Számítógépes képfeldolgozás

Kétdimenziós alakelemzés

Tófalvi Tamás

Eötvös Loránd Tudományegyetem, Informatikai Kar
tofi@inf.elte.hu



ELTE
EÖTVÖS LORÁND
TUDOMÁNYEGYETEM

Tartalom

1. Bináris képek adatstruktúrái
   - Bevezetés
   - Futam-hossz kód
   - Lánckód
   - Összehasonlítás
2. Terület alapú alakleírás
   - Alaknyomatékok
   - Orientáció
   - Értékelés
   - Alkalmazások
3. Kontúr alapú alakleírás
   - Alaktényező
   - Görbület
   - Kontúrsarkok
   - Alkalmazások
4. Összefoglalás

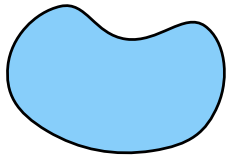
Tartalom

1. Bináris képek adatstruktúrái
   - Bevezetés
   - Futam-hossz kód
   - Lánckód
   - Összehasonlítás
2. Terület alapú alakleírás
3. Kontúr alapú alakleírás
4. Összefoglalás

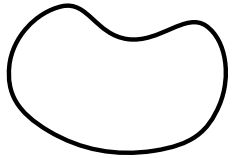
Az alakelemzés feladatai

- Kétdimenziós alakzatok leírása $\Rightarrow$ az objektumok strukturális elemzése, megkülönböztetése, osztályozása
  - lapos tárgyak és háromdimenziós tárgyak vetületei is
- Az alakelemzés (*shape analysis*) feladatai
  - alakzatleírás: számszerű leírók, alakzat-sajátságok (pl. kontúrsarkok) kiemelése
  - alakzat-szegmentálás és -illesztés
  - pozíció- és orientáció-meghatározás
- Kiindulás: bináris kép
  - küszöbölés, régió alapú szegmentálás, morfológiai feldolgozás után
  - objektum: 1, háttér: 0; az ábrákon az objektum fekete vagy színes

Terület és kontúr alapú módszerek



terület



kontúr

	terület alapú	**kontúr alapú**
operál	az alakzat teljes területén	az alakzat kontúrján
a pontok rendezése	2D-ben, a képsíkon	a kontúr mentén
2D lokális műveletek	támogatja	nem támogatja
számításigény	a területtől függ	a kerülettől függ
adatstruktúra	képmátrix, futam-hossz kód	lánckód

- Mindegyik adatstruktúránál az első lépés az **összefüggőség-elemzés**: az összefüggő objektum-régiók (komponensek) meghatározása

Összefüggő komponensek

- Két pixel **összefügg** ($p \leftrightarrow q$), ha szomszédos objektumpixelek sorozatán át eljuthatunk egyikből a másikba
  - 4- vagy 8-szomszédság (lásd a morfológia előadást)
  - az objektumra és a háttérre különböző: 8/4 vagy 4/8
- Az összefüggőség
  - reflexív: $p \leftrightarrow p$
  - szimmetrikus: ha $p \leftrightarrow q$, akkor $q \leftrightarrow p$
  - tranzitív: ha $p \leftrightarrow q$ és $q \leftrightarrow r$, akkor $p \leftrightarrow r$

  $\Rightarrow$ **ekvivalencia-reláció**: a bináris képet összefüggő komponensekre, azaz maximális összefüggő pixelhalmazokra bontja
- **Komponens-analízis**: minden objektumpixelhez hozzárendeljük a komponense címkéjét

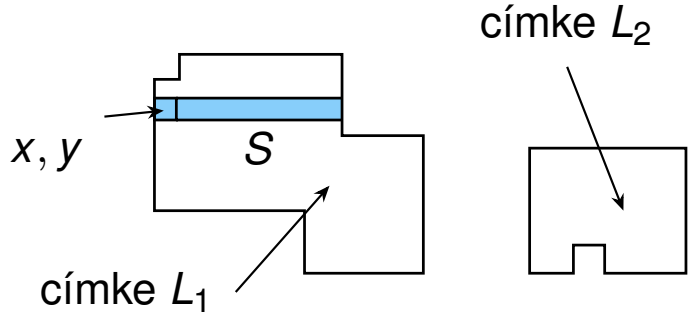
Az egyszerű rekurzív algoritmus

Komponensek keresése a képmátrixban

1. A képmátrixban megkeressük a következő be nem járt objektumpontot.
2. Rekurzív függvényhívással bejárjuk a szomszédait $\Rightarrow$ a kiinduló pont összefüggő komponensét.
3. A bejárt pixelekhez hozzárendeljük az aktuális címkét.
4. Növeljük a címkét, és megismételjük az eljárást.

