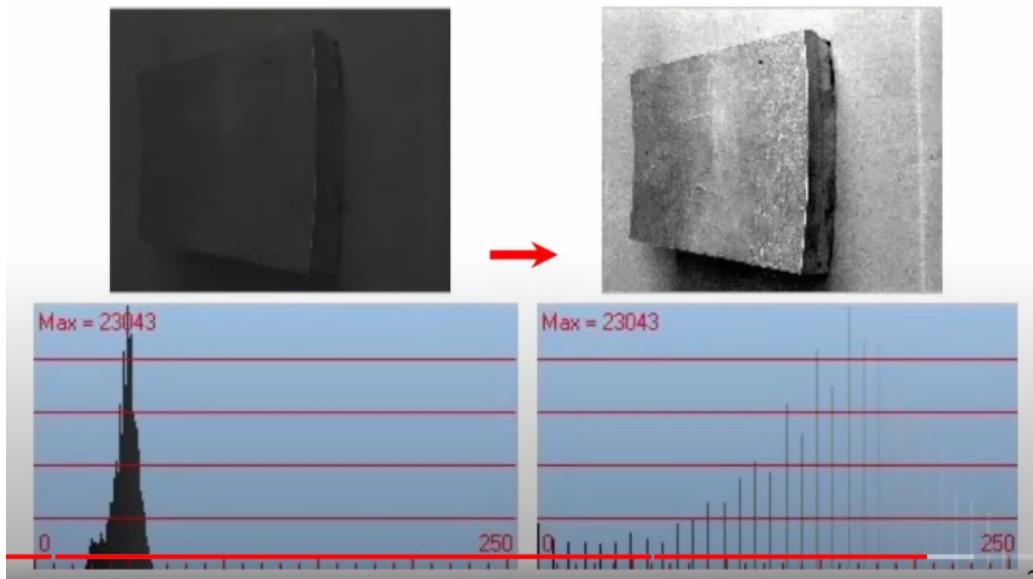


Lokális hisztogram

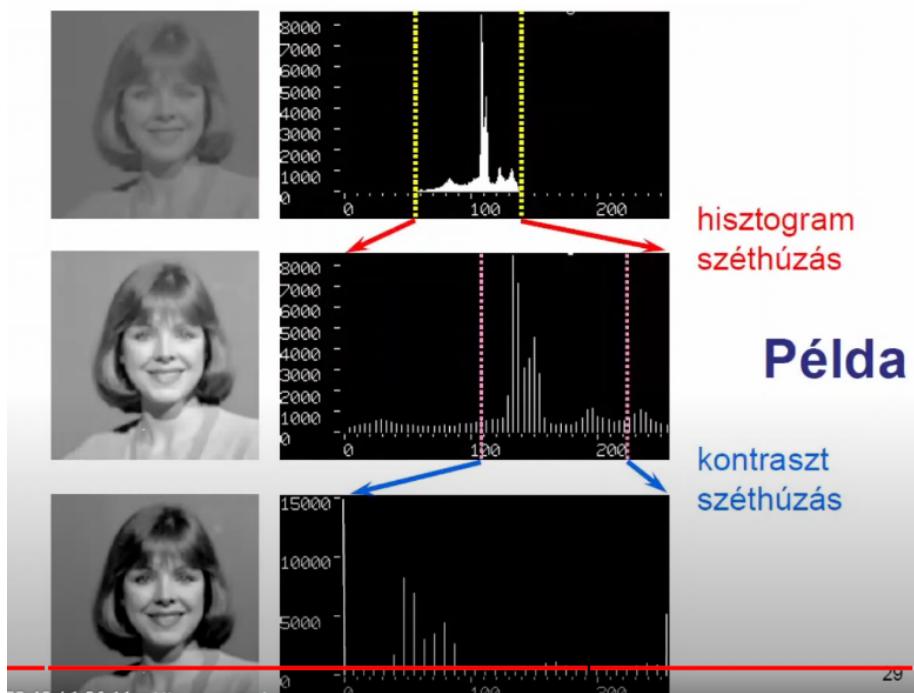


Hisztogram széthúzás: Pontoperáció. A képen megkeressük a min és a max intenzitást és azokat kitoljuk a 0 és a 255-be. Majd az egész intenzitásskálánkat szépen kitoljuk a két szélső értékkel együtt.



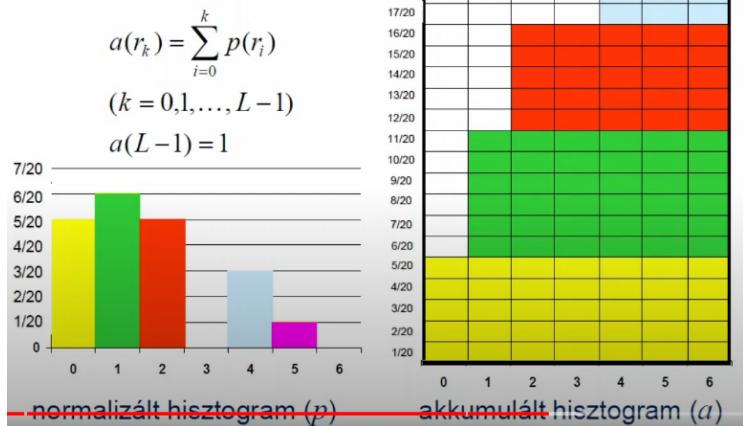


Kontraszt széthúzás: Hisztogramszéthúzás után, keresünk egy olyan intervallumot ahol a legtöbb intenzitás érték helyezkedik el, majd ezeken kívül minden másat letolunk a feketébe akkor egy szébb képet kapunk.

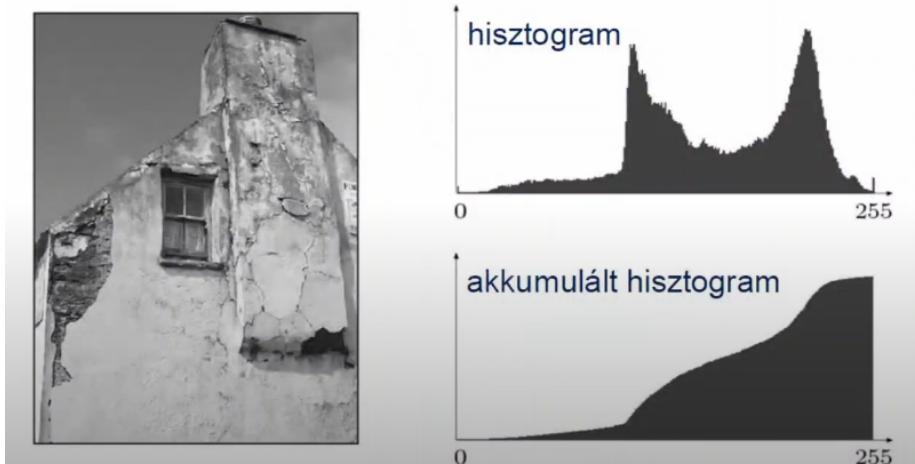


Akkumulált hisztogram: Az egyes intenzitások sorban rápakolódnak egymásra, ebből kifolyólag minden monoton nő.

Akkumulált hisztogram



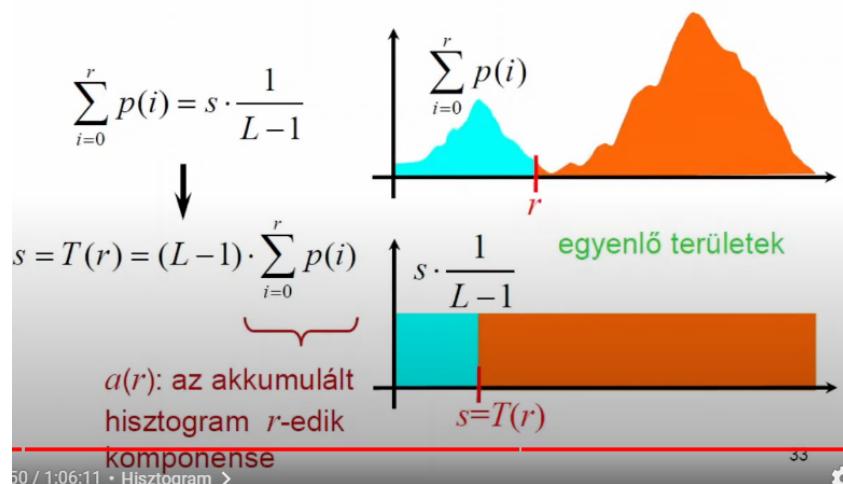
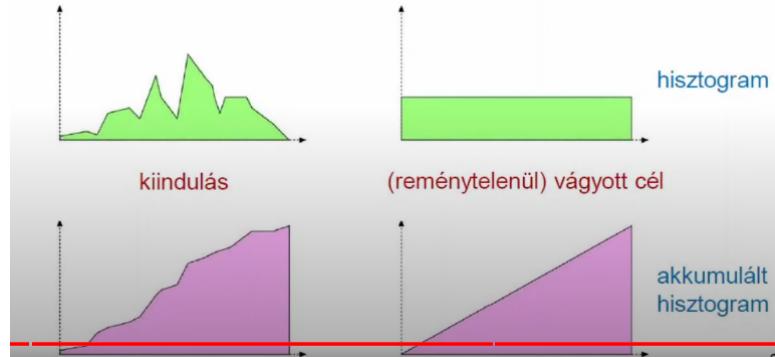
Példa hisztogramra



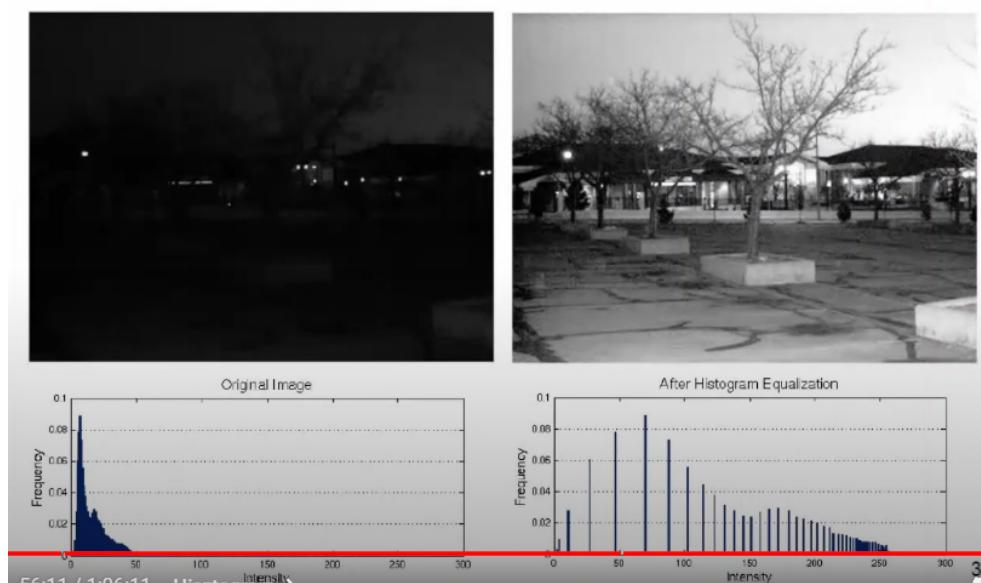
Hisztogram kiegyenlítés: Globális operáció. A cél az, hogy minden intenzitás közel egyforma legyen. A hisztogram sose fog egyenes lenni, de az akkumulált hisztogram megoldható, hogy lineáris képet mutasson. A hisztogram kiegyenlítést az akkumulált hisztogrammal végezzük el.

Hisztogram kiegyenlítés

„közel” konstans hisztogramú képet eredményező pont-operáció

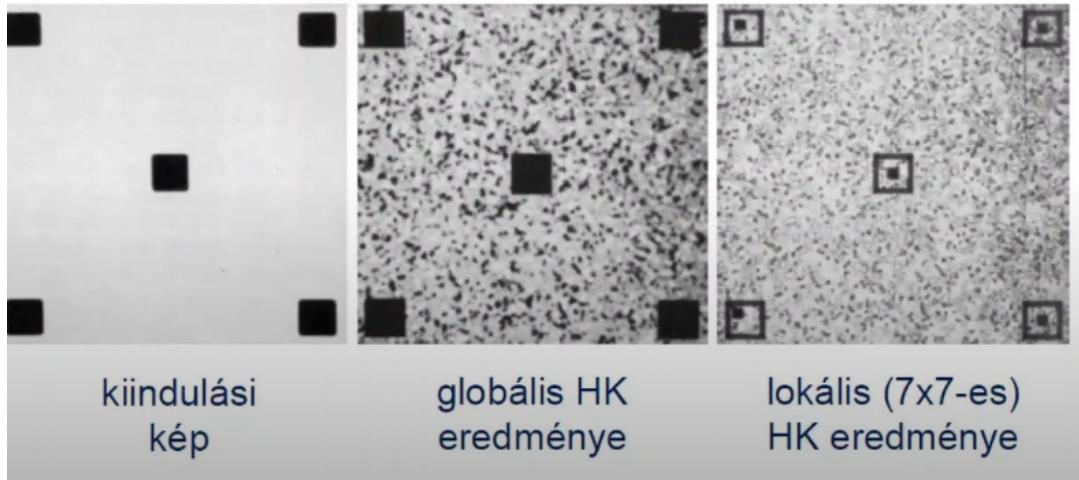


Példa hisztogram kiegyenlítésre



Lokális hisztogram kiegyenlítés: Lokális operáció. Az input kép minden egyes pontjára elvégezzük, hogy az adott pont és egy lokális környezetére (mondjuk 7x7-es) elvégezzük a hisztogram kiegyenlítést, majd a kapott eredményen a középső pixelt visszavisszük az inputképre. Más méretű környezettel más-más eredményt kapunk. Színesképeken is elvégezhető, de RGB-be nagyon rossz eredmény kapunk, viszont HSV színtérben, ha világosság szerint egyenlítenk ki akkor tökéletes eredményt kapunk színes képre is.

Lokális hisztogram kiegyenlítés



HK színes (HSV) képeken



kiindulási kép



HK a V csatorna alapján

HK színes (RGB) képeken



kiindulási kép

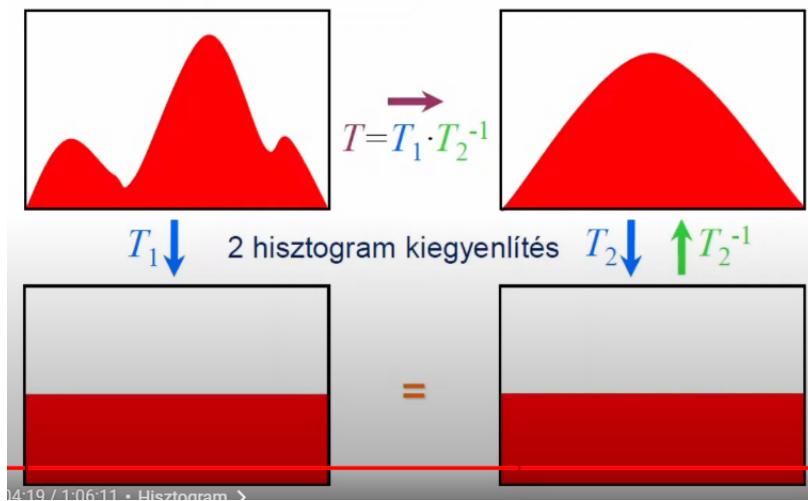


HK az R-csatorna alapján



Hisztogram specifikáció/illesztés: Előre megadjuk, hogy milyen intenzitás hisztogramot szeretnénk kapni. Az eredeti és a kívánt hisztogramra végezzünk hisztogram kiegyenlítést, majd ezeket az átszínező függvények tartjuk meg. A kívánt hisztogramon végezzett hisztogram kiegyenlítésen végezzünk visszafele transzformációt és ezt az átszínező függvényt, szorozzuk össze az eredeti kép átszínező függvényével és akkor megkapjuk a kívánt eredményt a képre. Nem tökéletes lesz az eredmény.

Hisztogram specifikáció

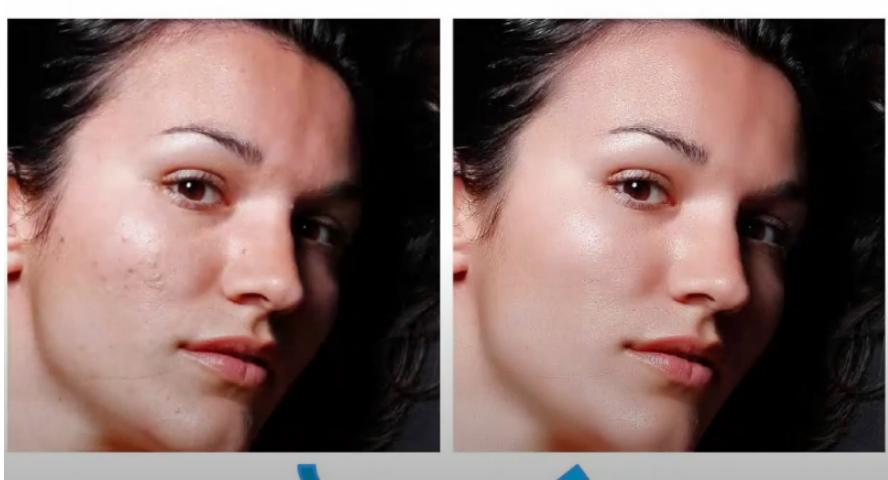


Többképes műveletek: Összeadás, kivonás, átlagolás, szorzás (bináris képpel való szorzás a maszkolás).

Zaj: Gauss-zaj esetében az intenzitásválozás normál eloszlású, a só-bors zajnál pedig a fehér és fekete pontok random fordulnak elő.



Simítás: Sminkelés. Eljárások: Átlag-szűrés, súlyozott átlag-szűrés,



Átlag-szűrés: Simító eljárás. Ekkor az adott pixelnek egy környezetét vesszük például 3x3-as környezetét, és az ebben a környezetben szereplő intenzitások átlagát számítjuk ki, és az lesz az új

intenzitás. Az **átlagszűrés felírható konvolúcióval**, ekkor a 3x3as konvolúciós maszk minden eleme 1/9 ed (1/3x3). A maszk elemek összege mindig 1. Homályosít, élek elvesznek, de a zaj csökkenthető vele.

$$g(i,j) = \frac{1}{|S_{(i,j)}|} \cdot \sum_{(m,n) \in S_{(i,j)}} f(m,n),$$

ahol f a kiindulási kép,
 g a szűrt kép,
 $S_{(i,j)}$ az (i,j) pont egy környezete,
 $|S_{(i,j)}|$ a környezetbe tartozó képpontok száma.

Pl. a  3x3-as környezet esetén:

$$g(i,j) = \frac{1}{9} \cdot \sum_{u=-1}^1 \sum_{v=-1}^1 f(i+u, j+v)$$

Átlag-szűrés konvolúcióval

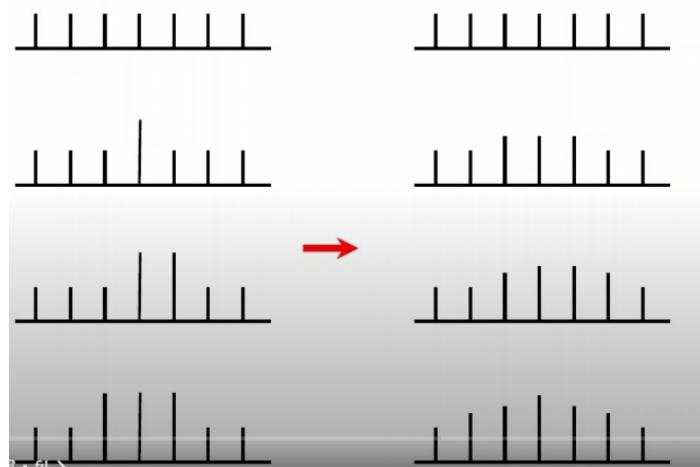


$$\begin{aligned} g(i,j) &= \frac{1}{9} \cdot \sum_{u=-1}^1 \sum_{v=-1}^1 f(i+u, j+v) \\ &= \sum_{u=-1}^1 \sum_{v=-1}^1 f(i+u, j+v) \cdot h(u,v) \\ &= (f * h)(i,j), \end{aligned}$$

ahol :

$$h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{konvolúciós maszk}$$

1D átlag-szűrés az [1/3,1/3,1/3] maszkkal



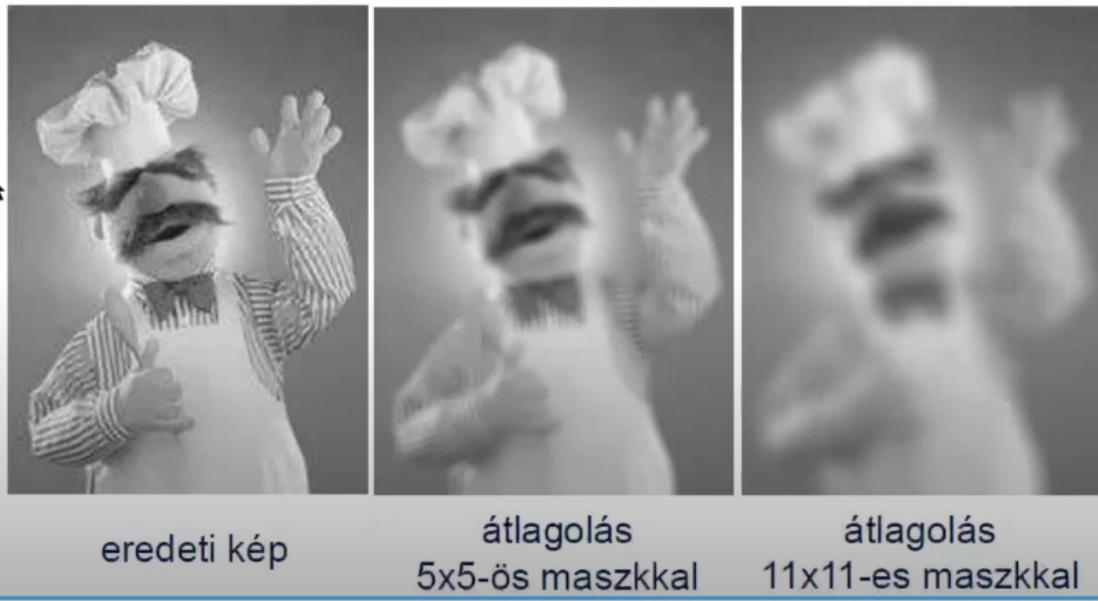
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0



f

$$h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$g=f^*h$



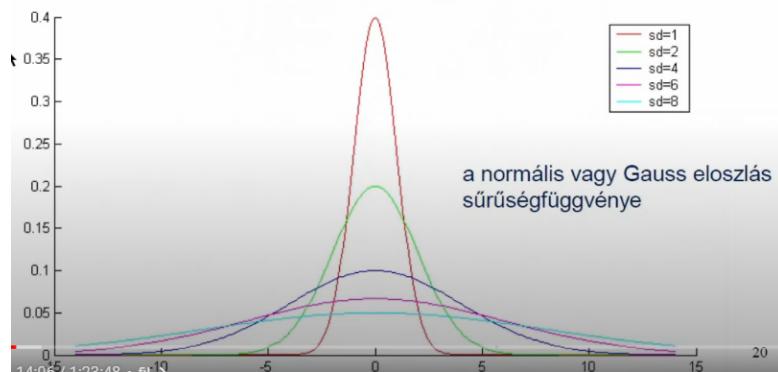
Súlyozott átlag-szűrés: Simítás, működik konvolúcióval. Hasonló az átlag-szűréshez, csak itt a maszk elemek értéke a pixeltől való távolságától függően arányosan csökken. Tehát ami közelebb van, nagyobb befolyást kap, ami messzebb az kisebbet. Maszkelemek összege itt is 1 (piros: átlag-szűrés, zöld: súlyozott átlag-szűrés). Ehhez a szűréshez használhatjuk a **Gauss függvényt**, hogy megkapjuk a maszkelemek intenzitását. A Gauss függvény 2D-ben a Gauss harang, amit két Gauss függvény összeszorzásával kaphatunk meg. Ez képként egy középen fehér és egyre halványuló szabályos kör. Ahhoz hogy ezt használni is tudjuk, diszkréten közelítenünk kell a Gauss függvényt. Ehhez a **Pascal háromszöget** hívjuk segítségül, például ha 3x3-as mátrixot generálunk, akkor a harmadik sort használjuk belőle. Tehát a simított szűrt képünket úgy kapjuk, hogy a képet konvolváljuk egy Gauss maszkkal, ezt nevezzük **Gauss szűrésnek** is.