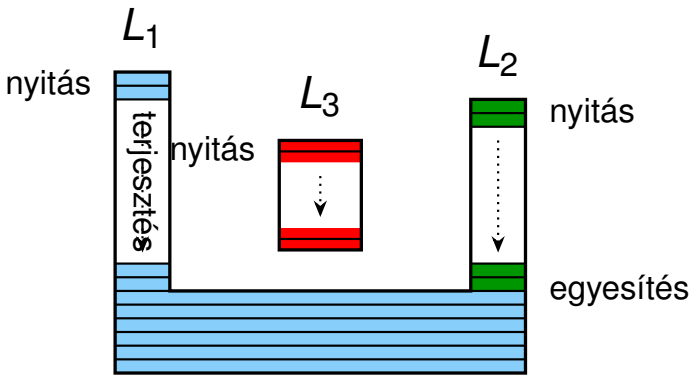
- A programkód nagyon egyszerű; nagy komponensekre viszont nem hatékony a mély rekurzió miatt
  - lassú, és el is szállhat, ha túlcsordul a verem (saját veremmel elkerülhető)
  - a bejárás sorrendje nem kontrollálható
- Csak akkor érdemes használni, ha a komponensek kicsik

A futam-hossz kód



- A bináris képet vízszintes **futamokkal** ábrázoljuk (*run-length code*, RLC); futam: objektumpixelek maximális összefüggő sorozata egy sorban
- Egy R futam kódja
  - x, y: az első pixel koordinátái; S: a futam hossza pixelben
  - L: a futam címkéje: R az L-edik komponenshez tartozik
- Címke nélküli RLC: a kép letapogatásával, egyszerűen és gyorsan
  - nagy, veszteségmentes tömörítés (kivéve, ha rengeteg apró komponens van); visszaállítás: a futamokat beírjuk a képmátrixba
- Címkézett RLC: komponens-analízissel, a címke nélküli RLC-ből

RLC alapú komponens-analízis



alapvető műveletek

- Bemenet: címke nélküli RLC; kimenet: a komponens sorszámával címkézett RLC
- Címke-ekvivalencia táblát használ, két menetben
- Az első menet három alapvető művelete
  - új címke nyitása (*open*)
  - címke továbbterjesztése (*propagate*)
  - ekvivalens címkék egyesítése (*merge*)
- A második menetben a tábla alapján átírjuk a címkéket $\Rightarrow$ folytonos, növekvő címkesor

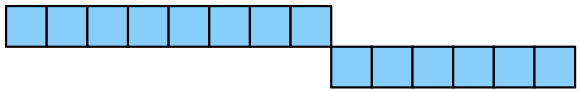
Az első menet

RLC alapú komponens-analízis: első menet

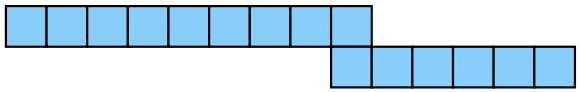
0. Választunk objektum-összefüggőséget (gyakran a 8-at: így az objektumot „preferáljuk”). Inicializálás: $E_{ij} = \delta_{ij}$ (Kronecker-delta), $L = 1$.
1. Minden új, címke nélküli R futamra megkeressük az előző sorban az R-rel összefüggő futamokat.
   - Nincs ilyen: **új címke**: R-hez L-et rendeljük, és növeljük L-et.
   - Egy van, $L' < L$ címkével: **továbbterjesztés**: R-hez L'-t rendeljük.
   - Több van, L'_k címkékkel: **egyesítés**: a legkisebb L'_m-t rendeljük R-hez, és minden k-ra $E_{km} = 1$.

- Második menet: az E tábla ekvivalencia-osztályai (tranzitív lezárás) szerint minden címkét az osztály sorszámára cserélünk
  - ehhez már csak a futamok listája és a tábla kell; a táblát a gyakorlatban *union–find* adatszerkezettel tároljuk

Futamok összefüggése



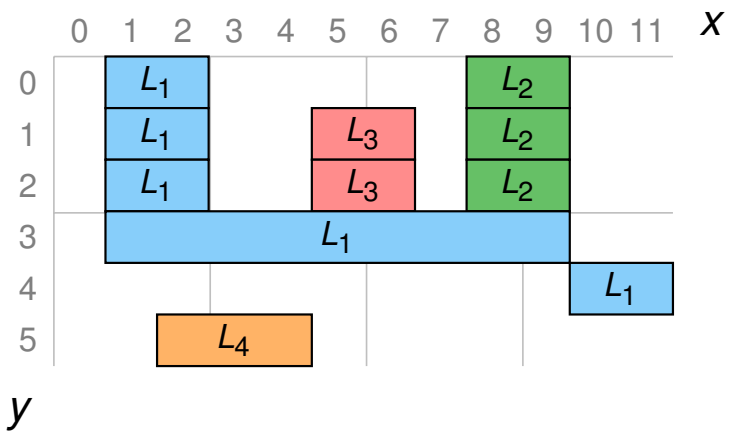
8-összefüggő futamok



4-összefüggő futamok

- Az y sor $[x_1, x_2]$ és az $y-1$ sor $[x'_1, x'_2]$ futama
  - **4-összefügg**, ha az oszloptartományok átfednek: $x_1 \leq x'_2$ és $x'_1 \leq x_2$
  - **8-összefügg**, ha átlósan is érintkezhetnek: $x_1 \leq x'_2 + 1$ és $x'_1 \leq x_2 + 1$
- Egy futam az előző sor több futamával is összefügghet $\Rightarrow$ ilyenkor kell egyesíteni
- A 4-összefüggő futampár 8-összefüggő is, fordítva nem mindig

Komponens-analízis: numerikus példa

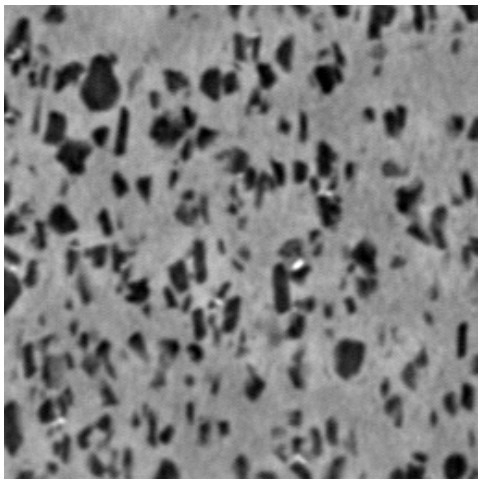


az első menet címkéi (8-összefüggés)

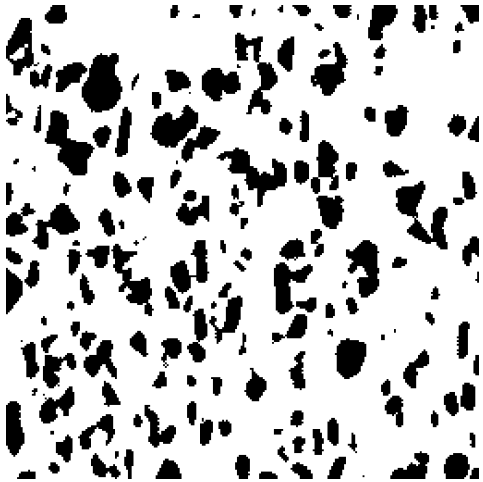
- $y = 3$: R_9 az L_1, L_3, L_2 futamokkal függ össze $\Rightarrow E_{21} = E_{31} = 1$
- 2. menet: $L_2, L_3 \rightarrow L_1$, $L_4 \rightarrow L_2$ $\Rightarrow$ 2 komponens
- 4-összefüggéssel R_{10} új címkét kap $\Rightarrow$ 3 komponens

R	y	x	S	1. menet		2. menet
1	0	1	2	L_1	nyitás	L_1
2	0	8	2	L_2	nyitás	L_1
3	1	1	2	L_1	terjesztés	L_1
4	1	5	2	L_3	nyitás	L_1
5	1	8	2	L_2	terjesztés	L_1
6	2	1	2	L_1	terjesztés	L_1
7	2	5	2	L_3	terjesztés	L_1
8	2	8	2	L_2	terjesztés	L_1
9	3	1	9	L_1	**egyesítés**	L_1
10	4	10	2	L_1	terjesztés	L_1
11	5	2	3	L_4	nyitás	L_2

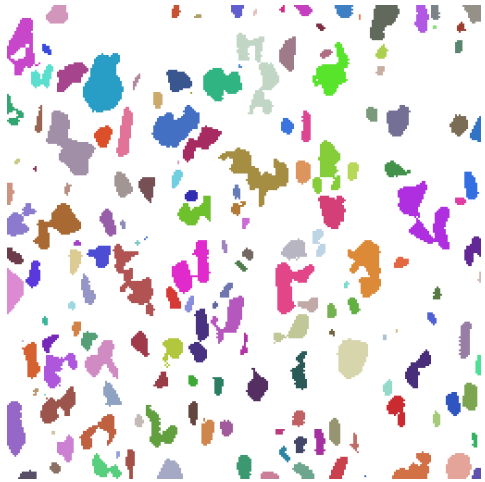
Komponens-analízis valós képen



szemcsekép



Otsu, $T = 103$

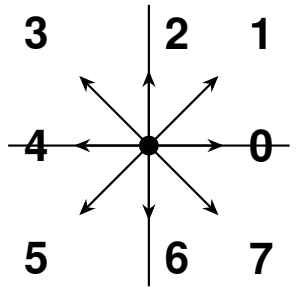


komponensek (8-ö.)