$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Az 1D Gauss függvény

$$G_\sigma(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{(-x^2/2\sigma^2)} \quad (\sigma > 0)$$



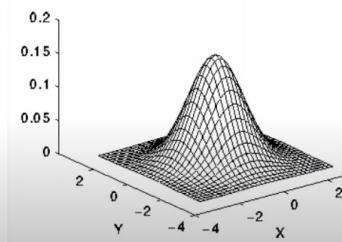
2D Gauss függvény

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-(x^2 + y^2)}{2\sigma^2}\right)$$

Szeparálható:

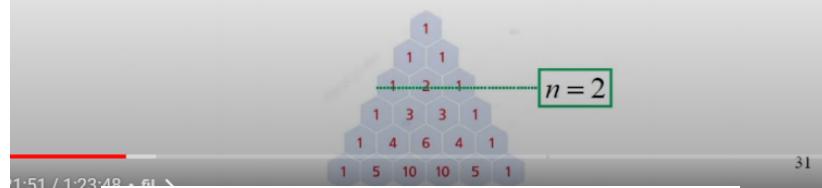
$$G_\sigma(x, y) = G_\sigma(x) \cdot G_\sigma(y)$$

2D Gauss függvény

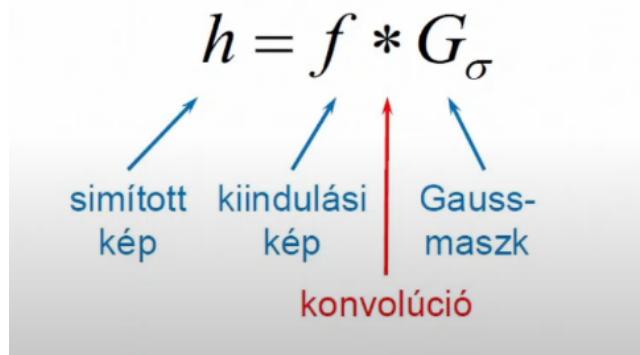


A 2D Gauss függvény diszkrét közelítése

$$\begin{bmatrix} \frac{1}{4} \\ \frac{2}{4} \\ \frac{1}{4} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{4} & \frac{2}{4} & \frac{1}{4} \end{bmatrix} = \begin{bmatrix} \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \\ \frac{2}{16} & \frac{4}{16} & \frac{2}{16} \\ \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



Gauss szűrés

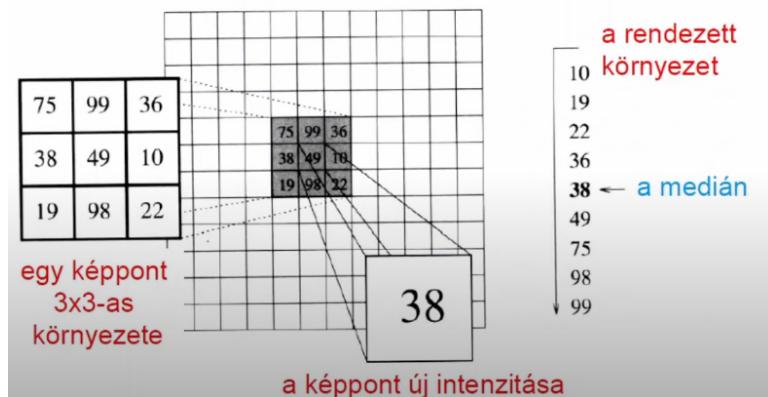


Medián szűrés: Simítás. Az adott pixel intenzitását helyettesítjük egy S környezetében lévő képpontok mediánjának intenzitásával. Ha tudjuk, hogy só-bors zajunk van akkor érdemes medián szűrést alkalmazni. Gauss zaj esetében jobban működik az átlagszűrés! Jobban megőrzi az éleket.

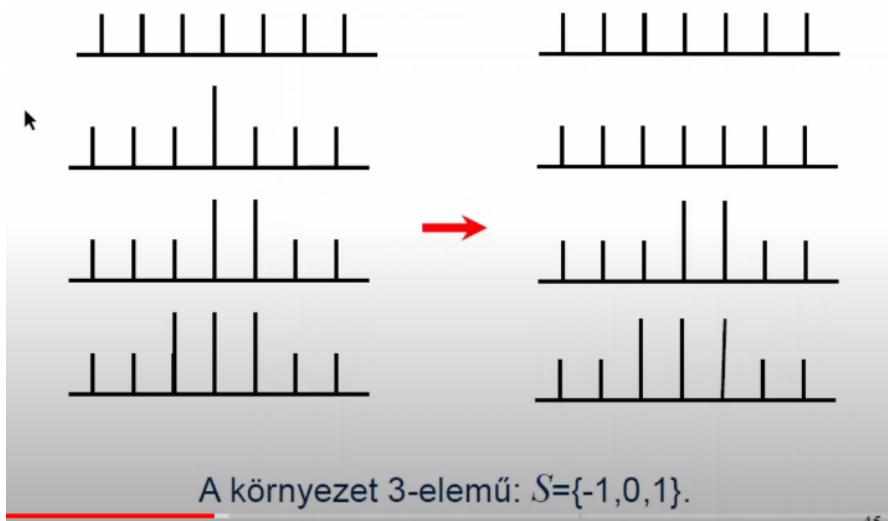
Medianszures

$$g(i,j) = \text{med}\{ f(i+u, j+v) \mid (u,v) \in S \}$$

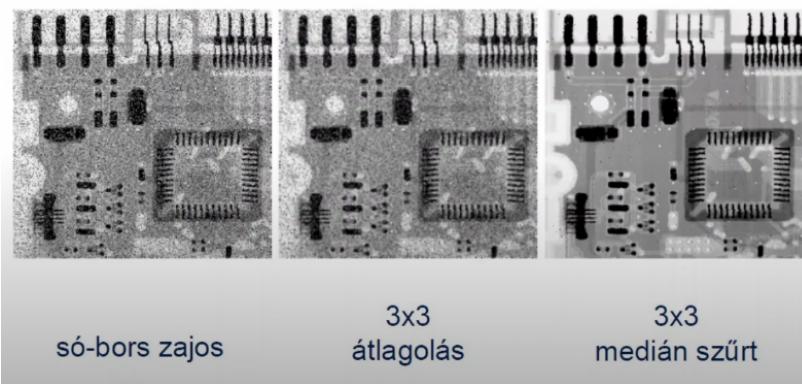
Példa mediánszűrésre



Példa 1D mediánszűrésre



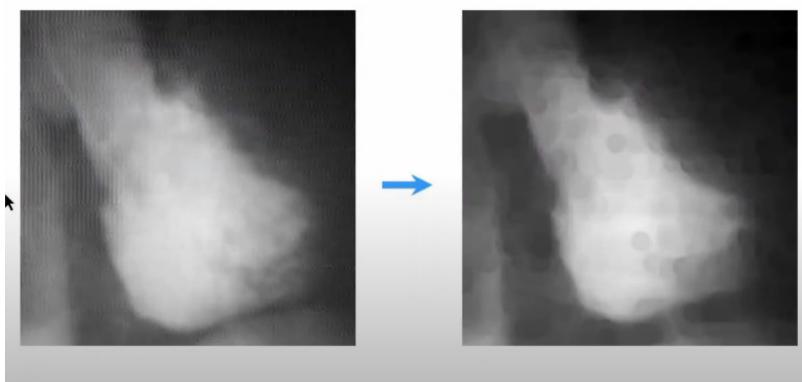
Átlagolás – medián-szűrés



Rank szűrők: Min/max/median (statisztikai jellemzők) szűrők tartoznak ide.

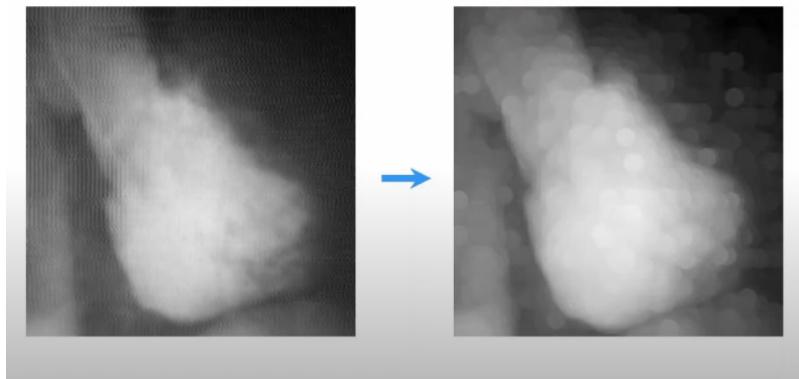
Min szűrő/Erózió: Simítás. Egy adott pixel intenzitása helyettesítődjön az adott képpont S környezetébe eső minimum (legsötétebb) intenzitású pixel intenzitás értékével. Nem szabad macskákra alkalmazni.

Példa erózióra



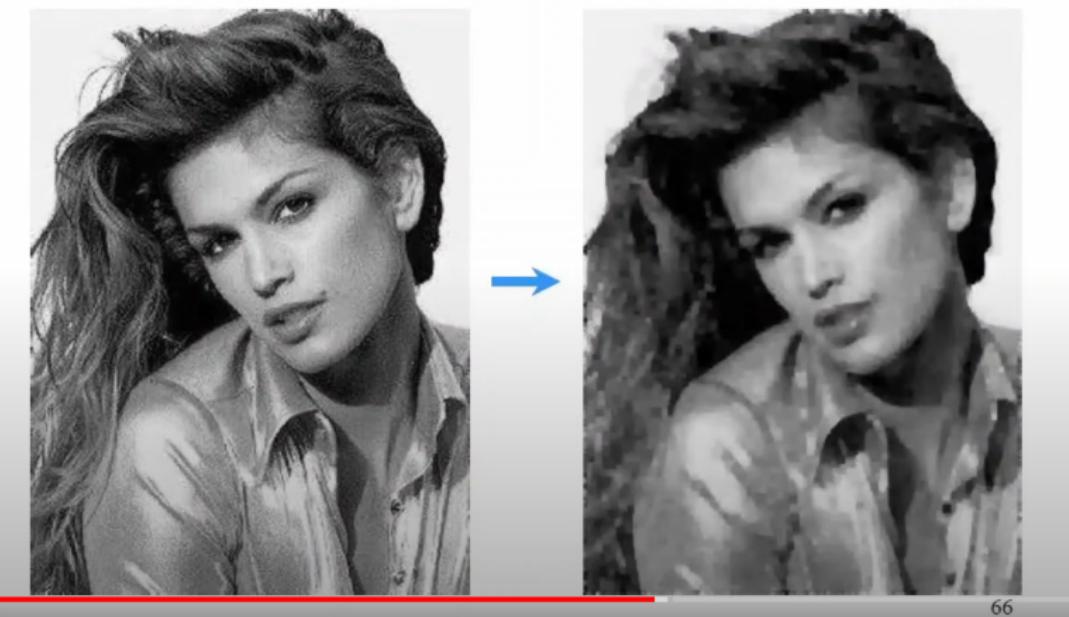
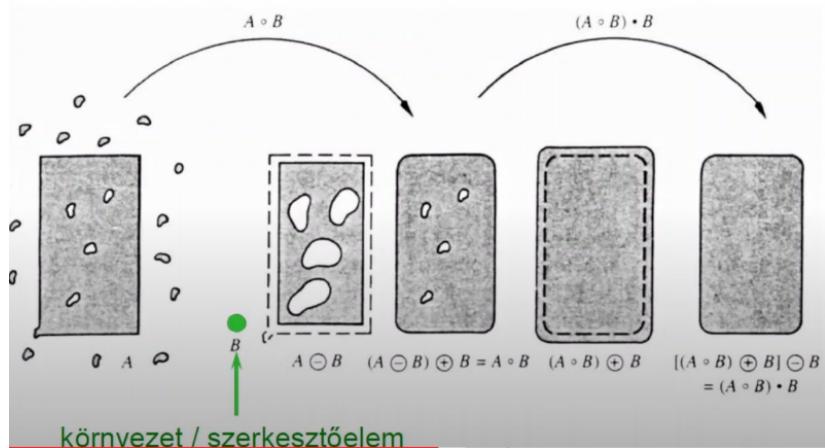
Max szűrő/Dilatáció: Simítás. Egy adott pixel intenzitása helyettesítődjön az adott képpont S környezetébe eső maximum (legvilágosabb) intenzitású pixel intenzitás értékével.

Példa dilatációra



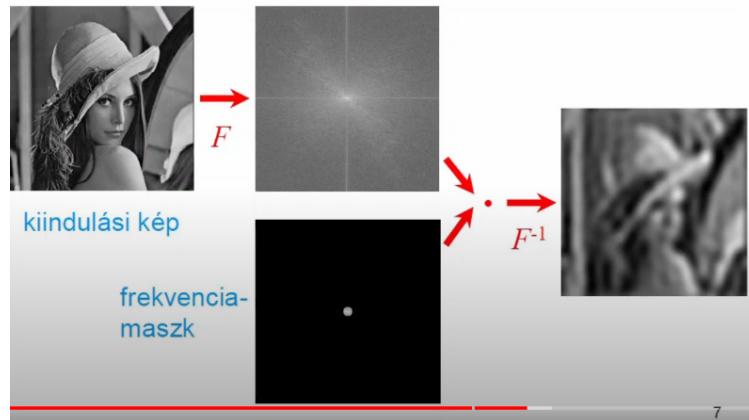
Morfológiai szűrés: Simítás. 4 darab szűrés az alábbi sorrendben: Erózió -> Dilatáció -> Dilatáció -> Erózió.

Példa morfológiai szűrésre



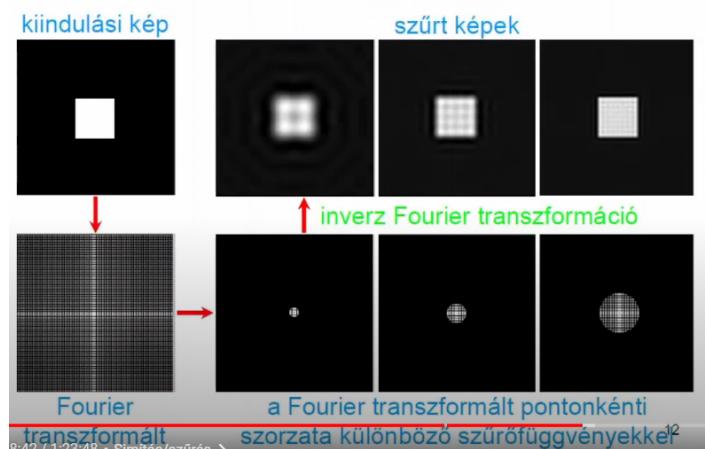
Ideális aluláteresztő szűrő: Erősen simít. Ez egy frekvenciaterben történő simítás. Egy D sugarú körön belül megtartjuk a frekvenciákat, ami azon kívül van azok elnyelődnek. Lényegében ez egy szűrő maszk amivel, ha összeszorozzuk (pontonkénti szorzás) az eredeti képünk Fourier transzformáltját, majd visszatérünk a képtérbe akkor egy homályosított képet kapunk. Zajszűrésre használható. Megjelennek felesleges hullámzások az éles levágás miatt. Minél kisebb a frekvenciamaszk sugara annál homályosabb az eredmény.

Ideális aluláteresztő szűrés

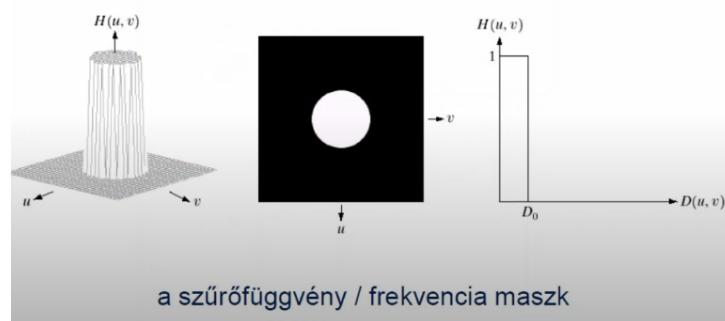


7

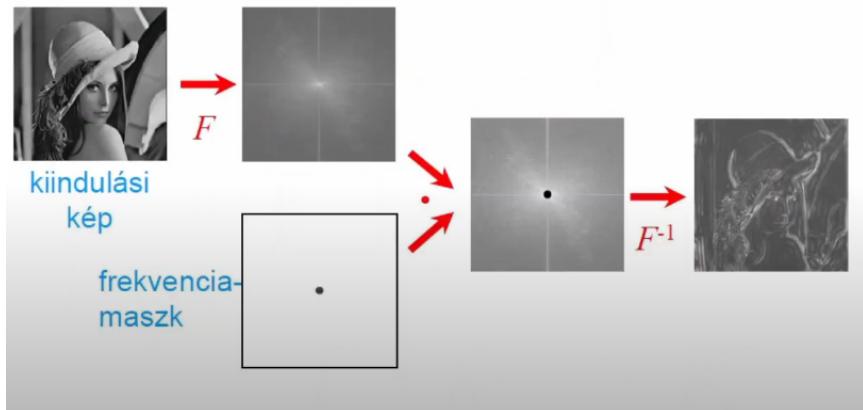
Ideális aluláteresztő szűrő



Ideális aluláteresztő szűrő (Ideal low-pass filter)



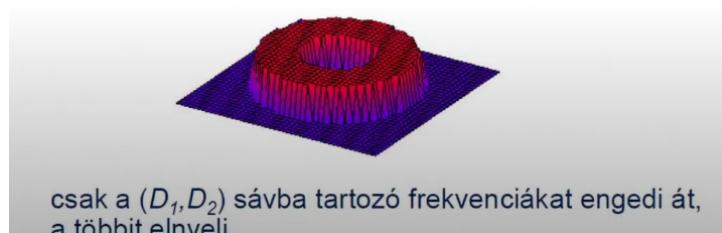
Ideális felüláteresztő szűrő: Aluláteresztő szűrő inverze. Egy D sugarú körön belül elnyelődnek a frekvenciák, amik azon kívül vannak megtartódnak. A magasabb frekvenciák fognak megmaradni. Éleket láthatunk az eredményképen. Hullámosodás előfordul itt is.



Ideális sáváteresztő szűrő: Ebben az esetben csak egy gyűrű alakú tartományba eső frekvenciák kerülnek áteresztésre, a többi elnyelődik.