- $256 \times 256 = 65\,536$ pixel, ebből 15 160 objektumpixel $\Rightarrow$ 2338 futam (átlagosan 6,5 pixel); futamonként 3 szám: 7014 szám
- 8-összefüggés: az első menet 207 címkét nyit, 34-szer egyesít $\Rightarrow$ **171 komponens**
- 4-összefüggés: 191 komponens (az átlósan érintkező részek szétesnek); mindkettő azonos a rekurzív bejárásával

Bináris alakzat lánckódolása

- Kiválasztjuk a kezdő kontúrpixelt és a körbejárási irányt
- Végigkövetjük az alakzat (összefüggő komponens) kontúrját, amíg vissza nem érünk a kezdőpontba
- Közben tároljuk a lánckódokat: a következő kontúrpontra mutató vektor irányát
  - diszkrét képen 8 irány van (a számozás néha más)



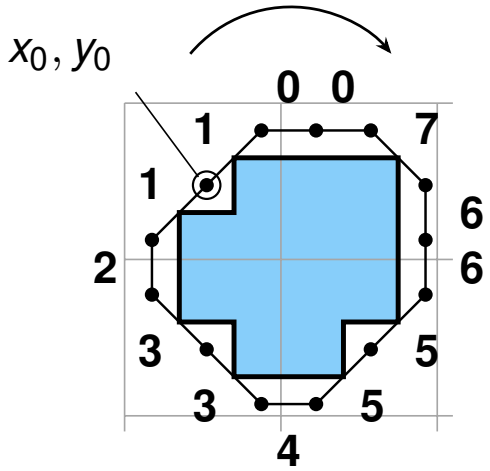
a nyolc lánckód

- Egy bináris alakzat lánckódja két részből áll
  1. a kezdőpont koordinátái: (x_0, y_0)
  2. a következő kontúrpontra mutató irányok sorozata: $\{c_1, c_2, \ldots\}$

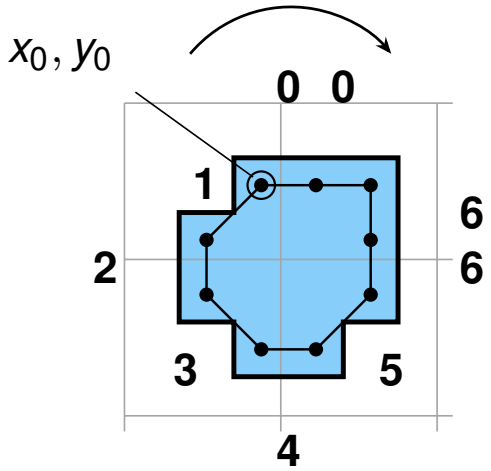
Bináris kép lánckódolása

- Egy bináris kép több összefüggő komponensből állhat
- A kép lánckódját úgy kapjuk meg, hogy
  - végigpásztázzuk a képet
  - ha találunk egy körbe nem járt komponenst, körbejárjuk és lekódoljuk
- A kép lánckódja az összes kontúr kódjából áll
  - benne lehetnek a belső kontúrok (lyukak) lánckódjai is $\Rightarrow$ a lyukakat külön meg kell jelölni
  - egymásba ágyazott (*nested*) komponensek is lehetnek
- Egy alakzatot kívül is, belül is körbe lehet járni: ez szabadon választható

Példák lánckódolásra



8-összefüggő háttér



8-összefüggő objektum

- Bal: a háttér 8-, a komponens 4-összefüggő; **kívül** járjuk körbe, a kezdőpont az első olyan háttérpont, amelytől jobbra objektumpont van: `1,0,0,7,6,6,5,5,4,3,3,2,1`
- Jobb: a háttér 4-, a komponens 8-összefüggő; **belül** járjuk körbe, a kezdőpont az első olyan objektumpont, amelytől balra háttérpont van: `0,0,6,6,5,4,3,2,1`

A kontúrkövetés algoritmusa

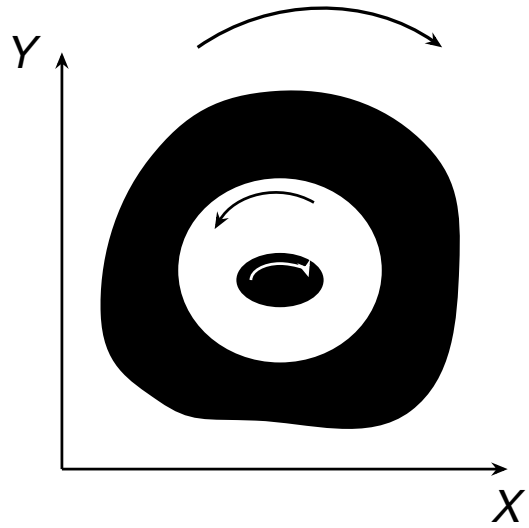
Kontúrkövetés lánckóddal (8-összefüggő objektum, belül)