Ideális sáváteresztő szűrő (Ideal band-pass filter)

$$H_{IBPF}(X, Y) = \begin{cases} 1 & , \text{ha } D_1^2 \leq X^2 + Y^2 \leq D_2^2 \\ 0 & , \text{különben} \end{cases}$$



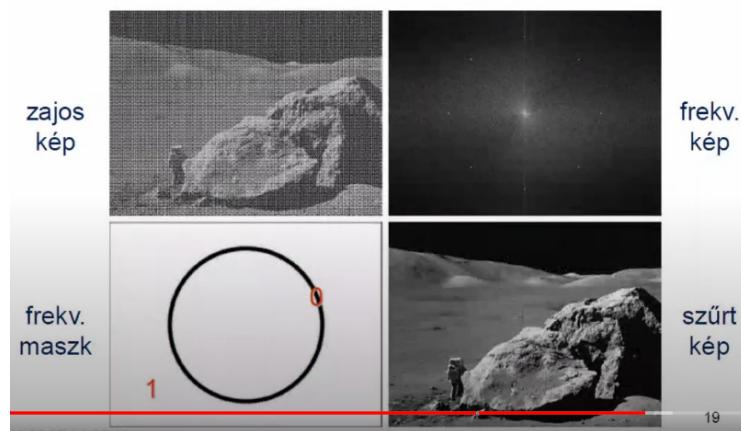
Ideális sávletiltó szűrő: Ebben az esetben egy gyűrű alakú tartományon kívüli frekvenciák kerülnek áteresztésre, a többi elnyelődik. Nagyon hasznos, ha a képen az egyes zajokért pont egy gyűrűbe eső tartomány felelős.

Ideális sávletiltó szűrő (Ideal band-stop/reject filter)

$$H_{IBSF}(X, Y) = \begin{cases} 0 & , \text{ha } D_1^2 \leq X^2 + Y^2 \leq D_2^2 \\ 1 & , \text{különben} \end{cases}$$



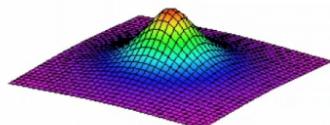
Ideális sávletiltó szűrő



Butterworth-aluláteresztő szűrő: Azért, hogy ne jelenjen meg a szűrt képen hullámosodás, Butterworth kitalálta, hogy ne élesen történjen a szűrés, hanem szépen folytonosan.

Butterworth-aluláteresztő szűrő

$$H_{BLPF}(X, Y) = \frac{1}{1 + \left(\frac{\sqrt{X^2 + Y^2}}{D_0} \right)^{2n}}$$



n : a szűrő rendje

Tulajdonságai:

- nem simít olyan erősen,
- nincs gyűrű hatás (folytonos szűrő),
- élek elmosódnak.

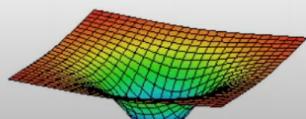


Butterworth-felüláteresztő szűrők

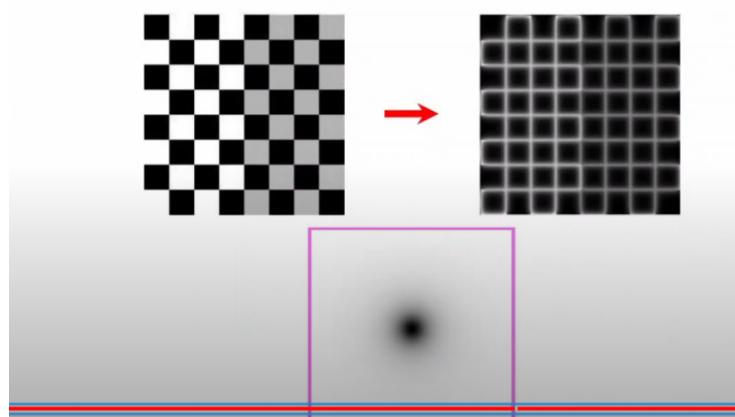
Butterworth-felüláteresztő szűrő: Azért, hogy ne jelenjen meg a szűrt képen hullámosodás, Butterworth kitalálta, hogy ne élesen történjen a szűrés, hanem szépen folytonosan. Feltűnhet, hogy ez hasonló a Gauss szűréshez, viszont a különbség az, hogy más a lecsengéseknek az íve.

Butterworth-felüláteresztő szűrő

$$H_{BHPF}(X, Y) = \frac{1}{1 + \left(\frac{D_0}{\sqrt{X^2 + Y^2}} \right)^{2n}}$$

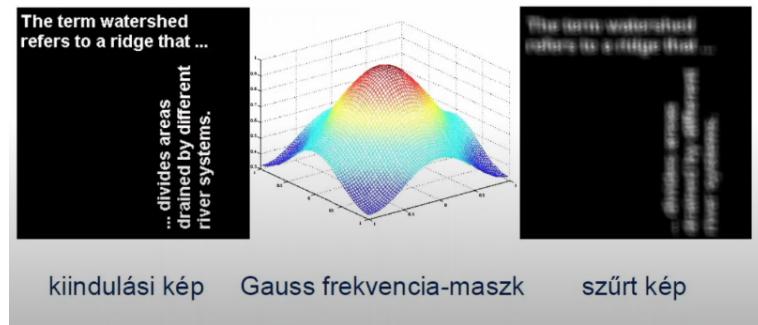


Példa Butterworth-felüláteresztő szűrésre

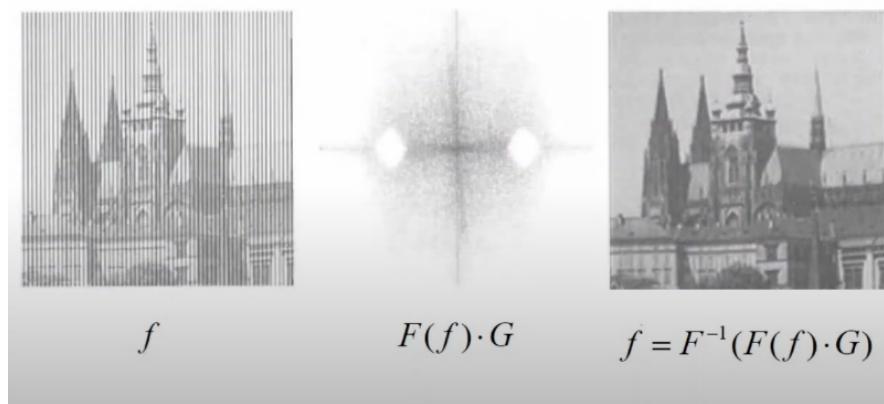


Gauss aluláteresztő szűrő: Fourier transzformáció segítségével gyorsabban elvégezhetők a szűrések, például az aluláteresztő szűrések.

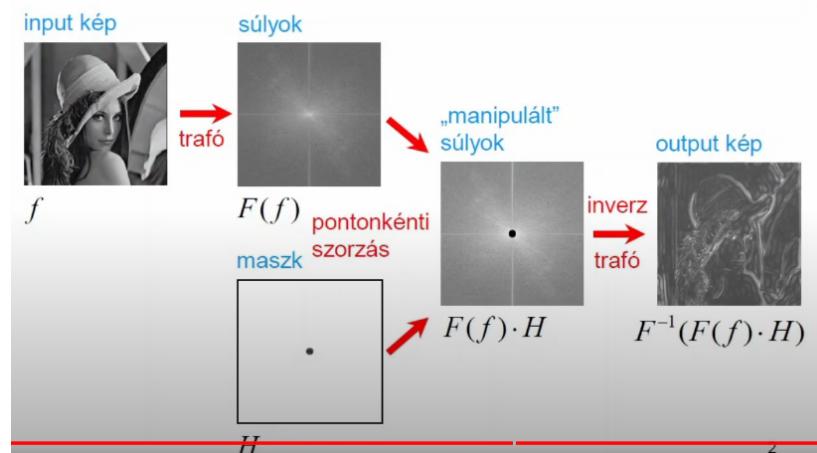
Példa Gauss aluláteresztő szűrésre



Adott frekvenciák szűrése: Frekvenciatérben, ha tudjuk, hogy mely gócpontokhoz mely részlet tartozik a képen, akár különböfél zajok/csíkozások is gond nélkül eltávolíthatóak a képről. Ez képtérben nem menne.



A módszer összegzése



Pont-detektálás: Olyan pontok detektálása, amely a környezetéből kiugrik. Ezeket a pontokat tudjuk detektálni konvolúcióval. A konvolúciós maszk ebben az esetben egy olyan mátrix melynek, elemeinek összege 0 illetve középpött egy kiugró intenzitás körülötte pedig negatív értékek szerepelnek. Ha ezzel a masszal konvolúciót végzünk, akkor egy homogén környezetre ahol nincs kiugró érték 0-át ad, kiugró érték esetén pedig nagy abszolútértéket fog eredményezni.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

(A maszkelemek összege 0.)

Példa pont-detektálásra



Vonal-detektálás: Olyan görbek, melyek intenzitása eltér a környezetétől, kiugró intenzitású görbek. Detektálni az egyes irányokra érzékeny maszkokkal tudjuk. Ezekkel a maszkokkal kell konvolválnunk. Maszkelemek összege 0. Minél nagyobb a maszkunk mérete annál több irányt tudunk megfigyelni.

3x3-as vonal-detektáló maszkol

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$$

0° 45°

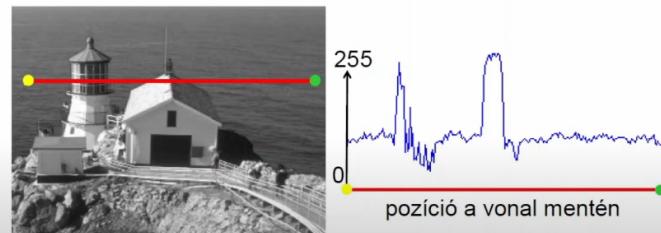
$$\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

90° 135°

(A maszkelemek összege 0.)

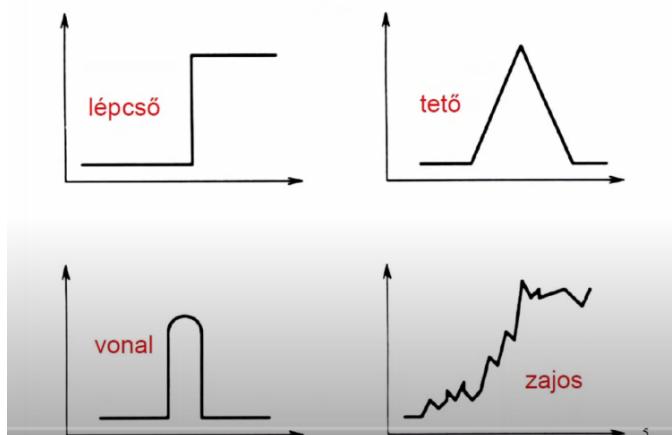
Éldetektálás: Egy képen ott vannak élek, ahol az intenzitás valamelyen irányban felugrik vagy lecsökken. A vonalakat speciális éleknek tekinthetjük. Egy képen ott van él, ahol az intenzitás függvényének deriváltja abszolút értékben nagy.

Él (edge)

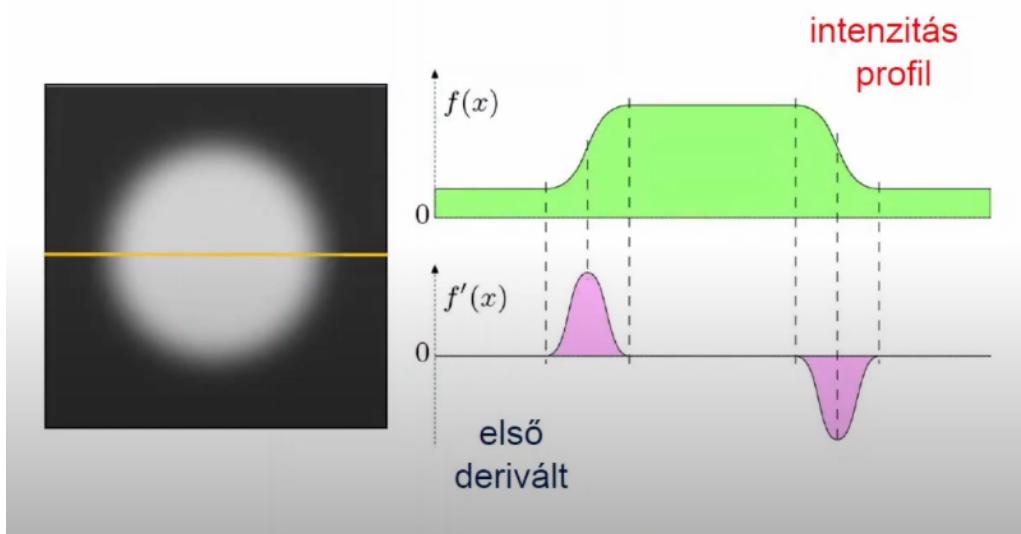


A képen ott található él, ahol a képfüggvény valamely irány mentén hirtelen változik.

Tipikus élprofilok



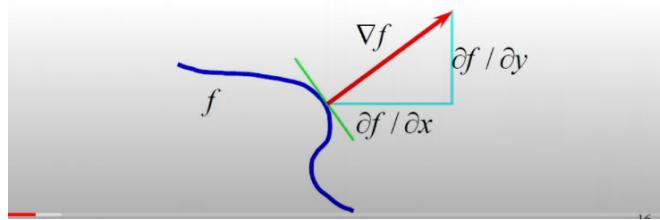
Első derivált



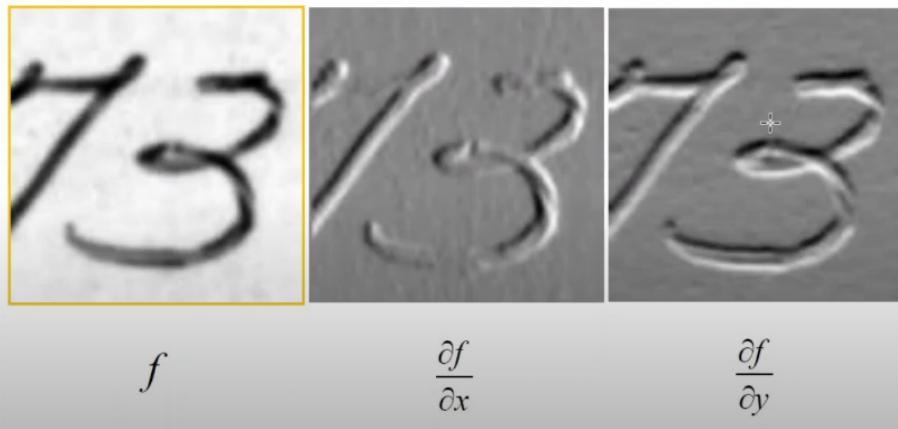
Gradiens: Többváltozós függvényeknél ahány változónk van, annyi első parciális deriváltunk van. Ha ezekből a parciális deriváltakból csinálunk, egy vektort azt nevezzük gradiensnek. Másképp, görbék esetében az érintőre merőleges vektor a gradiens. Az ábrán a **gradiens komponensek** a türkizkékkel jelölt nyílak. Ezeket, ha külön-külön szemléltetjük, akkor domborműves képeket kapunk. Az x az a visszintesben számított gradiens komponens, az y pedig függőlegesen számított. Ezek előjeles képek.

Gradiens (*gradient*)

$$\nabla f = \nabla f(x_1, \dots, x_n) = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T$$



Gradiens komponensek



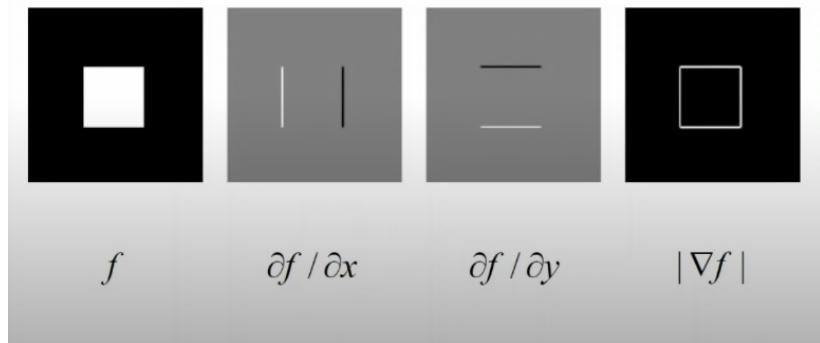
Gradiens magnitúdó: Gradiens nagyságát jelenti. Kiszámítható a két gradiens komponens négyzetösszegének a gyöke, azaz a pitagorasz téTEL kerül alkalmazásra.

Gradiens nagysága (*gradient magnitude*)

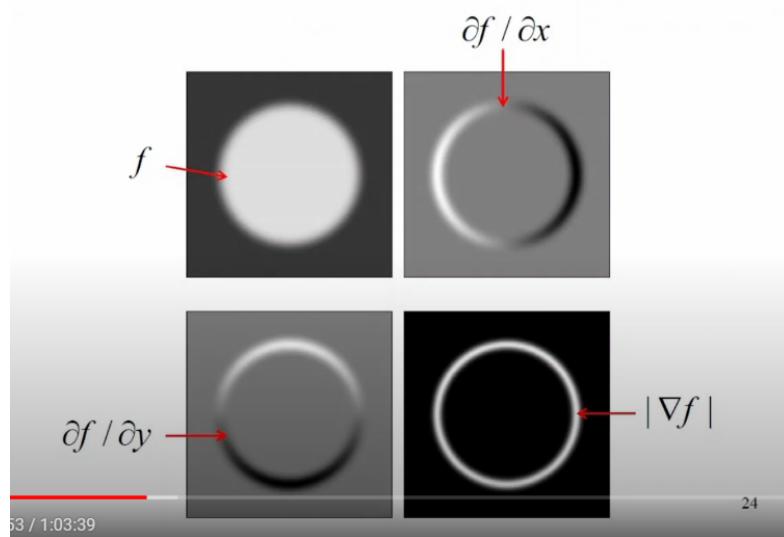
$$\begin{aligned} |\nabla f(x_1, \dots, x_n)| &= \|\nabla f(x_1, \dots, x_n)\|_2 = \sqrt{\sum_{i=1}^n \left(\frac{\partial f}{\partial x_i} \right)^2} \\ &\approx \|\nabla f(x_1, \dots, x_n)\|_1 = \sum_{i=1}^n \left| \frac{\partial f}{\partial x_i} \right| \end{aligned}$$



Gradiens komponensei és nagysága



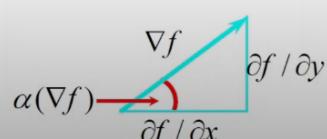
Gradiens komponensei és nagysága



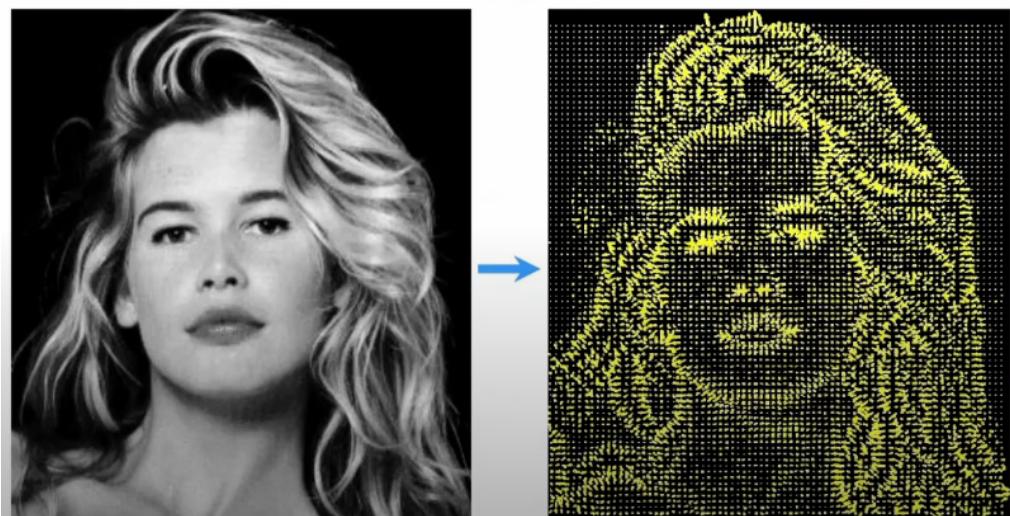
Gardiens iránya: A nagyság és az él ábrázolható képen, ahol nyilak teszik kia kép részleteit. A nyilak élekre mutatnak, és a nyilák hossza az él markánsságát szemlélteti.

Gradiens iránya (gradient direction)

$$\alpha(\nabla f(x, y)) = \arctan \left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right)$$



Gradiens nagysága és iránya^I



Diszkrét gradiens operátorok: Robert, Prewitt, Sobel, Frei-Chen módszerek. Konvolúciós maszkpárok alkalmazó eljárások. Gradiens maszkok tulajdonságai: szimmetrikusak, konstans régiókra nem ad eredményt (maszkelemek összege 0), egyik irányba se húzzunk el jobban.

Gradiens maszk tervezése (x-irányban)

A feltételeket kielégítő maszk:

$$\begin{bmatrix} p & 0 & -p \\ q & 0 & -q \\ p & 0 & -p \end{bmatrix}$$

Prewitt : $p = 1, q = 1$

Sobel : $p = 1, q = 2$

Frei - Chen : $p = 1, q = \sqrt{2}$

Roberts operátor: Két külön konvolúció elvégzése, az egyik az x a másik az y irányból. Ebben az esetben a konvolválás felesleges, egyszerűbben megvalósítható kivonással. Tehát ezt könnyű számolni, de zajérzékeny. Maszkelemek összege 0.