1. Kezdőpont: az első objektumpont sorfolytonosan; legyen $d = 1$.
2. Az aktuális pont szomszédait az óramutató járása szerint (**csökkenő kóddal**) vizsgáljuk, a $j = (d + 1) \bmod 8$ (d páros), ill. $j = (d + 2) \bmod 8$ (d páratlan) iránytól kezdve: így az előző lépésben látott háttérpont után.
3. Az első objektumpont a következő kontúrpont, az iránya a következő lánckód: c; legyen $d = c$.
4. Folytatjuk, amíg a kezdőpontból ugyanabba az irányba nem lépnénk, mint az első lépésben.

- A kezdőpontba való visszaérés nem elég a leálláshoz: az egy pixel vastag részeken kétszer is áthaladunk rajta
- Az előző példa: `0,0,6,6,5,4,3,2,1`; kívül (háttérpontokon, növekvő kóddal, $d = 0$-ról): `1,0,0,7,...,2,1`

Külső és belső kontúrok

- Letapogatás közben nem tudjuk, hogy lyukban vagyunk-e; ez a körbejárás közben sem derül ki
- Megoldás: minden kontúrra kiszámítjuk a komponens **területét**, az **előjele** adja meg a választ
  - a külső kontúrokat jobbra fordulva, az óramutató járása szerint járjuk körbe $\Rightarrow$ a terület **pozitív** (X jobbra, Y felfelé)
  - a belső kontúrok körbejárása ebből automatikusan ellentétes $\Rightarrow$ a terület **negatív**
- Más választással az előjelek megfordulhatnak, de a különbséget az előjel jelzi



külső és belső kontúrok egymásba ágyazva

Kerület és terület a lánckódból

- A c_i kód lépése $(\Delta x_i, \Delta y_i)$, pl. $c = 1$: $(1, 1)$, $c = 6$: $(0, -1)$
- **Kerület**: a páros kódok lépése 1, a páratlanoké (átlós) $\sqrt{2}$:

$$L = n_{\text{páros}} + \sqrt{2}\, n_{\text{páratlan}}$$

- **Előjeles terület** (trapézszabály, y_i: az i-edik lépés utáni y):

$$S = \sum_i \Delta x_i \left(y_{i-1} + \frac{\Delta y_i}{2} \right)$$

  az óramutató járása szerinti körbejárásra pozitív, ellentétesre negatív
- A 10.3. ábra példái (a pixelszám 13)
  - belül: $L = 6 + 3\sqrt{2} \approx 10{,}24$, $S = 7{,}5$
  - kívül: $L = 6 + 7\sqrt{2} \approx 15{,}90$, $S = 18{,}5$
- Ezek a kontúrpontokon átmenő sokszög adatai $\Rightarrow$ a digitális alakzat „valódi” kerületét és területét csak közelítik

Külső és belső kontúrok: példa

Számítógépes képfeldolgozás
Kétdimenziós alakelemzés © 2026

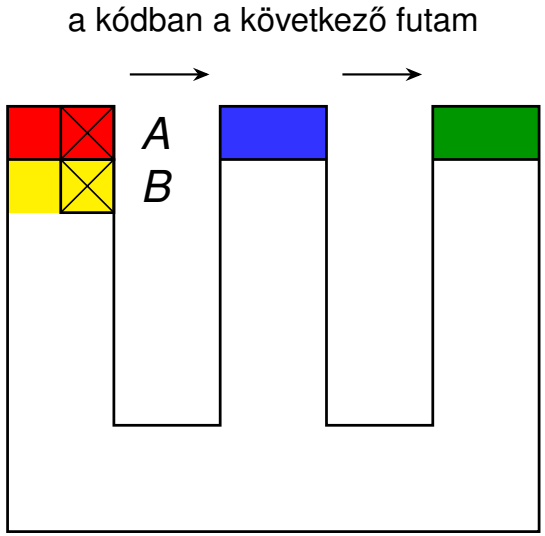
- Két sor szöveg bináris képe; a betűk nem érnek össze
  - minden kontúrt belül, ugyanazzal a szabállyal járunk körbe
  - piros: pozitív terület (külső kontúr, 66 db), kék: negatív terület (lyuk, 26 db)
- Az előjel a betűk szerkezetét adja vissza
  - minden betű, ékezet és i-pont külön komponens
  - lyuk van pl. az *a*, *d*, *e*, *g*, *o*, *p*, *0*, *6* karakterben
  - a © jel egymásba ágyazott: gyűrű (+), lyuk (–), a lyukban a *c* betű (+)
- Valós képen is működik: a szemcseképen 171 külső kontúr és 8 lyuk