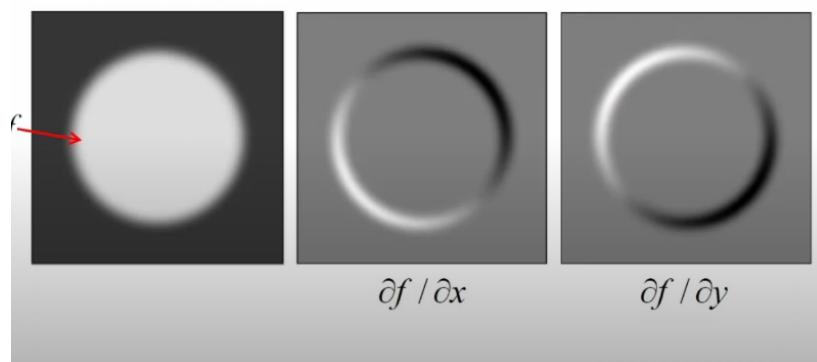
Roberts operátor

$$\left[\begin{array}{c} 0 & \boxed{0 & -1} \\ 0 & \boxed{1 & 0} \\ 0 & 0 & 0 \end{array} \right] \quad \left[\begin{array}{c} \boxed{-1 & 0} \\ \boxed{0 & 1} \\ 0 & 0 \end{array} \right] \quad \begin{array}{l} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{array}$$

(A maszkelemek összege 0.)

30

Roberts operátor



Prewitt operátor: Maszkelemek összege itt is 0. Itt sem célszerű konvolválni, egyszerűbben kiszámítható összeadással és kivonással.

Prewitt operator

$$\left[\begin{array}{ccc} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{array} \right] \quad \left[\begin{array}{ccc} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{array} \right] \quad \begin{array}{l} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{array}$$

(A maszkelemek összege 0.)

33

Sobel operátor: Ez is megoldható konvolválás nélkül, a c-ben a biteltolással (kacsacsőr). Simító hatással bír, mert súlyozott hatással számol.

Sobel operator

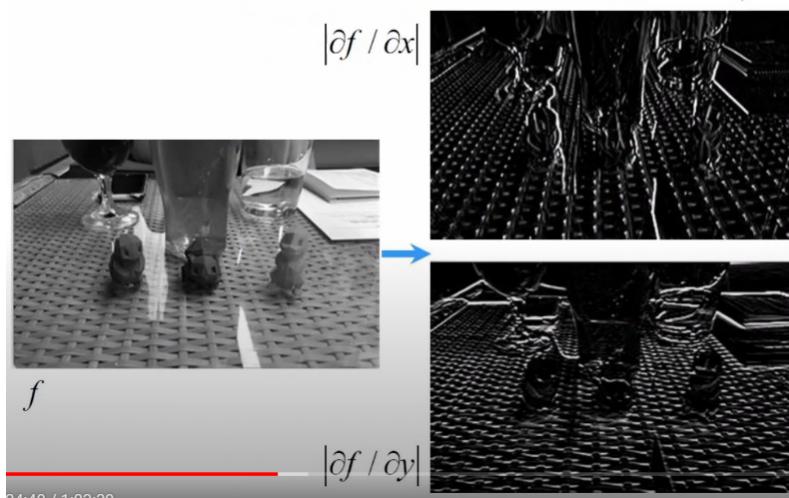
$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\frac{\partial f}{\partial x} \qquad \qquad \qquad \frac{\partial f}{\partial y}$$

(A maszkelemek összege 0.)

35

Sobel operátor



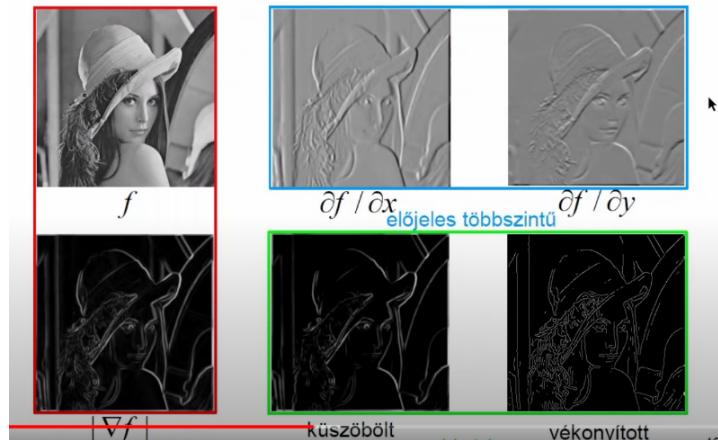
Frei-chen (izotopikus) operátor: Hasonlít a Sobelre annyi különbséggel, hogy itt a maszkokban 2-esek helyett gyök kettesek szerepelnek. Maszkelemek összege 0. Pontosabbak a távolság arányok.



42

Gradiens alapú éldetektálás: Gradiens magnitúdót (piros keret) meghatározzuk a gradiens komponensek (kék keret) segítségével, valamelyik felsorolt módszerrel, majd ezen a képen (piros keret), végrehajthatunk küszöbölést vagy vékonyítást (zöld keret). (Elsőrendű deriváltakkal közelítjük az éleket.)

Gradiens-alapú éldetektálás



8 irányban kereső gardiens (compass) operátorok: Prewitt, Robinson (3), Robinson (5), Kirsh. A maszkelemek összege 0.

Prewitt compass operátor: 3 féle szám szerepel a maszkokban (-1, 2, 1).

Prewitt compass operátor

$$\begin{array}{cccc}
 \begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix} & \begin{bmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
 \text{E} & \text{NE} & \text{N} & \text{NW} \\
 \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & -1 \end{bmatrix} \\
 \text{W} & \text{SW} & \text{S} & \text{SE}
 \end{array}$$

(A maszkelemek összege 0.)

49

Robinson-3 compass operátor: 3 féle szám szerepel a maszkokban (-1, 0, 1).

Robinson-3 compass operátor

$$\begin{array}{cccc}
 \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix} & \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \\
 \text{E} & \text{NE} & \text{N} & \text{NW} \\
 \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix} \\
 \text{W} & \text{SW} & \text{S} & \text{SE}
 \end{array}$$

(A maszkelemek összege 0.)

50

Robinson-5 compass operátor: 5 féle szám szerepel a maszkokban (-2, -1, 0, 1, 2). Robinson maszkjai között fellelhető a Sobel maszkok.

Robinson-5 compass operátor

$$\begin{array}{cccc}
 \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix} & \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} & \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix} \\
 \text{E} & \text{NE} & \text{N} & \text{NW} \\
 \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} & \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix} \\
 \text{W} & \text{SW} & \text{S} & \text{SE}
 \end{array}$$

(A maszkelemek összege 0.)

51

Kirsch compass operátor: 4 féle elem szerepel a maszkokban (-3, 0, 5, 3).

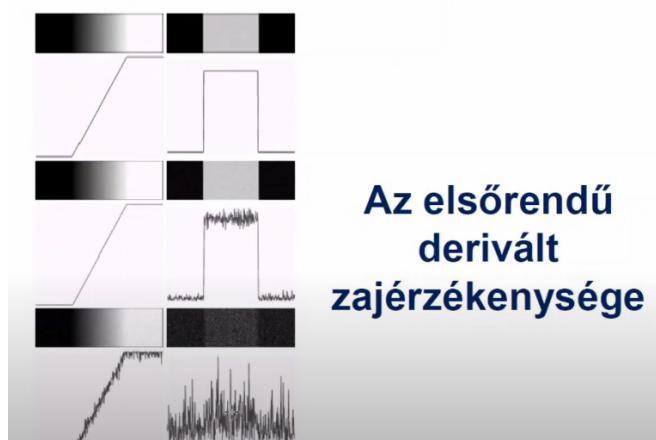
Kirsch compass operátor

$$\begin{array}{cccc}
 \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} & \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} & \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} & \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix} \\
 \text{E} & \text{NE} & \text{N} & \text{NW} \\
 \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} & \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} & \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} & \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \\
 \text{W} & \text{SW} & \text{S} & \text{SE}
 \end{array}$$

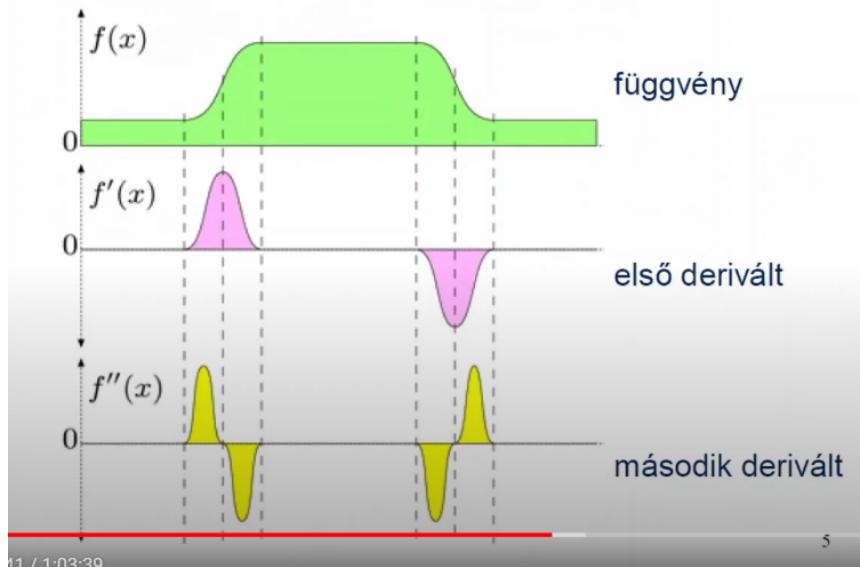
(A maszkelemek összege 0.)

52

Első derivált zajérzékenysége: Az első derivált rendkívül zajérzékeny. Viszont ez kiküszöböltethető másodrendű deriválttal. Másodrendű derivált esetében akkor van él, ha előjelváltás történik a függvényben.



Másodrendű derivált

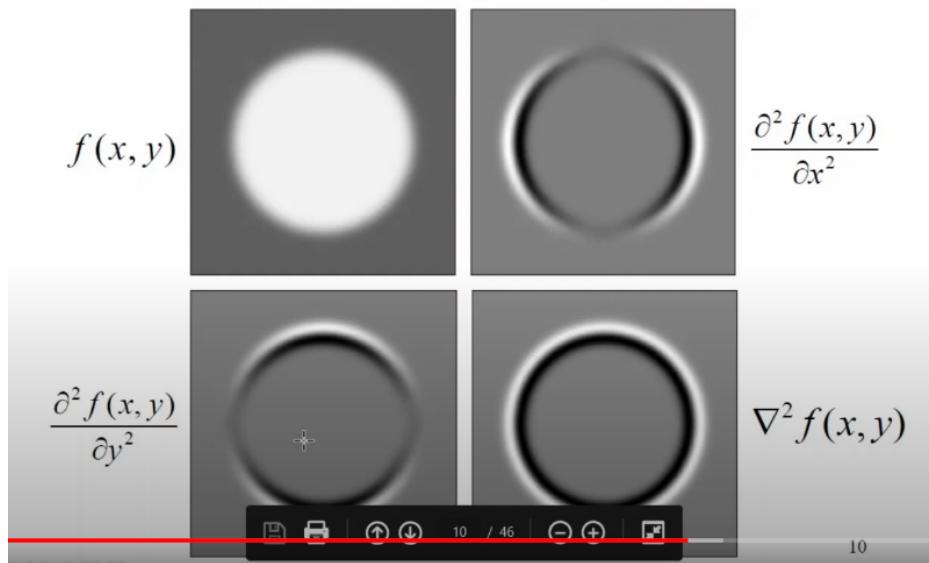


Laplace operátor: Mivel másdorendű deriváltakat diszkrét esetben neház számolni ezért közelítést kell alkalmaznunk. Vegyük az elsőrendű derivált négyzetét, azaz az önmagával való szorzatát. Ekkor egy Nx1 es mátrix kerül összeszorzásra egy 1xN-esel. Ekkor kapunk egy 1x1-es mátrixot amit skalárnak is nevezünk. A Laplace operátor segítségével keressük az előjelváltásokat.

$$\nabla^2 = \nabla \cdot \nabla = \left[\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_n} \right] \cdot \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix} = \sum_{i=1}^n \frac{\partial^2}{\partial x_i^2}$$

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

Példa Laplace operátorra



Egy diszkrét Laplace operátor

$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\approx \frac{\partial^2 f(x,y)}{\partial x^2} \quad \approx \frac{\partial^2 f(x,y)}{\partial y^2} \quad \approx \nabla^2 f(x,y)$$

(A maszkelemek összege 0.)

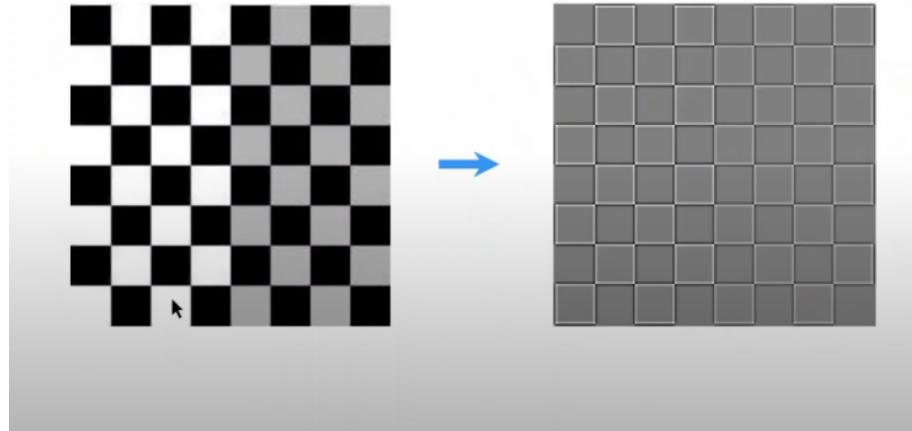
11

További diszkrét Laplace operátorok

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

→

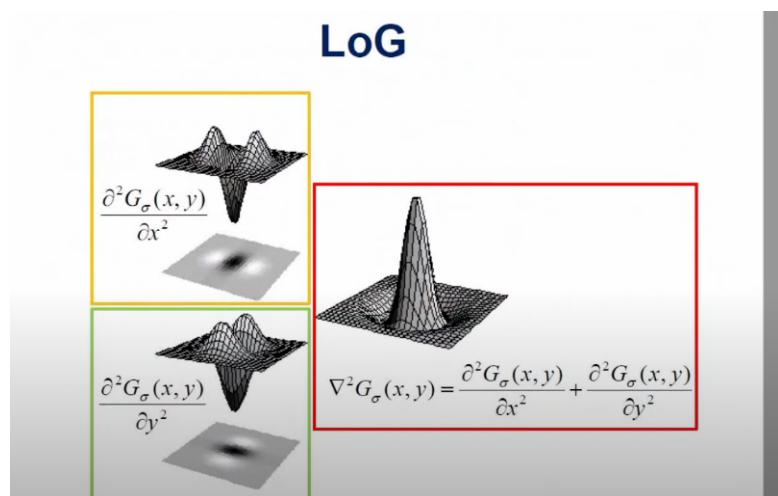
Példa Laplace operátorra



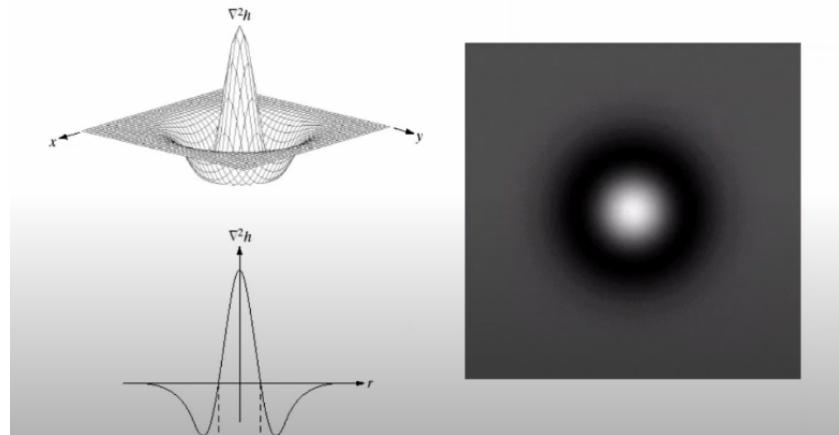
LoG (Gauss függvény Laplace transzformáltja): Mivel a másodrendű derivált is érzékeny a zajra, ezért simítást kell alkalmaznunk. Például a Gauss simítást. minden függvénynek van Laplace transzformáltja, ami a második deriváltaknak az összege.

A Gauss függvény Laplace transzformáltja (LoG – Laplacian of Gaussian)

$$\begin{aligned}\nabla^2 G_\sigma(x, y) &= \frac{\partial^2 G_\sigma(x, y)}{\partial x^2} + \frac{\partial^2 G_\sigma(x, y)}{\partial y^2} \\ &= -\frac{x^2 - \sigma^2}{2\pi\sigma^6} e^{-(x^2+y^2)/2\sigma^2} - \frac{y^2 - \sigma^2}{2\pi\sigma^6} e^{-(x^2+y^2)/2\sigma^2} \\ &= -\frac{x^2 + y^2 - 2\sigma^2}{2\pi\sigma^6} e^{-(x^2+y^2)/2\sigma^2}\end{aligned}$$



LoG (Mexican hat)



LoG diszkrét közelítése:

A LoG diszkrét közelítése – 1.

$$\nabla^2 G \approx \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

Konvolúció LoG függvényel: Lényegében egy simítást követően élet detektálunk, de elvégezhető úgy is hogy először létrehozzuk a „mexican hat”-ünket majd azzal konvolválunk. Ez azért jó, mert így ezeket a sapkákat elég egyszer kiszámolni.

$$\nabla^2(G * f) = (\nabla^2 G) * f$$

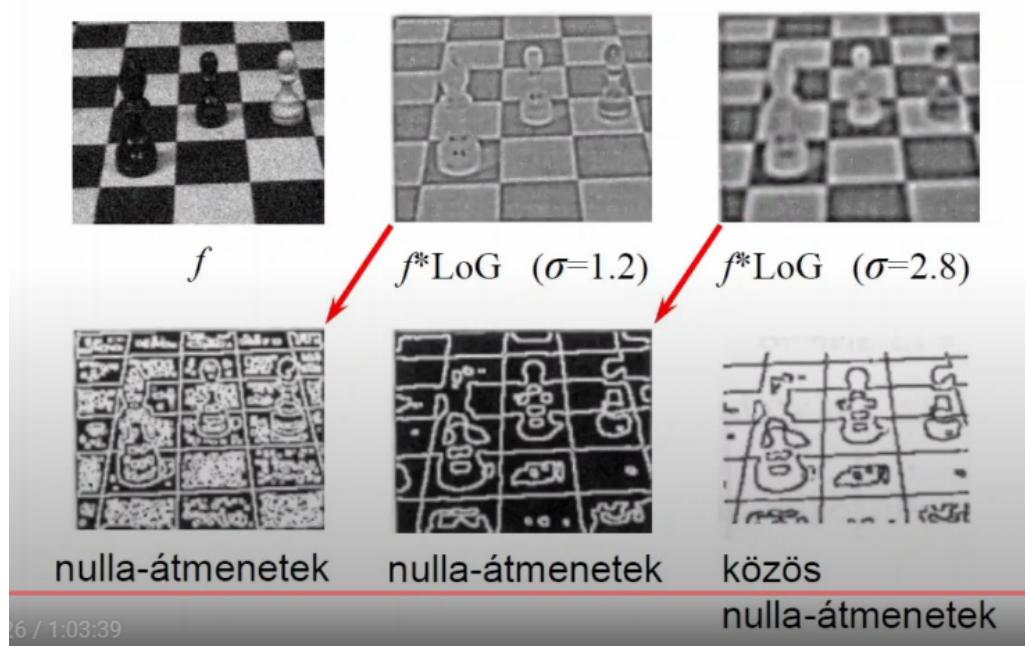
↑ ↑ []
simítás (egyetlen) konvolúció
a simított függvény
Laplace transzformáltja a LoG függvényvel

Konvolúció LoG fügvénnyel



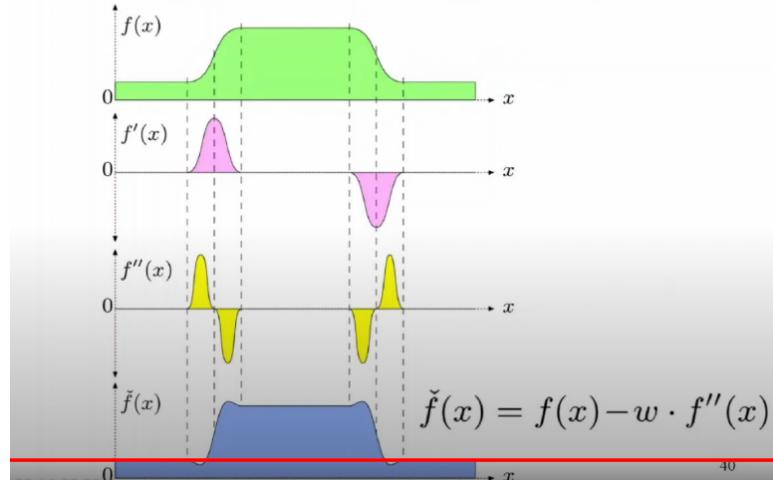
Marr-Hildreth éldetektor: Konvolváljuk a képünket egy vagy több LoG függvénnnyel, majd megkeressük a közös 0 átmeneteket. 0 átmenet ott fog előfordulni ahol egy adott pont kis környezetében (2x2 vagy 3x3) pozitív és negatív értékek is előfordulnak, ezek lesznek az él pontok. Ennek az éldetektornak az outputja egy bináris éltérkép lesz, ahol van 0 átmenet ott 1 ahol nincs, ott 0. Zajos képekre jól működik. Ez a konvolúció gyorsabban kiszámítható a frekvenciatérben.

Példa Marr-Hildreth éldetektorra



Élesítés: Az eredeti képhez adjuk hozzá az éldetektor eredményét. A maszk elemek összege 1.

Élesítés (*sharpening*)

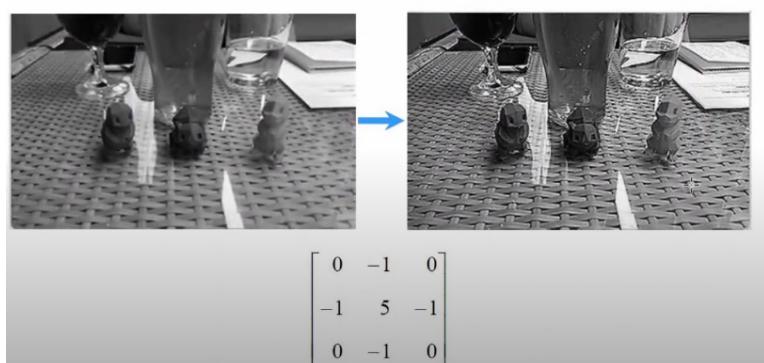


Élesítés egy diszkrét megvalósítása

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$
$$f \approx \nabla^2 f(x, y)$$

(A maszkelemek összege 1.)

Élesítés



Unsharp masking: Simított képet összeadjuk az élképpel.

Unsharp masking

$$I = I + 0$$

$$I = I + I * G - I * G$$

$$I = \underbrace{I * G}_{\text{simított kép}} + \underbrace{(I - I * G)}_{\text{élkép}}$$

$$I' = I * G + \alpha \cdot (I - I * G)$$

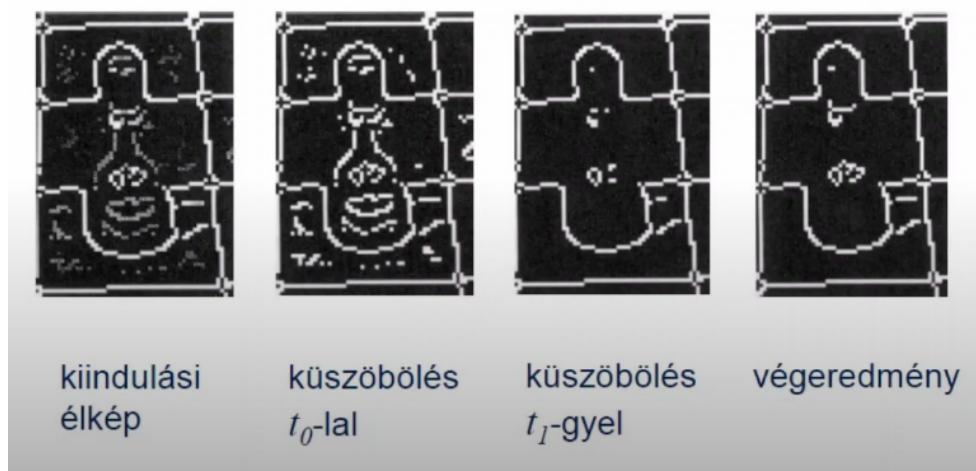
>1 (skalár)

Unsharp masking



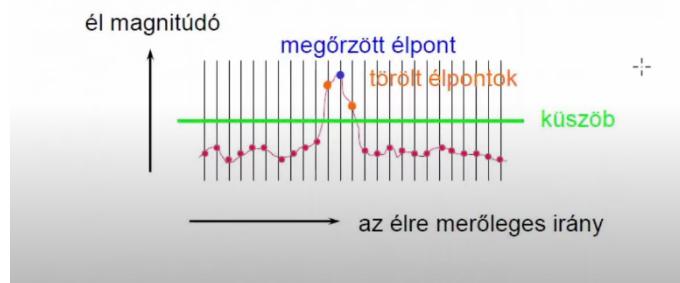
Hiszterézis küszöbölés: Ez a küszöbölés kap inputnak egy már előre éldetektált képet illetve két küszöb értéket, ami egy intervallum lesz $[t_0, t_1]$. Az a gondolat, hogy vizsgáljuk meg az összes képpontot, és ahol az élerősség meghaladja a t_1 -et azt biztosan vegyük be az élek közé. Ha viszont valamelyik pont élerősége nem haladja meg a t_0 -t sem akkor azt ne vegyük be. Ha valamelyik pont a $[t_0, t_1]$ intervallumban van, akkor vizsgáljuk meg azt, hogy van-e szomszédos éle, ami már bekerült az élek közé, ha van, akkor bővítsük az élek halmazát ezzel a ponttal is. Ezt a vizsgálatot addig végezzük amíg el nem fogynak a vizsgálandó pontjaink. Az output egy korrigált bináris kép lesz.

Példa hiszterézis küszöbölésre



Nem maximális élek detektálása: A vastag nem egypont vastag éleket, egy pont vastag élekké összehúzzuk. Az összes él pontot hozzárendeljük valamely irányhoz a 8 fő irány közül. Vizsgáljuk meg minden ponton a pont irányára merőlegesen a két szomszédos pontot, ha van nagyobb intenzitású, mint a vizsgált pontunk akkor biztos, hogy nem az a fő él, jelöljük meg, majd töröljük ki az összes így megjelölt pontot.

Nem-maximális élek elnyomása

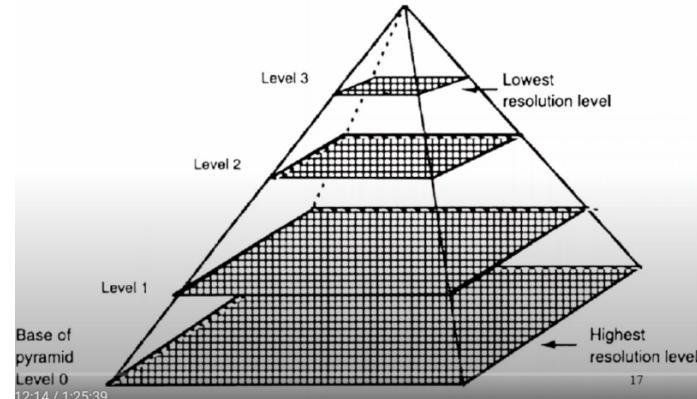


Nem-maximális élek elnyomása



Skálatérelmélet: Fontos a skálázás, annak segítségével is tudunk küszöbölni, ha csökkentjük a kép méretét. Például a **többfelbontású piramis**, ahol folyton felezzük a kép méretét, ekkor minden negyed annyi pixelhez jutunk. Ekkor minél feljebb vagyunk a piramis csúcsa felé, annál gyorsabbak lesznek az algoritmusaink, viszont annál kevesebb képrészlet is fog látszódni. Ha a legalsó szint n akkor felette lévő szintek pixel számának össze $1/3 n$. Ilyen piramisokat **méretcsökkentő simítással** vagy a kép egy-egy blokkjának helyettesítse a blokk mediánjával/átlagával kaphatunk. Méretcsökkentő simítás például a Gauss simítás/piramis.

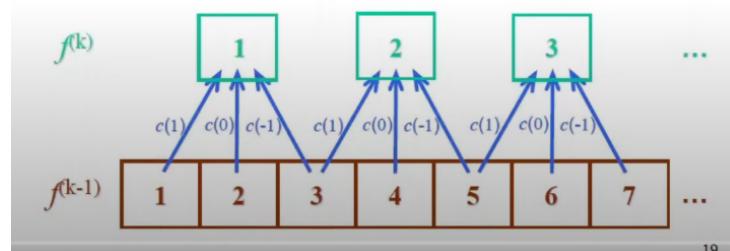
Többfelbontású piramis



Méretcsökkentő simítás

$$f^{(0)}(x) = f(x)$$

$$f^{(k)}(x) = \sum_{n=-\infty}^{\infty} c(n) \cdot f^{(k-1)}(2x-n) \quad (k=1,2,\dots)$$



Gauss piramis



Skála-tér kép: Az egyes frekvenciákat egyre jobban kiegyenesítgetjük. Így elmosódott képet kapunk. Kontúrokra is alkalmazható.

Skála-tér kép (*scale-space image*)

Simitsuk az $f(x)$ függvényt $G(x, \sigma)$ -val egyre növekvő σ értékek mellett. Az

$$F(x, \sigma) = f(x) * G(x, \sigma)$$

függvények által megadott (x, σ) sík a skála - tér kép. Az $F(x, \sigma_0)$ görbe inflexiós pontjai (különböző σ_0 értékekre)

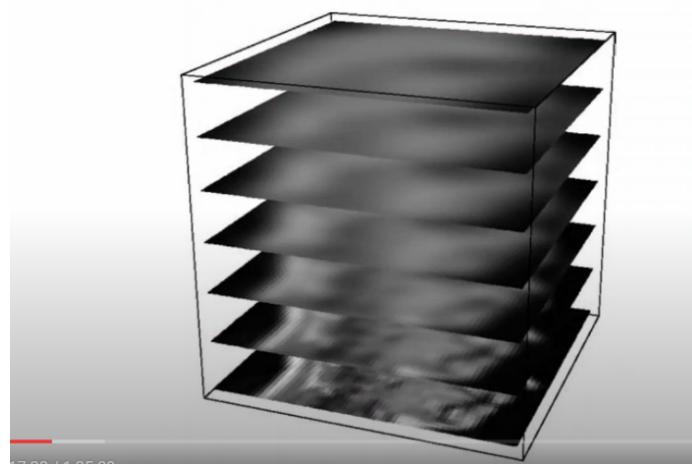
$$\frac{\partial^2 F(x, \sigma_0)}{\partial x^2} = 0, \quad \frac{\partial^3 F(x, \sigma_0)}{\partial x^3} \neq 0$$

jól jellemzik $f(x)$ -et.

2D kép skála-tere



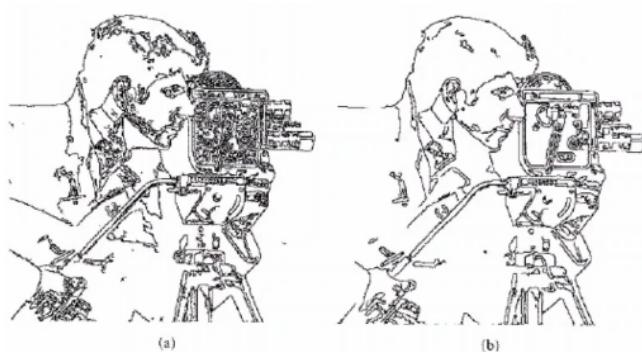
2D kép skála-tere



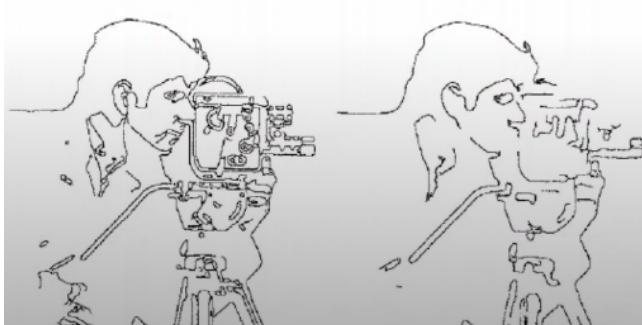
Jellemző szintézis: Készítsünk egy képpiramist, változó simítású képekkel. Majd az éljellemzőket vonjuk ki a képpiramis minden szintjén. Az alsóbb szintek alapján próbáljuk becsülni a felsőbb szinten lévő éleket. Az előre becslés és a konkrét éltérkép között lényeges különbség van, akkor azokat az éleket adjuk hozzá a végső él-reprezentációinkhoz.

Canny féle éldetektálás: Fontos élek ne maradjanak ki, ahol pedig nincs, él ott ne jelezzünk élt, minden él ott legyen, ahol a valójában van, illetve egy él csak egyszer forduljon elő. Használ hiszterézis küszöbölést, nem maximális élek detektálását, skálatérelméletet, jellemző szintézist. **Gauss-simítással** skálatérképet készít, a második deriválton a nulla-átmenetek keresése a legnagyobb első derivált irányá szerint működik. Minél nagyobb a szigma annál nagyobb a simítás anál kevesebb él látszik.

Példa Canny-féle éldetektorra



**Canny-féle
éldetektor**



($[t_0, t_1) = [4, 10)$):

- (a) $\sigma = 0.5$
- (b) $\sigma = 1$
- (c) $\sigma = 2$
- (d) $\sigma = 4$

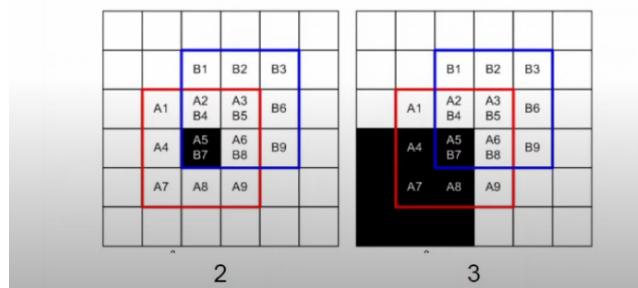
Moravec-féle sarok detektálás: Eltoljuk a képünket u, v –vel majd itt elkezdjük rezegtetni a, b-vel majd az eltolt és rezegtetett képből kivonjuk a rezegtetett képet majd ennek a négyzetét vesszük, minden képpontra. A rezegtetés körbe történik például egy 3x3-as környezetben. Azaz készítünk még 8 másik képet, tehát az eltolt képpel együtt van 9 képünk. Ezek közül vesszük pontonként a minimumot, ez lesz a sarkosság mérték kép. A sarkosság márték képre használjuk a nem-maximális értékek elnyomását, és megkapjuk a sarkosság-térkép képet. Élek mentén és konstans területen nem jelez be. Viszont a sarkoknál és nagy intenzitású pontoknál, amik egyedül vannak ott igen. Nem tökéletes, de gyors.

Moravec-féle sarok-detektor

Intenzitás-ingadozás (u, v) eltolás mellett:

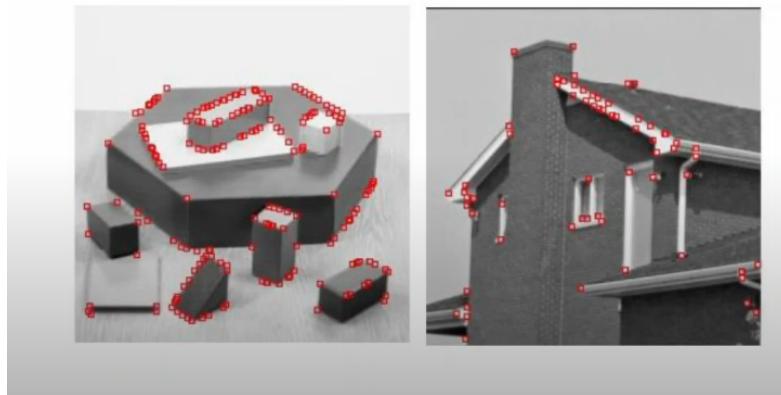
Moravec-féle sarok-detektor

Az intenzitás-ingadozás jobbra és felfelé tolás esetén két bináris képen:



Moravec-féle sarok-detektor

Moravec-féle sarok-detektor



Harris/Plessey féle sarok-detektor: Ugyan úgy elolt képeket gyártunk, viszont használjuk hozzá a gradiens komponenseket. Felírható mátrix vektor szorzásban. Ezeknek a mátrixoknak keressük meg a sajátvektorait és a sajátértékeit. Ahol a két sajátérték közel 0 ott belső pont van, ahol az első sajátérték jóval nagyobb vagy jóval kisebb, mint a második az élpont, ha viszont mind a kettő nagy, akkor ott sarok pont van. Ezután határozzuk meg a sarkosság mérték képet, majd küszöböléssel és nem maximális értékek elnyomásával határozzuk meg a sarkosság-térképet. Sokkal jobb eredmény.

Harris/Plessey sarok-detektor

Intenzitás-ingadozás (u, v) eltolás mellett:

$$V_{u,v}(x, y) = \sum_{i \in S(x,y)} w_i \left(u \frac{\partial I_i}{\partial x} + v \frac{\partial I_i}{\partial y} \right)^2$$

(x, y) középponttú Gauss-szűrő eleme
ablak az i pozícióban

Harris/Plessey sarok-detektor

$$V_{u,v}(x, y) = [u \quad v] M \begin{bmatrix} u \\ v \end{bmatrix},$$

ahol :

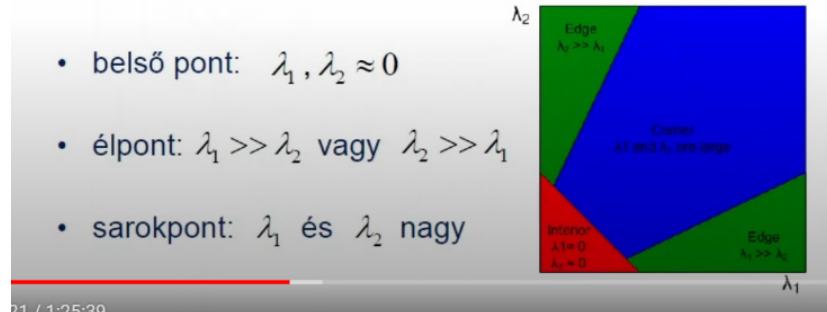
$$M = \begin{bmatrix} \frac{\partial I^2}{\partial x^2} * w & \frac{\partial I^2}{\partial x \partial y} * w \\ \frac{\partial I^2}{\partial x \partial y} * w & \frac{\partial I^2}{\partial y^2} * w \end{bmatrix}$$

Harris/Plessey sarok-detektor

Az M mátrix λ_1 és λ_2 sajátértékei, vagyis az

$$M \cdot \underline{v} = \lambda \cdot \underline{v}$$

egyenlet megoldásai alapján osztályozható az (x,y) pont:



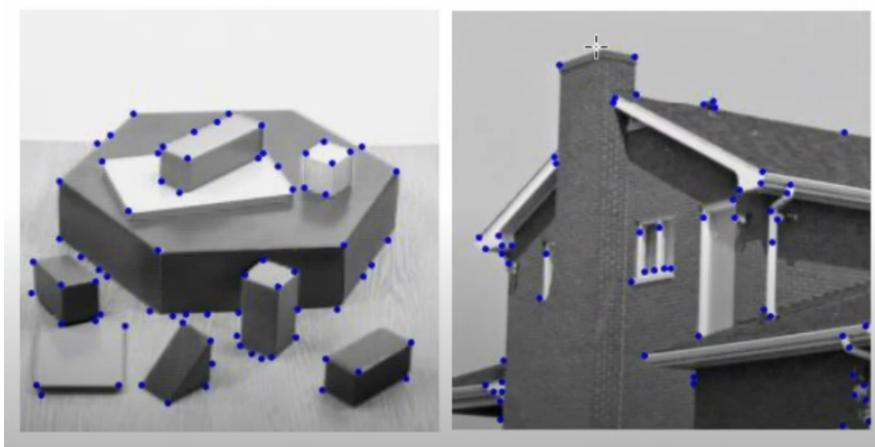
Harris/Plessey sarok-detektor

A Harris és Stephens által javasolt „sarkossági mérték”:

$$\begin{aligned} C(x, y) &= \det(M) - k(\text{trace}(M))^2 \\ &= \lambda_1 \cdot \lambda_2 - k(\lambda_1 + \lambda_2)^2 \end{aligned}$$

A gyakorlati vizsgálatok alapján a $k \in [0.04, 0.06]$ értékek szolgáltatnak jó eredményt.

Harris/Plessey sarok-detektor



Szegmentálás: Címkézéssel vagy kontúrozással. A **hierarchikus szegmentálás**, címkézéses szegmentálást végzünk egyre durvább részletességgel. Szegmentálás során bármely kettő szegmensnek nem lehet közös része, illetve minden képpont kerüljön be egy szegmensbe. minden régió összefüggő legyen, lyukmentes. Szomszédos régiók jól elkülöníthetőek legyenek.

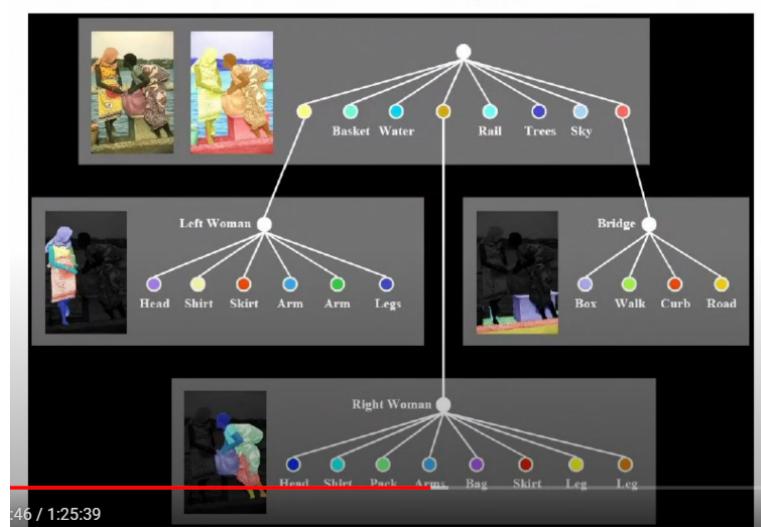
Példa szegmentációra



Példa szegmentációra



Hierarchikus szegmentáció



A szegmentálás formálisan

R : a teljes kép,

P : az összetartozás predikátuma,

R_1, R_2, \dots, R_n : az R szegmensei P szerint,

ahol:

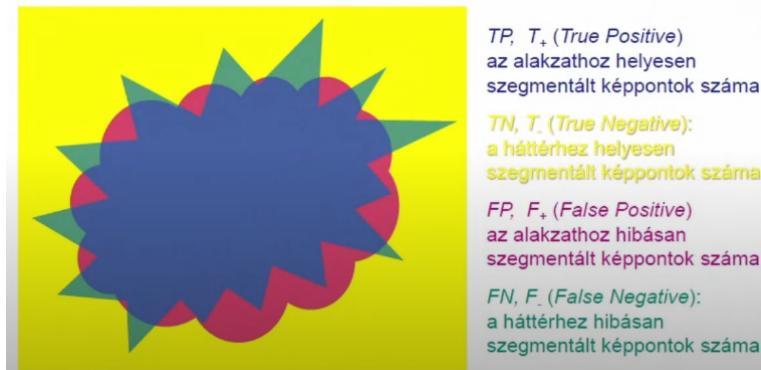
- $\bigcup_{i=1}^n R_i = R$, $R_i \cap R_j = \emptyset$,
- R_i összefüggő,
- $P(R_i) = 1$ (igaz),
- $\underline{P(R_i \cup R_j) = 0}$ (hamis) $(1 \leq i \neq j \leq n)$.