Eltolás és elforgatás

- Gyakran változik az alakzat helye és orientációja
- **Eltolás**
  - futam-hossz kód: minden futam kezdőpontja változik, a hosszak nem
  - lánckód: csak a kezdőpont változik, maguk a kódok nem
- **Elforgatás**
  - mindkét kódot újra kell számolni: a változás csak triviális esetekben követhető
  - pl. 90°-os elforgatásnál minden lánckód $c \to (c+2) \bmod 8$
  - a **különbségi lánckód** $(c_i - c_{i-1}) \bmod 8$ erre invariáns
- Példa: `0,0,6,6,5,4,3,2,1` különbségi kódja `7,0,6,0,7,7,7,7,7`; a 90°-kal elforgatott alakzaté (`2,2,0,0,7,6,5,4,3`) ugyanez

A futam-hossz kód felhasználása

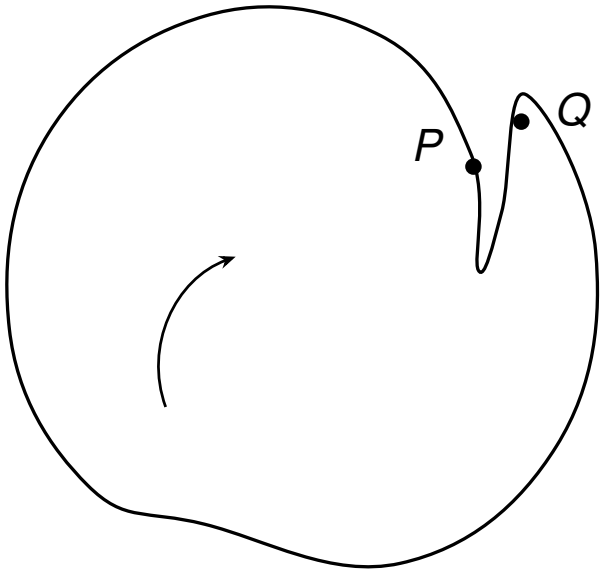
- A terület alapú reprezentációk (képmátrix, RLC) **támogatják** az integrális alakjellemzők számítását
  - pl. terület, nyomatékok
- **Nem támogatják** a kontúrelemzést
  - az *A* és a *B* pont a kontúron szomszédos, az RLC-ben viszont távol van: több futam választja el őket
  - a képmátrix síkbeli szomszédsága itt nehezebben érvényesíthető



futam-hossz kód

A lánckód felhasználása

- A kontúr alapú reprezentációk (lánckód) **támogatják** a kontúrelemzést és egyes integrális jellemzők számítását
  - pl. görbület, kerület, terület, nyomatékok
- **Nem támogatják** a lokális képműveleteket
  - a *P* és a *Q* pont a képen közel van egymáshoz, a kontúron viszont távol
  - a síkbeli szomszédság még nehezebben érvényesíthető: végig kell járni a kontúrt



lánckód

Tartalom

1 Bináris képek adatstruktúrái

2 Terület alapú alakleírás
- Alaknyomatékok
- Orientáció
- Értékelés
- Alkalmazások

3 Kontúr alapú alakleírás

4 Összefoglalás

Centrális nyomatékok

- A Q alakzat pq-rendű **centrális nyomatéka** (*shape moment*), $p, q = 0, 1, \ldots$

$$\text{folytonos esetben:} \quad \mu_{pq} = \frac{1}{S} \iint_Q (x - x_C)^p (y - y_C)^q \, dx \, dy$$

$$\text{diszkrét esetben:} \quad \mu_{pq} = \frac{1}{S} \sum_{(x,y) \in Q} (x - x_C)^p (y - y_C)^q$$

  - S: a Q területe (pixelszáma)
  - centrális: a súlyponthoz (*centroid*) képest, $x_C = \frac{1}{S} \sum_{(x,y) \in Q} x$, $y_C = \frac{1}{S} \sum_{(x,y) \in Q} y$
- Elvben az összes μ_{pq}-ből az alakzat visszaállítható; a gyakorlatban kevés, alacsony rendű ($p + q \leq 4$) nyomatékot használunk $\Rightarrow$ nem kontrollálható információveszteség
  - a magasabb rendűek zaj- és torzításérzékenyek: nagy p-re x^p deriváltja nagy, ezért x^p felerősíti az x zaját
- A továbbiakban a **másodrendű** nyomatékok ($p + q = 2$): μ_{20}, μ_{02}, μ_{11}