5:39

20

Szegmentálás jóságának mérése:

A szegmentálás jóságának mérése



Mérőszámok – 1.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$sensitivity = \frac{TP}{TP + FN}$$

$$specificity = \frac{TN}{TN + FP}$$

a sensitivity további elnevezései: *recall* vagy *TPr* (*True Positive rate*)

Validáció

		Predicted condition	
		Total population	Predicted Condition positive Predicted Condition negative
True condition	condition positive	True positive	False Negative (Type II error)
	condition negative	False Positive (Type I error)	True negative
confusion matrix			

Szegmentálás küszöböléssel: A küszöbölés függhet a helytől, a környezettől és a régi színtől. Veszünk egy T küszöböt, T-nél sötétebb képpontot a hátérhez soroljuk, a többit pedig az objektumhoz. Intervallumosan is küszöbölhetünk. A küszöbérték belőhető próbálkozással.

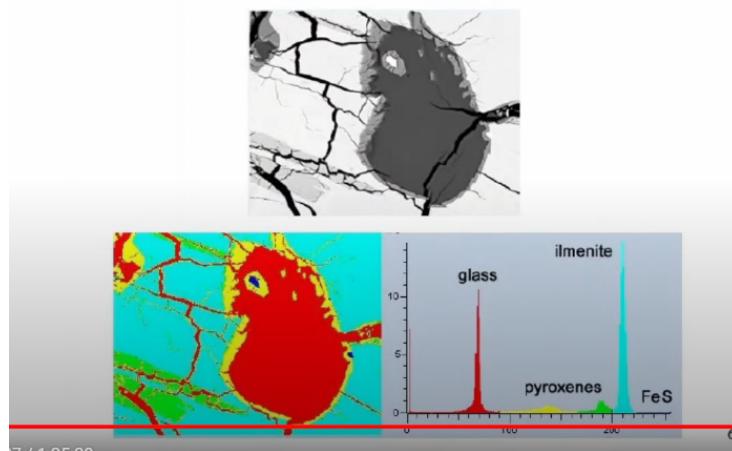
A küszöbölés típusai

Globális: $T = T\{f(x,y)\}$

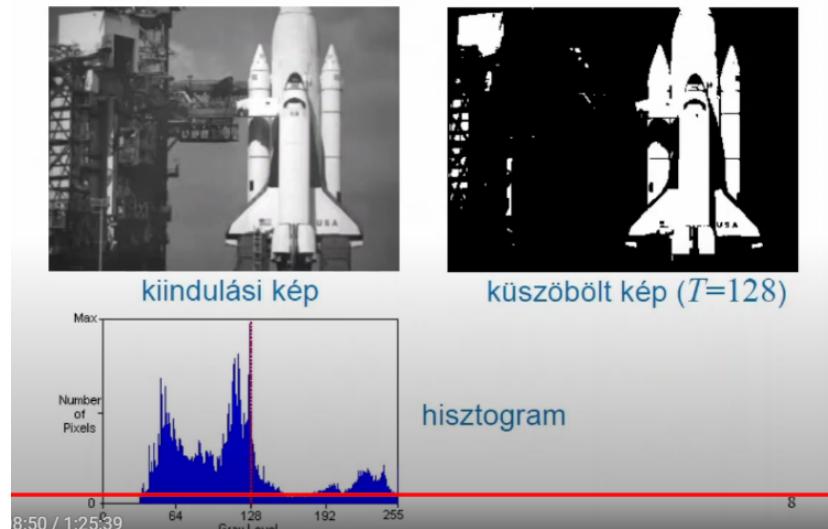
Lokális: $T = T\{A(x,y), f(x,y)\}$

Dinamikus: $T = T\{x, y, A(x,y), f(x,y)\}$

Szegmentálás küszöböléssel



Példa egyszerű küszöbölésre



Többsávos küszöbölés

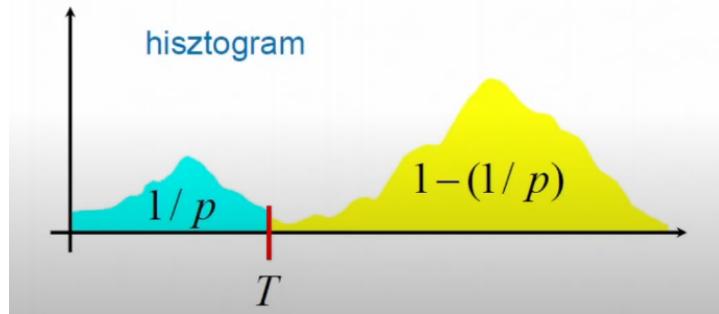
$$g(x, y) = \begin{cases} v_1, & \text{ha } f(x, y) \in D_1 \\ v_2, & \text{ha } f(x, y) \in D_2 \\ \dots & \\ v_k, & \text{ha } f(x, y) \in D_k \\ v_d, & \text{különben} \end{cases}$$

Köszöbérték meghatározásá:

- p-edrésznel
- adaptív
- dinamikus (lokális)
- átlag és szórás alapján
- konvencionális
- optimális
- Otsu

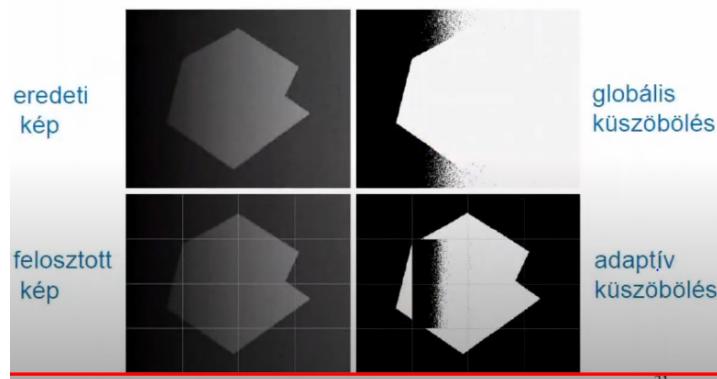
Küszöbölés p-edresznéllel: Akkor alkalmazzuk, ha tudjuk a képen szereplő objektumok eloszlásának arányát. Például a nyomtatott lapon 1/10 arányban van írás.

Küszöbölés p-edresznél



Adaptív küszöbölés: Osszuk fel a képet, egymást nem fedő részképekre, és mindegyikre lokális küszöböt határozzunk meg a többi képrészlettől függetlenül. Ahol nem találunk küszöbot/üres területeknél ott a környező képrészletek küszöbértékéből számítsuk ki, mondjuk valamilyen átlagolással.

Példa adaptív küszöbölésre



Dinamikus (lokális) küszöbölés: minden egyes képpont saját küszöbbel rendelkezik. Az adott pont és a valamelyen méretű környezetében számított átlag intenzitásból, vonjunk ki egy konstans értéket. Ez lesz a küszöb.

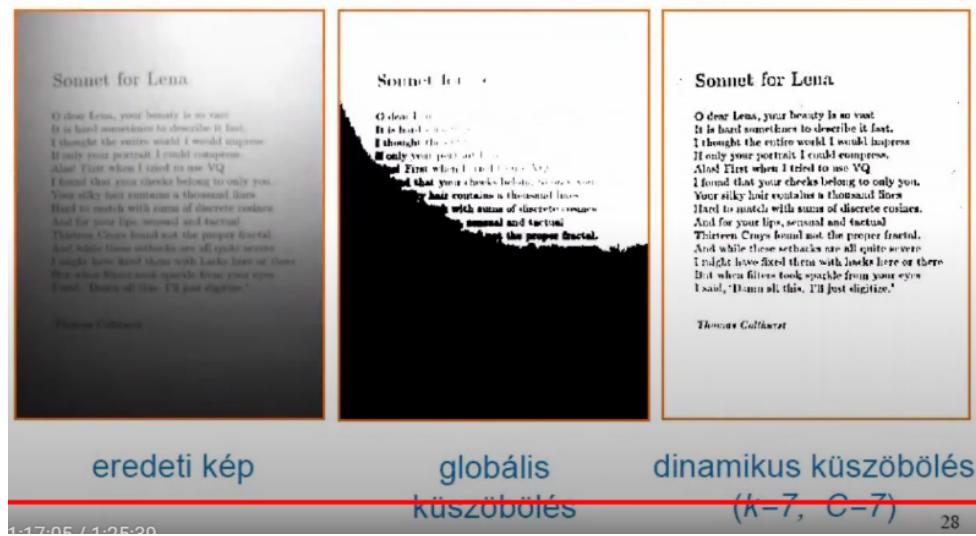
Dinamikus (lokális) küszöbölés

$$T(x,y) = ablak_átlag(x,y) - C,$$

ahol

- *ablak_átlag(x,y)* az (x,y) pont $k \times k$ -as környezetében számított átlag
- C : konstans.

Példa dinamikus küszöbölésre



Küszöbértékek választása átlag és szórás alapján:

Küszöbérték választása átlag és szórás alapján

$$T = k_1 \cdot \mu + k_2 \cdot \sigma,$$

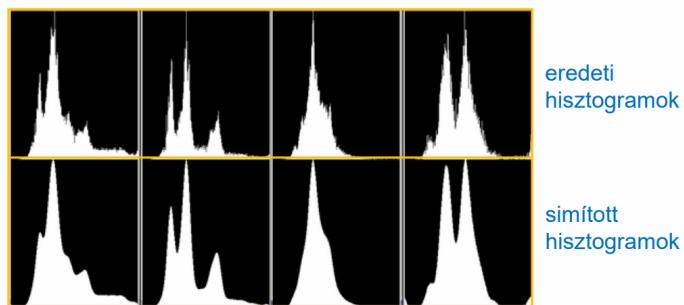
ahol

$$\sigma = \sqrt{\frac{1}{m \cdot n} \sum_{i=1}^m \sum_{j=1}^n (f(i, j) - \mu)^2}$$

k_1 és k_2 képtől függő konstansok.

Hisztogram simítás

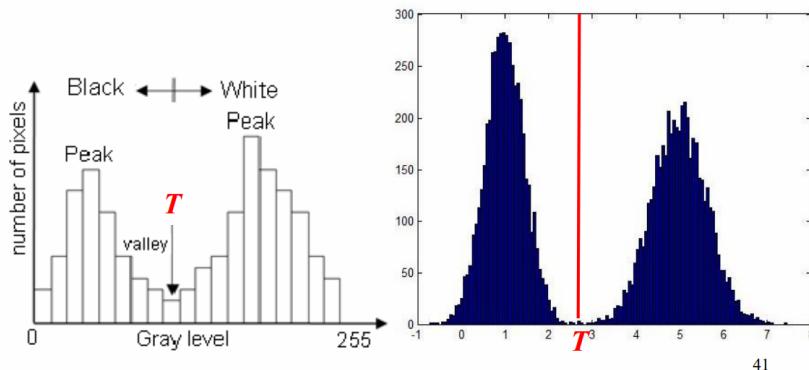
$$p'(z) = \sum_{i=-k}^k w_i \cdot p(z+i), \quad \sum_{i=-k}^k w_i = 1$$



Konvencionális küszöbölés: Keressük meg a két maximum intenzitást, majd a maximumok közötti minimumot. Ez lesz a küszöb. Előnyös, ha a hisztogramunk **bimodális** azaz két csúcsú.

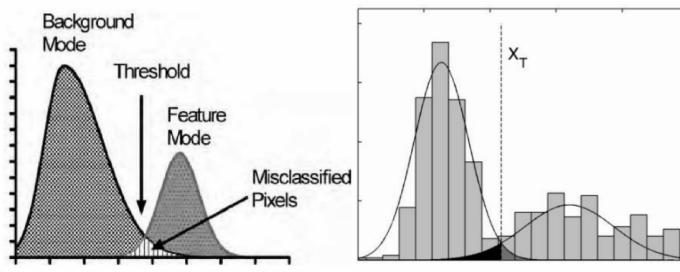
Konvencionális küszöbölés

A küszöb a maximumok közötti minimum.



Optimális küszöbölés: Olyan, mint a konvencionális, csak a két maximális érték között úgy válasszuk meg a minimum küszöböt, hogy a szegmentálás hibája minimális legyen.

Optimális küszöbölés



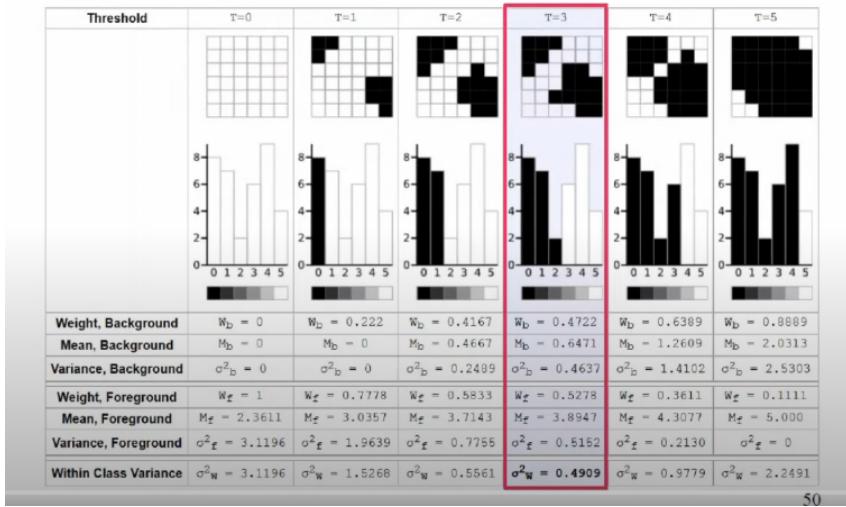
Otsu módszer küszöbölésre: Végig veszi az összes lehetséges küszöböt, és az eredményekből választja a minimálisat. Átlagot és varianciát használ.

Otsu módszere küszöbölésre

Egy olyan küszöbértéket keres, amire minimális az osztályon belüli variancia (*intra-class variance*):

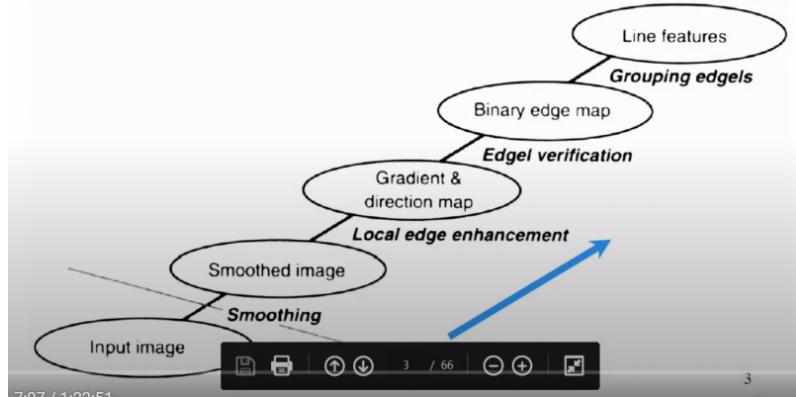
$$\sigma_w^2 = W_b \cdot \sigma_b^2 + W_f \cdot \sigma_f^2$$

Otsu módszere küszöbölésre



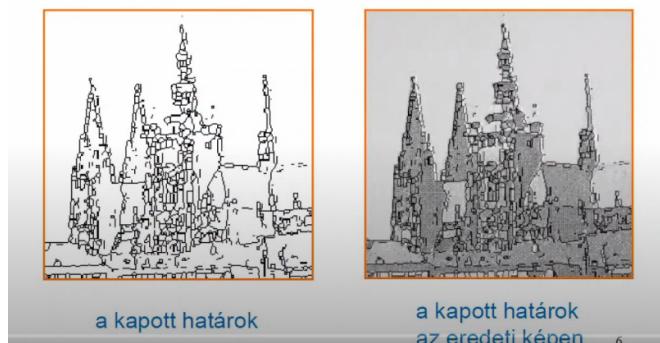
Él alapú szegmentálás:

Egy él-alapú szegmentáló módszer modellje



Él-relaxáció: Iteratívan újraértekeljük, a képpontok osztályozását. Egy élnek, ha nincs folytatása, akkor valószínűleg nem tartozik semmilyen határhoz, illetve ha egy gyenge él két erős él között van, akkor valószínűleg a határvonalhoz tartozik.

Példa él-relaxációra



Élkapcsolás: Ha szomszédos él pontoknak hasonló tulajdonságaik vannak, akkor kapcsoljuk össze őket. Ha a nagyságuk vagy az irányuk között kicsi a különbség kapcsoljuk őket össze.

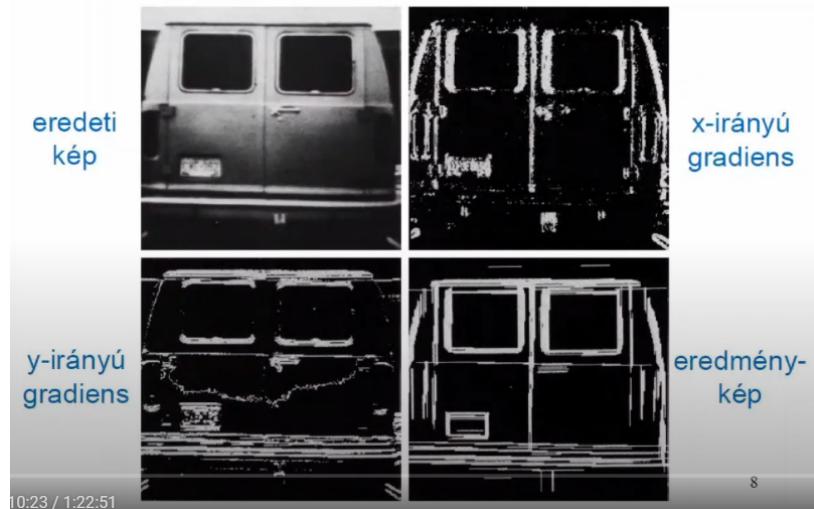
1. Hasonló a gradiensük nagysága valamely küszöbértékre:

$$\|\nabla f(x_1, y_1) - \nabla f(x_2, y_2)\| \leq T_m$$

2. Hasonló gradiensük irása valamely küszöbértékre:

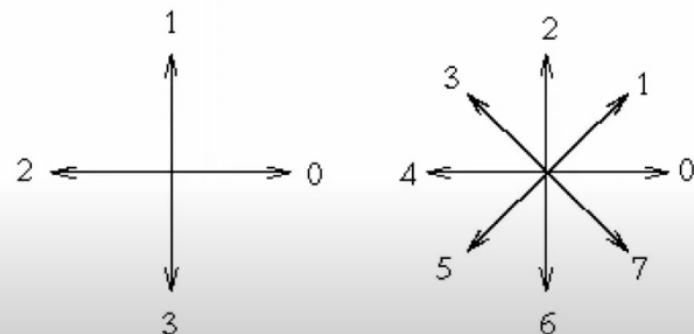
$$|\alpha(\nabla f(x_1, y_1)) - \alpha(\nabla f(x_2, y_2))| \leq T_\alpha$$

Példa él-kapcsolásra



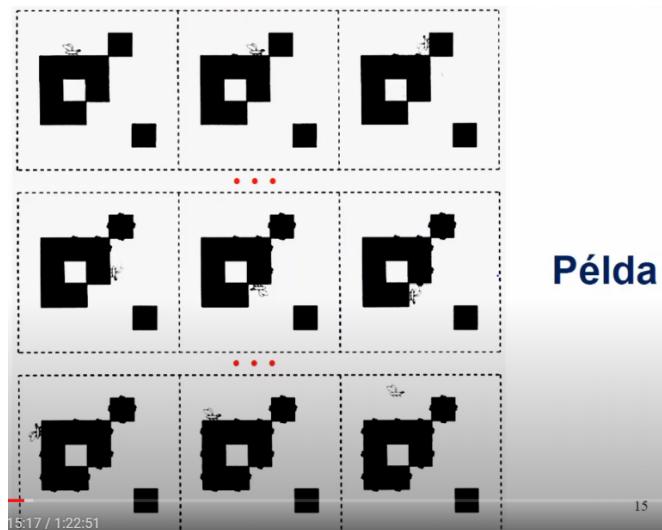
Határkövetés bejárási irányokkal:

Bejárási irányok határkövetéshez



... a 4- és a 8-szomszédság/összefüggőség esetén

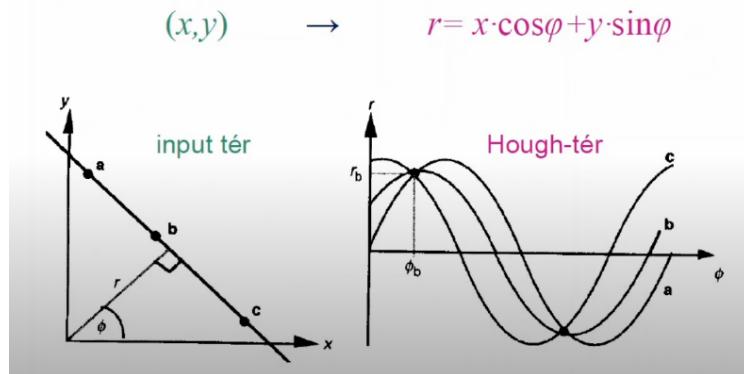
Házi légy algoritmus: Kontúr bejárás.



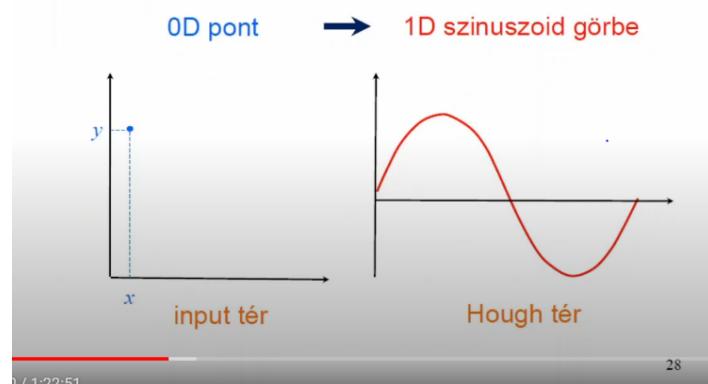
Többszintű képeken határkövetés: Nem csak bináris éldetektált képeken tudunk kontúrokat keresni. A gráfelmélet segítségével, ha két érdekesebb él pont ha szomszédos, akkor a gráfban futassunk közöttük élt. Tehát az érdekes él pontok a csúcsok a gráfban a közöttük futó él pedig azt jelenti, hogy szomszédosak e. Az élekhez rendeljünk költséget, úgy hogy ha a két pont intenzitása közel van egymáshoz, akkor kicsi legyen a költség.



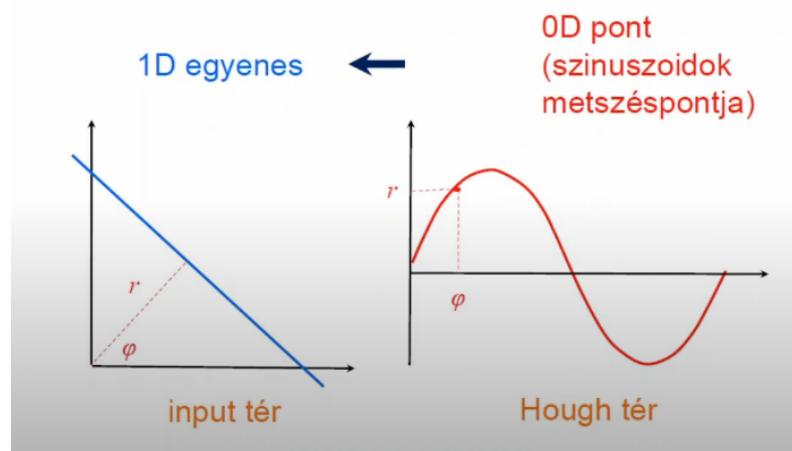
Hough transzformáció: A sima térbeli pontok a Hough-térben szinuszoid görbéknek felelnek meg. Ahol ezek a görbék metszik egymást, ott lesz nekünk egy egyenesünk. A Hough-térben ennek a pontnak lesz egy x és egy y tulajdonsága. Az x tengelyen a szöget kapjuk meg, hogy az egyenesre vett derékszög az origóra milyen szöget zár be. Az y tengelyen pedig az előbb említett egyenesre vett merőleges egyenes hosszát mutatja meg.



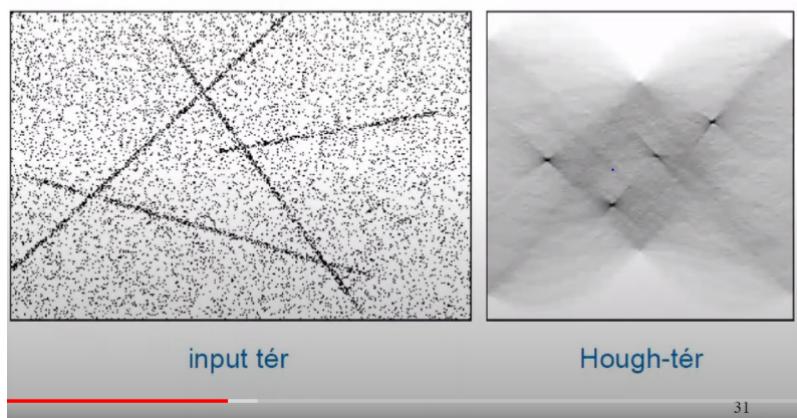
Egyenes detektálása



Egyenes detektálása

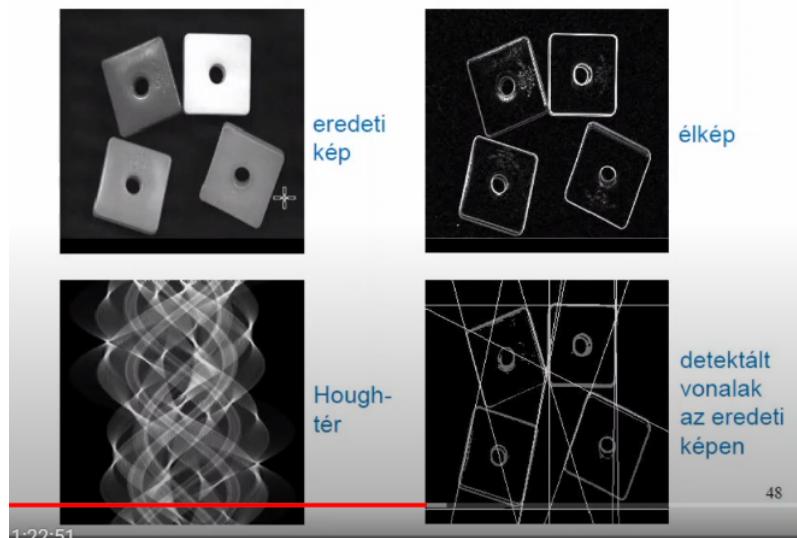


Egyenes detektálása



31

Példa egyenes detektálására



48

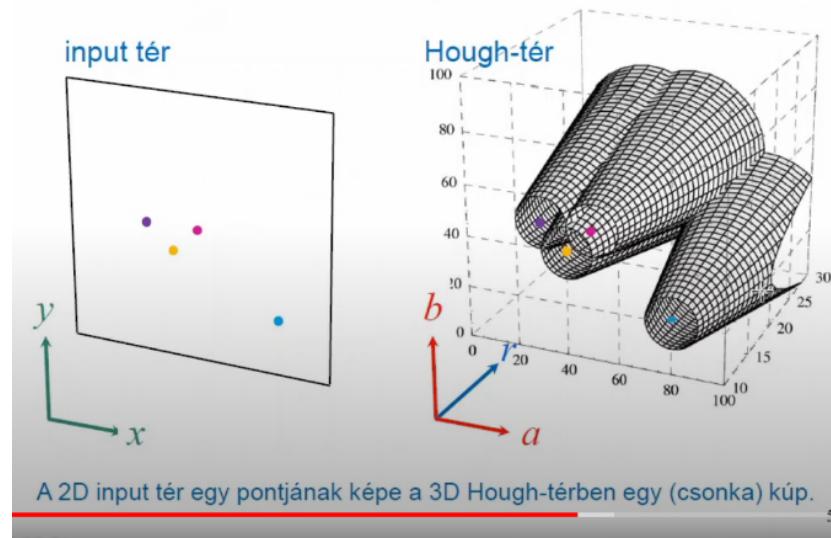
Kör detektálás: A 2D-s input térben egy egy pontnak, megfeleltethető a 3D-s Hough térben egy-egy csonka kúp. Ha nem ismerjük a sugarat, ha viszont ismerjük akkor csak egy kört fogunk kapni.

Körvonal detektálása

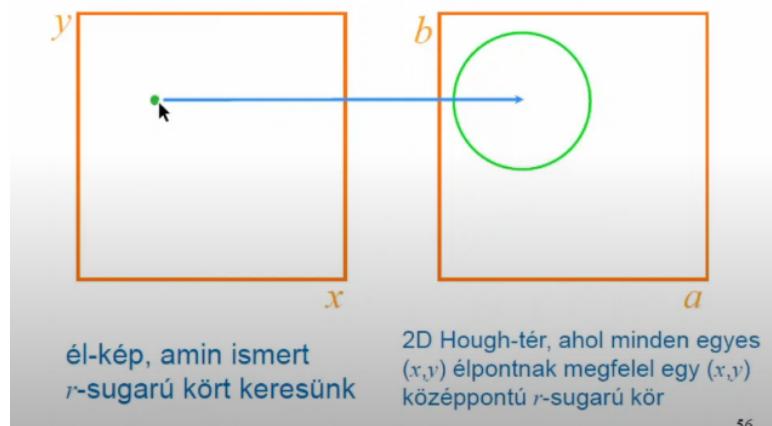
$$(x-a)^2 + (y-b)^2 = r^2 \rightarrow f(x,y,a,b,r) = (x-a)^2 + (y-b)^2 - r^2 = 0$$

Az (a, b, r) Hough-tér 3-dimenziós.
(Amennyiben pl. adott (konstans) r -sugarú kört keresünk, akkor a paraméter-tér 2-dimenziósra csökken.)

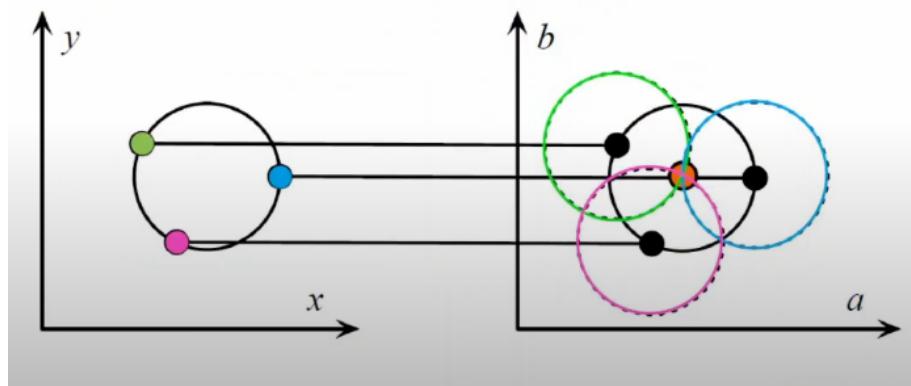
Körvonval detektálása



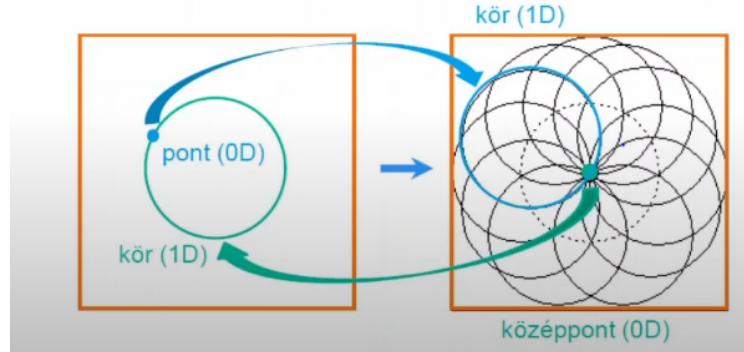
Körvonval detektálása



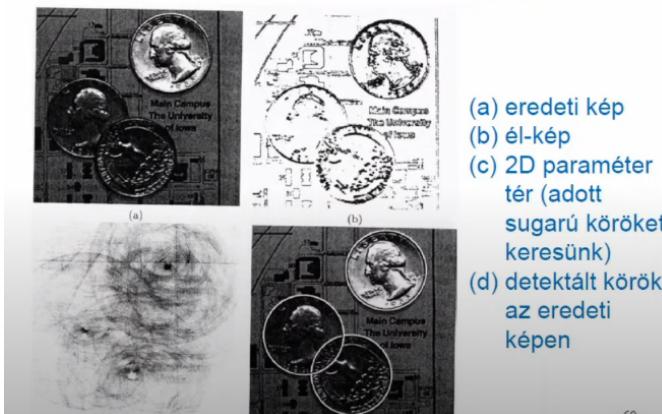
Körvonval detektálása



Körvonal detektálása



Példa körvonal detektálására



- (a) eredeti kép
- (b) él-kép
- (c) 2D paraméter tér (adott sugarú köröket keresünk)
- (d) detektált körök az eredeti képen

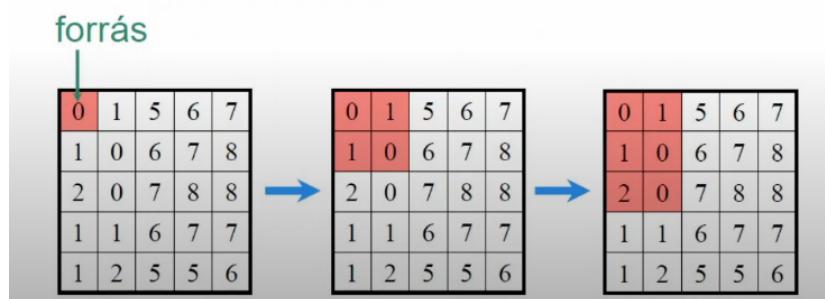
60

Régió növelő szegmentálás: Válasszunk egy kezdeti forráspontot, vagy egy kisebb összefüggő régiót. Ezt kezdjük el hizlalni úgy, hogy a szomszédos pontokat vizsgáljuk, meg és amelyek kielégítik, a hasonlósági kritériumokat adjuk hozzá a régiókhöz. Ha már nem bővíthető álljunk meg, vagy válasszunk egy új pontot és kezdjük előről. Akár egyszerre több régiót is hizlalhatunk, akik versenyeznek egymással.

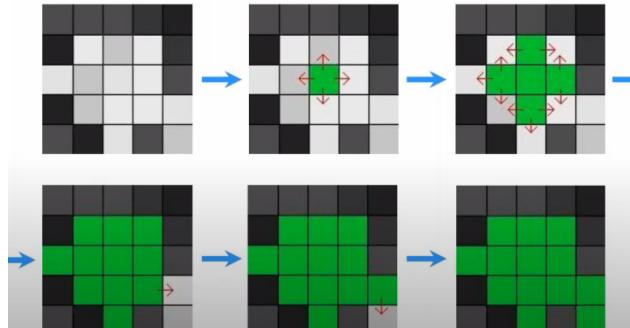
Példa régió növelésre

homogenitási kritérium:

$|f(u) - f(v)| < 3$, ahol v egy szegmentált u pont szegmentálatlan 8-szomszédja



Példa régió növelésre



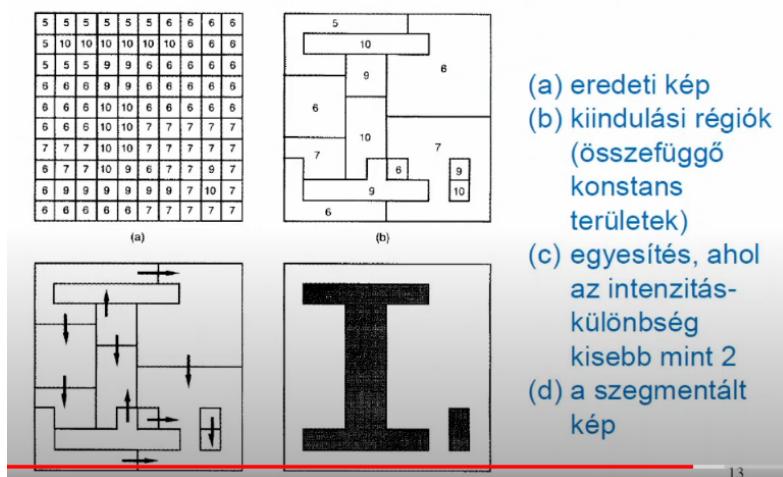
Régió egyesítés: Az inputképünk egy olyan kép, amin rengeteg szegmens van, tehát túlszegmentált a kép. Megvizsgáljuk a szomszédos régiókat, és amelyek megfelelnek, az összetartozási kritériumoknak azokat összeolvastjuk. Ha már nincs, több összevonás megállunk. Ez működik úgy is, hogy a kép feldolgozásakor már egyből régiókba soroljuk az adott pontokat, és ha utána találunk egyesíthetőt egyesítjük, ez az **egyemenes régió egyesítés**.

Példa egyesítési kritériumra

$$|\mu_i - \mu_j| < c \cdot \sigma_i, \quad |\mu_i - \mu_j| < c \cdot \sigma_j ,$$

ahol (μ_i, σ_i) és (μ_j, σ_j) az i -edik és a j -edik régió átlagos fényességét és szórását jelölik.

Példa régiók összefűzésére



Kis régiók megszüntetése: Előfordul, hogy szegmentáláskor túl szegmentálunk, vagy éppen alul. Ekkor hasznos ez az eljárás. Adjunk meg előre egy min méretet, aminél kisebb régiókat nem szeretnénk a képen látni. Keressük egy ilyen régiót, majd ahhoz nézzük meg a szomszédos régiókat, amelyik a leghasonlóbb hozzá egyesítünk össze vele. Ezt végezzük addig, amíg a minimum értékünk alatti összes régió eltűnik.

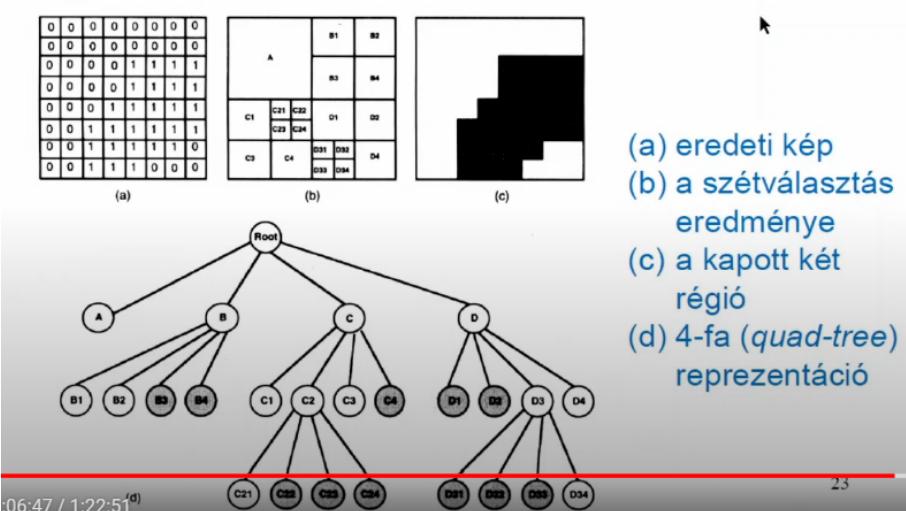
Régió szétválasztás: Előfordul, hogy szegmentáláskor túl szegmentálunk, vagy éppen alul. Ekkor hasznos ez az eljárás. A régió egyesítés ellentéte. Az egyes régiókat több féle szempont szerint akár szétválasztjuk kisebb részekre.

Régió szétválasztás és egyesítés

- Nem uniform régiókat bontsunk negyedekre.
- Uniform és szomszédos régiókat egyesítünk.



Példa szétválasztásra és egyesítésre (split&merge)



Illeszkedés alapú szegmentálás: Egy előre megadott minták, képek előfordulását keressük a képen.

Példa



Kereszt-korreláció: Illeszkedés alapú szegmentáláshoz használt eljárás. Megmutatja, ha a képen végig toljuk a kis képrészletet, hol egyezik a legjobban a képen.

Kereszt-korreláció (1D)

$x = (x_0, \dots, x_{N-1})$ és $y = (y_0, \dots, y_{N-1})$ normalizált kereszt-korrelációja:

$$r_{xy}(d) = \frac{\sum_{i=0}^{N-1} (x_i - \mu_x) \cdot (y_{i-d} - \mu_y)}{\sqrt{\sum_{i=0}^{N-1} (x_i - \mu_x)^2} \cdot \sqrt{\sum_{i=0}^{N-1} (y_i - \mu_y)^2}}$$

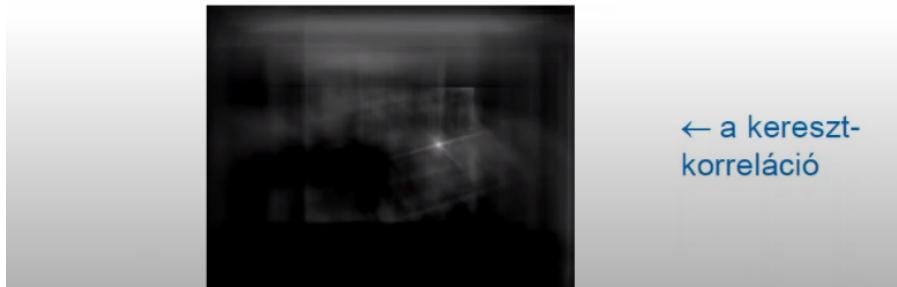
Normalizált kereszt-korreláció (2D)

$$r_{xy}(d, e) = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x_{ij} - \mu_x) \cdot (y_{i-d, j-e} - \mu_y)}{\sqrt{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x_{ij} - \mu_x)^2} \cdot \sqrt{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (y_{ij} - \mu_y)^2}}$$

2D példa



← az illesztési minta és a kép



← a keresztkorreláció

Illesztési kritériumok: illeszkedés alapú szegmentálásnál használjuk, megmutatja, mennyire illeszkedik egy adott részlet a kép adott helyére.