Kompaktság és excentricitás

- Két elforgatás-invariáns jellemző a másodrendű nyomatékokból

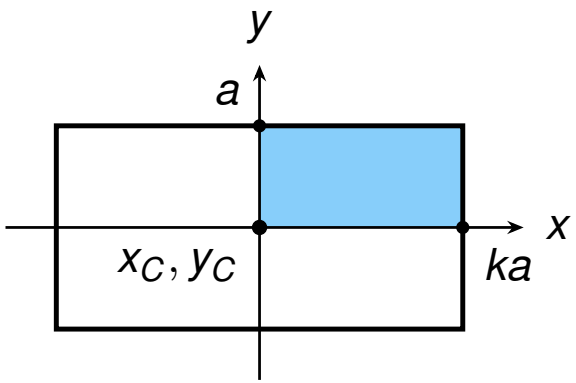
$$M_{cmp} = \frac{1}{2\pi} \cdot \frac{S}{\mu_{20} + \mu_{02}}, \qquad M_{ect} = \frac{\sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}}{\mu_{20} + \mu_{02}}$$

- **Kompaktság** (*compactness*): $0 \leq M_{cmp} \leq 1$
  - végtelen hosszú alakzatra $M_{cmp} = 0$, lemezre $M_{cmp} = 1$
  - ellenőrzés: R sugarú lemezre $S = \pi R^2$, $\mu_{20} = \mu_{02} = R^2/4$

$$\Rightarrow M_{cmp} = \frac{\pi R^2}{2\pi \cdot R^2/2} = 1$$

- **Excentricitás** (*eccentricity*): ugyanarra a tartományra normált, de fordítva viselkedik
  - végtelen hosszú alakzatra $M_{ect} = 1$, lemezre $M_{ect} = 0$
- Folytonos esetben mindkettő invariáns minden hasonlósági transzformációra: eltolásra, elforgatásra, izotróp skálázásra
- Diszkrét esetben a rácson való eltolásra invariánsak, az elforgatás- és skálázásinvariancia csak közelítő

Példa: téglalap



$2ka \times 2a$-s téglalap, $k \geq 1$

$$S = 2a \cdot 2ka = 4ka^2$$

$$\mu_{20} = \frac{4}{S} \int_0^a \int_0^{ka} x^2 \, dx \, dy = \frac{k^2 a^2}{3}$$

$$\mu_{02} = \frac{4}{S} \int_0^a \int_0^{ka} y^2 \, dx \, dy = \frac{a^2}{3}$$

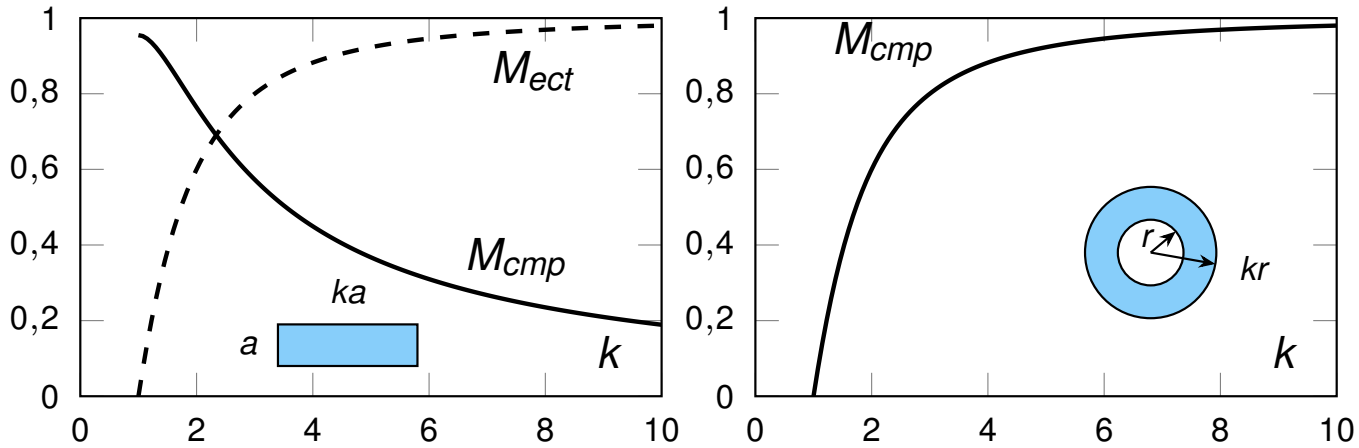
$$\mu_{11} = \frac{1}{S} \int_{-a}^a \int_{-ka}^{ka} xy \, dx \, dy = 0$$

- A két jellemző

$$M_{cmp} = \frac{1}{\pi} \cdot \frac{6k}{k^2+1} \approx \frac{6}{\pi k}, \qquad M_{ect} = \frac{k^2-1}{k^2+1} \approx 1 - \frac{2}{k^2} \quad (k \gg 1)$$