Illesztési kritériumok

$$C_1(u, v) = \frac{1}{1 + \max_{(i,j) \in V} |f(i+u, j+v) - h(i, j)|}$$

$$C_2(u, v) = \frac{1}{1 + \sum_{(i,j) \in V} |f(i+u, j+v) - h(i, j)|}$$

$$C_3(u, v) = \frac{1}{1 + \sum_{(i,j) \in V} (f(i+u, j+v) - h(i, j))^2}$$

Példa

1	1	0	0	0
1	1	1	0	0
1	0	1	0	0
0	0	0	0	0
0	0	0	0	8

(a)

1	1	1
1	1	1
1	1	1

(b)

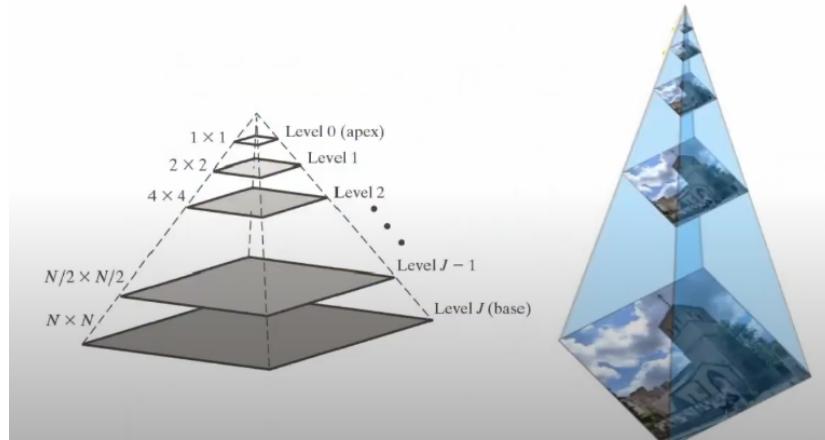
1/2	1/5	1/7	x	x
1/4	1/6	1/7	x	x
1/7	1/8	1/56	x	x
x	x	x	x	x
x	x	x	x	x

(c)

- (a) eredeti kép;
- (b) illesztési minta;
- (c) a C_3 kritérium szerinti értékek

Hierarchikus illesztés: Építünk egy képpiramist egyre kisebb képekből, majd a keresendő részlet lekicsinyített mását keressük a kisebb felbontású képen ezzel gyorsítható az algoritmus. Ha nem találunk, tovább megyünk az alsóbb szintekre ahol nagyobb a felbontás, és ott érdemes csak azokat a pontokat nézni, ahol az előzőök alapján gyaníthattuk, hogy illeszkedés van.

Kép piramis



Redundancia: A redundáns adat olyan adat melynek a mérete nagyobb mint az általa hordozott információ mérete. A kódolás/tömörítéssel tudunk a redundancián csökkenteni.

Legyen n_1 és n_2 ($n_1 > n_2$) a mérete az ugyanazon információt hordozó két adatfolyamnak. Ekkor

a tömörítési arány $C = \frac{n_1}{n_2}$,

a relatív redundancia $R = 1 - \frac{1}{C} = 1 - \frac{n_2}{n_1}$.

A képek tömörítésekor kihasználható redundanciák

- **kódolási**
(az egy képpontot kódoló bitek száma csökkenthető)
- **pixelek közötti**
(kihasználható, hogy a szomszédos képpontok hasonló intenzitásúak)
- **pszichovizuális**
(kihasználható, hogy az emberi látás nem egyformán érzékeny minden vizuális információra)

Fix és változó hosszúságú kódok: Fix, ha minden kódszó egyforma hosszú, változó, ha a kódszavak hossza különbözik valami alapján, például előfordulás alapján. Ami gyakrabban fordul elő rövidebb, ami ritkábban hosszabb. A változó hosszú kódoknak prefix mentesnek kell lennie, tehát nem fordulhat benne elő olyan kódszó, aminek egy rövidebb kódszó kezdő szelete lehet. Az **átlagos kódhossz** a változó kódoknál kedvezőbb.

Átlagos kódhossz

Az adat / üzenet / kép betűi / képpont-intenzitásai:

$$r_0, r_1, \dots, r_{L-1}$$

Az adatban az egyes kódszavak hossza (bitekben):

$$l_k = l(r_k) \quad (k = 0, 1, \dots, L-1)$$

Az adatban az egyes kódszavak relatív gyakorisága:

$$p_k = p(r_k) = \frac{n_k}{n} \quad (k = 0, 1, \dots, L-1)$$

Az adat átlagos kódhossza:

$$L = \sum_{k=0}^{L-1} l_k \cdot p_k$$

Kódolási redundancia

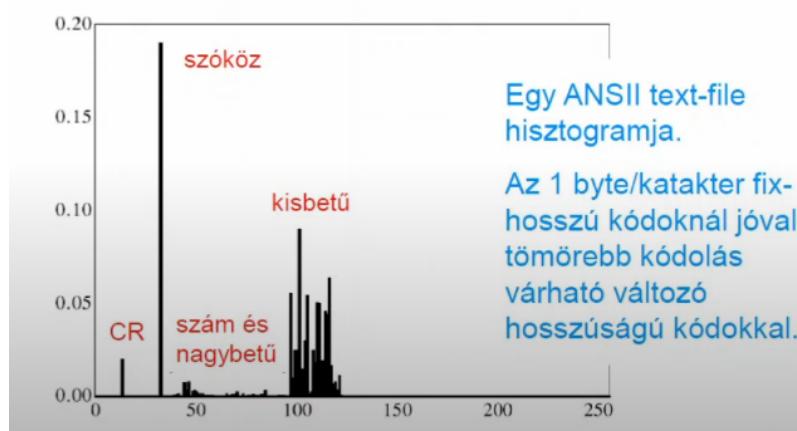
r_k	$p_r(r_k)$	Code 1	$I_1(r_k)$	Code 2	$I_2(r_k)$
$r_0 = 0$	0.19	000	3		11
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

átlagos kódhossz: **3** **2.7**

Az 1. kód redundáns, a relatív redundancia:

$$R = 1 - 2.7 / 3 = 0.099$$

Szövegfile-ok kódolása



Képpontok közötti redundancia: A képen előforduló szomszédos képpontok hasonló intenzitásúak, ezért elég letárolni az intenzitás különbségeket. Ezt nevezük delta kódolásnak. A nagy különbségeknél él fordul elő.

Képpontok közötti redundancia

A képen (az „erős” él-pontokat nem számítva) a szomszédos képpontok hasonló intenzitásúak, így az intenzitás különbségek várhatóan rövidebb átlagos kódhosszal kódolhatók.

a kép egy sora:

128	127	127	129	128	200	201	201	202	202
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

a különbségek:

128	-1	0	2	-1	72	1	0	1	0
-----	----	---	---	----	----	---	---	---	---

$$\Delta f(x, y) = \begin{cases} f(x, y) & , \text{ha } y = 0 \\ f(x, y+1) - f(x, y) & , \text{ha } y > 1 \end{cases}$$

15

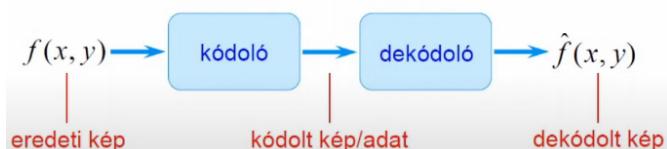
Pszchiovizuális redundancia: Azt használjuk ki, hogy amit nem érzékel, a szemünk azt nem fogja észrevenni. Akár eldobhatjuk az alsó 4 bitet, ekkor ugyan úgy fogjuk érzékelni az amúgy 8 bites képet csak 4 biten. Felére csökkent viszont a mérete. Vagy akár 256 árnyalat helyett használunk, csak 16 plusz adjunk hozzá zajt, így a különbség észrevehetetlen.

Pszichovizuális redundancia



Képkódolás: Kódolásból és dekódolásból álló rendszer. Ahhoz hogy dekódolni tudjunk, kell tudni a kódolás milyenségét. Ha dekódolás után a dekódolt kép nem egyezik az eredeti kódolt képpel, akkor veszteséges a kódolás. A kódolás és a dekódolás egyformán gyors legyen.

A képkódolás / tömörítés modellje



A kódolás veszteségményes, ha

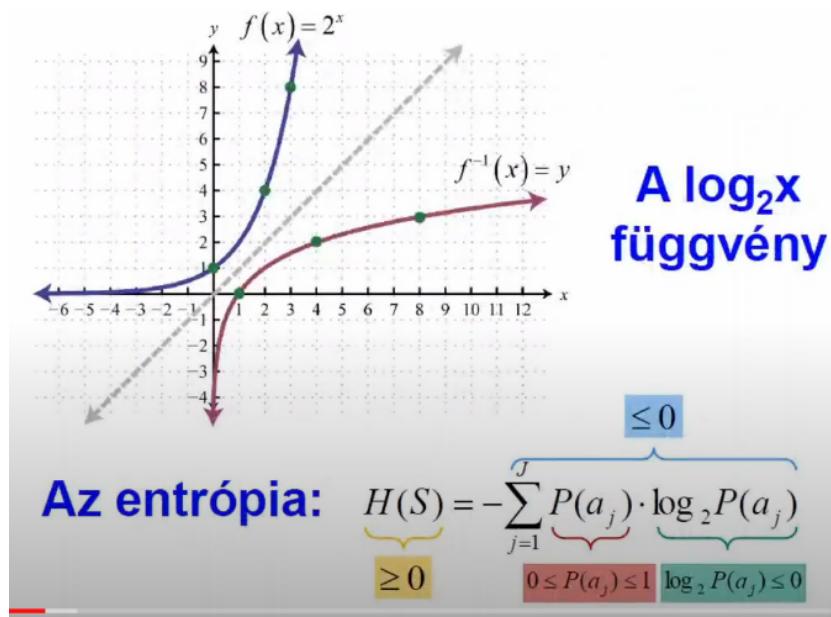
$$\hat{f}(x, y) = f(x, y) \quad (\forall (x, y) \in D)$$

Veszteségmentes és veszteséges tömörítés

A kódolás lehet veszteségmentes (*loss-less*) és veszteséges (*lossy*).

- **Veszteségmentes:**
 - a dekódolt kép megegyezik az eredetivel
 - képanalízisbeli alkalmazások igénylik
 - a tipikus tömörítési arány 2-10-szeres
- **Veszteséges:**
 - a dekódolt kép különbözik az eredetitől
 - leggyakrabban a kép- és a hang-kommunikációban alkalmazott
 - A tipikus tömörítési arány 10-30-szoros

Entrópia: A rendszerek rendezetlenségére bevezetett mérőszám. Azért kell a negatív előjel, mert az egyes pontok valószínűségeivel számolunk, ami 0 és 1 közötti érték, és a logaritmus függvény 0 és 1 közötti számokra minden minusz szám. Ezeket adogatjuk össze, de ahhoz hogy pozitív számot kapjunk, kell az előjelváltás. Az entrópia akkor maximális, ha a képen szereplő valamennyi szimbólunk egyforma valószínűsséggel fordul elő (2 esetén tehát 0,5 és 0,5). Az entrópia maximum $\log_2 J$ értéket vehet fel, a J az elemszám. Az entrópia akkor minimális, ha az üzenet tökéletesen egysíkú, azaz egy valami nagyon erősen jelen van a képen. Az egyszínű kép entrópiája 0.



Maximális entrópia

A $J=256$ lehetséges intenzitást tartalmazó képünk legyen olyan, amelyre

$$P(a_1) = P(a_2) = \dots = P(a_{256}) = 1/256$$

(vagyis minden árnyalat egyformán valószínű). Ekkor:

$$\begin{aligned} H(S) &= -\sum_{j=1}^{256} P(a_j) \cdot \log_2 P(a_j) = -\sum_{j=1}^{256} \frac{1}{256} \cdot \log_2 \frac{1}{256} \\ &= -\sum_{j=1}^{256} \frac{1}{256} \cdot (-8) = (-256) \cdot \frac{1}{256} \cdot (-8) = 8. \end{aligned}$$

(A kép lehet 8 bit/pixel-es).

27

•37:44

Minimális entrópia

Legyen az üzenet „egysíkú”:

$$P(a_1) = P(a_2) = \dots = P(a_{J-1}) = 0,$$

$$P(a_J) = 1.$$

Ekkor:

$$\begin{aligned} H(S) &= -\sum_{j=1}^J P(a_j) \cdot \log_2 P(a_j) \\ &= -\sum_{j=1}^{J-1} 0 \cdot \log_2 0 - 1 \cdot \log_2 1 = -0 - 0 = 0. \end{aligned}$$

(Pl. „0” az entrópiája az egyszínű képnek.)

28

Shannon-féle kódolási tétele: Azt mondja ki, hogy egy adott információ forrásnak az entrópiája megadja azt a minimális átlagos kódhosszt, amely a veszteségmentes kódoláshoz szükséges. Tehát ennél jobb veszteségmentes kódolást nem fogunk tudni készíteni.

$$H(S) \leq m(S),$$

Magasabb rendű entrópiák: Ez a fajta entrópia azt vizsgálja, hogy párban hogyan fordulnak elő az egyes elemek. Például egy szövegben x után hány esetben szerepel z. Ebben mátrixban tárolódik az információ.

Magasabb-rendű entrópiák

$$H_2(S) = -\sum_{i=1}^J \sum_{j=1}^J P(a_i, a_j) \cdot \log_2 P(a_i, a_j),$$

ahol $P(a_i, a_j)$ annak valószínűsége, hogy a_i -t a_j követi.

Bizonyított, hogy

$$H_1(S) \geq H_2(S) \geq \dots .$$

Veszteségmentes kódolások:

Veszteségmentes kódolások

- Huffman
- aritmetikai
- LZW
- delta
- konstans terület
- 4-fa
- futamhossz (bináris képekre)
- kontúr követés
- CCITT
- prediktív

Huffman kódolás: Mohó algoritmust alkalmaz. Egy optimális változó hosszúságú kódot ad eredményül. Tökéletes entrópia kódolás.

Példa fix hosszúságú és Huffman kódolásra

r_k	$p_r(r_k)$	Code 1	$I_1(r_k)$	Code 2	$I_2(r_k)$
$r_0 = 0$	0.19	000	3		11
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

átlagos kódhossz: 3 2.7

fix hosszú kód Huffman kód

Aritmetikai kódolás: Valamilyen szót, egy 0, 1 intervallumban lévő számra kódol le. Nem lehet vele nagy adatmennyiséget lekódolni, mert megtelne a lebegőpontos tárolás helye.

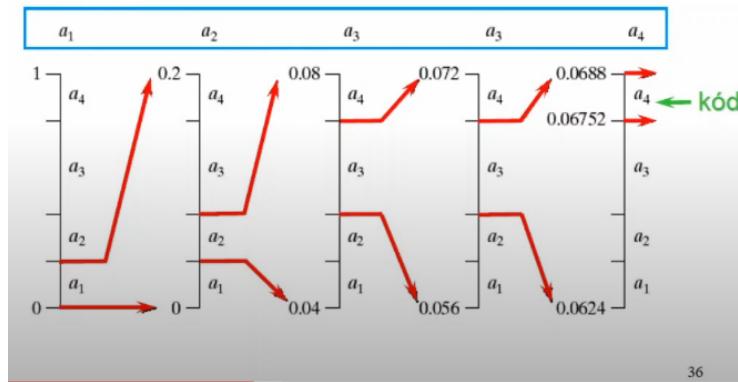
Példa aritmetikai kódolásra

A $[0,1)$ intervallum kezdeti felbontása:

forrás-szimbólum	valószínűség	rész-intervallum
a_1	0.2	$[0.0, 0.2)$
a_2	0.2	$[0.2, 0.4)$
a_3	0.4	$[0.4, 0.8)$
a_4	0.2	$[0.8, 1.0)$

Példa aritmetikai kódolásra

Üzenet:



36

LZW kódolás (Ziv kódolás): Végig meg egy stringen vagy egy képen, és felépít egy kódtáblát, majd kódolja az üzenetünket úgy, hogy a kód táblaindexek sorozata lesz. A dekódoló eljáráshoz nem kell tudni a kódoló táblát, mert dekódolás során a kód felépíti magának a kódtáblát, tehát elég csak az indexsorozatot megadni neki. Veszeség mentes tömörítés, nem nagyon tömörít.

Gray-kód: Jobb mint a bitsíkokra bontási kódolás.

Gray-kód

Az r m -bites szám/világosságkód bináris alakja

$$[r_{m-1} \dots r_1 r_0]$$

ahol:

$$r = 2^{m-1} r_{m-1} + \dots + 2^1 r_1 + 2^0 r_0 ,$$

Gray kódja

$$[g_{m-1} \dots g_1 g_0]$$

ahol:

$$g_i = \begin{cases} r_{m-1} & , \text{ha } i = m-1 \\ r_i \text{ XOR } r_{i+1} & , \text{ha } 0 \leq i < m-1 \end{cases} .$$

Példa Gray-kódra

m=3 :	szám	bináris	Gray
	0	000	000
	1	001	001
	2	010	011
	3	011	010
	4	100	110
	5	101	111
	6	110	101
	7	111	100

A szomszédos számok Gray kódjai csak egy bitben térnek el egymástól.

Konstans terület kódolás: Bináris kódolás. Bontsuk fel a képen át nem fedő blokkokra. Lesz fekete, fehér, vegyes.

Konstans terület kódolás (constant area coding, CAC)

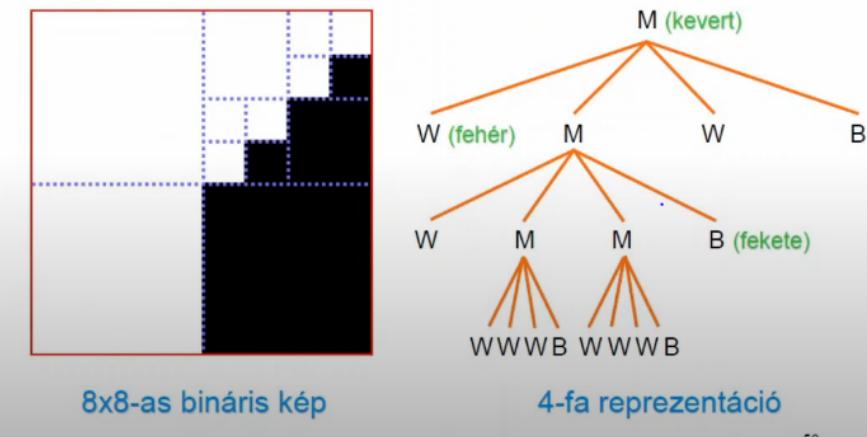
Ha a kép túlnyomórészt fehér pontokból áll (pl. nyomtatott szöveg, ahol a lap ~90%-a fehéren marad), akkor a kódolás:

- fehér: 0
- többi: 1+ *pxq* biten a blokktartalom.

A fenti WBS (*white block skipping*) kódolásnál gyakran a blokk a kép egy sora.

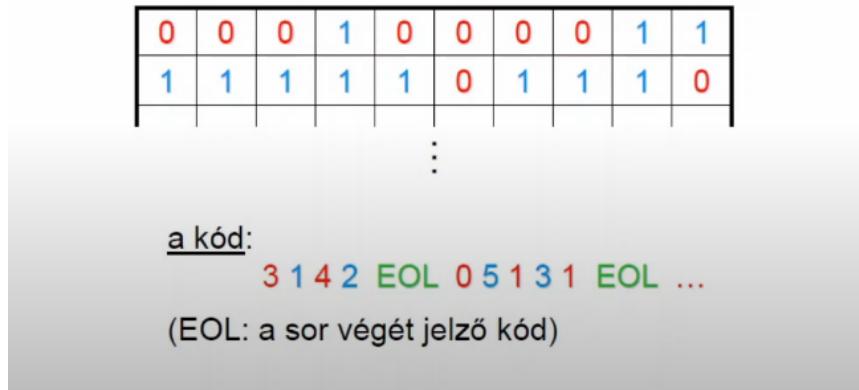
4-fa kódolás: Bontsuk a képet 4 részre, ahol kevert kép van azt újból.

4-fa kódolás (quad-tree)

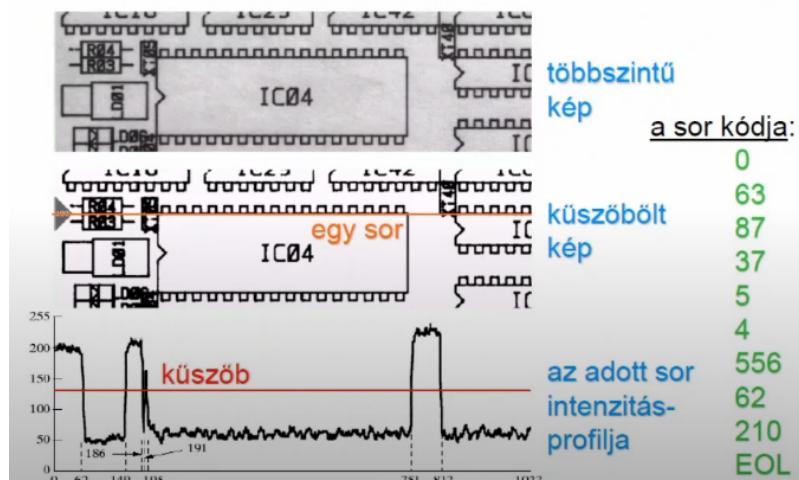


1D futamhossz kódolás: Képet analóg jelre kódolják.

Példa 1D futamhossz kódolásra



Példa 1D futamhossz kódolásra



2D futamhossz kódolás: Azt használja ki, hogy az egymás alatt lévő sorok nagy valószínűséggel, nem nagyon térnek el egymástól csak ritkán.

2D futamhossz kódolás – RAC