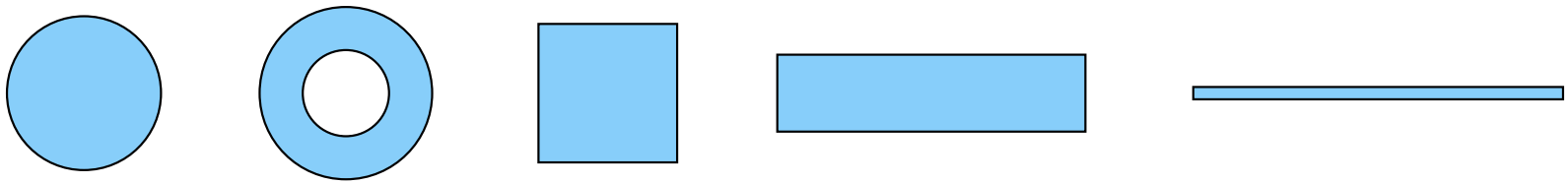
- Hosszabb téglalapra a kompaktság csökken, az excentricitás nő

Téglalap és gyűrű



- Téglalap: mindkét jellemző változása lassul, de M_{ect} hamarabb laposodik el $\Rightarrow$ hosszú téglalapokat ($k > 5$) nem különböztet meg; négyzetre ($k = 1$) $M_{cmp} \approx 0{,}96$, de k-val gyorsan csökken
- Gyűrű ($k \geq 1$: minél nagyobb, annál vastagabb): $M_{ect} = 0$ (körszimmetrikus), $M_{cmp} = \dfrac{k^2 - 1}{k^2 + 1}$
  - $1 < k < 4$ között jól megkülönböztet, vastag gyűrűket ($k > 5$) nem; lyukas lemezre ($k \gg 1$) $M_{cmp} \approx 1$

Mit mér a két jellemző?

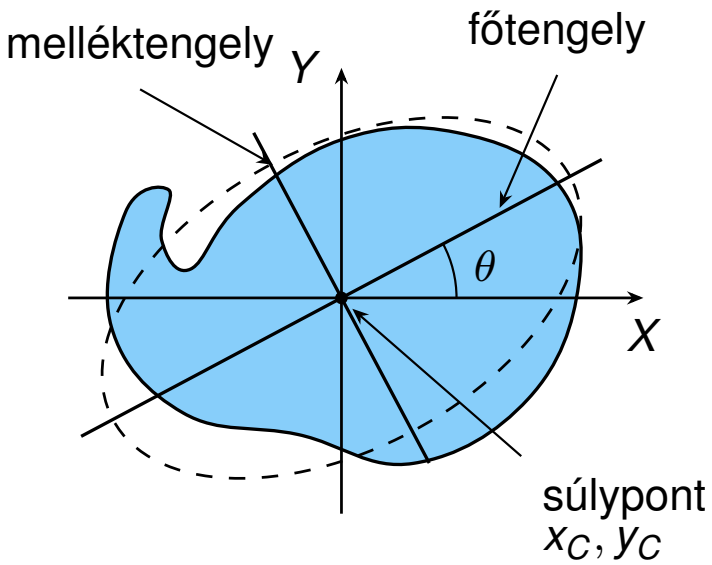


lemez	gyűrű ($k = 2$)	négyzet	téglalap 1:4	vonal
1 / 0	0,6 / 0	0,955 / 0	0,449 / 0,882	$\to 0$ / $\to 1$

M_{cmp} / M_{ect}

- M_{cmp}: a pontok **radiális eloszlása**
  - lemezre $M_{cmp} = 1$, gyűrűre $M_{cmp} < 1$
  - robusztus zajra és a diszkrét képen történő elforgatásra
- M_{ect}: az alakzat **elnyújtottsága**
  - lemezre és gyűrűre (és négyzetre!) $M_{ect} = 0$, vonalra $M_{ect} = 1$
  - nagyobb mértékben érzékeny zajra és elforgatásra

Inerciatenzor és inerciatengelyek



- μ_{20}, μ_{02}, μ_{11}: az I_{ij} **inerciatenzor** elemei
  - a súlyponton átmenő tengelyek körüli forgást írja le
- A tenzor az alakzatot a legjobban közelítő **ellipszissel** modellezi
  - **főtengely** (a hosszú): erre minimális az inercia, $I_{\min}$
  - **melléktengely** (a rövid): erre maximális, $I_{\max}$
- $M_{ect} \propto I_{\max} - I_{\min}$: hosszabb ellipszis, nagyobb excentricitás

Orientáció

- Az alakzat **helye** (pozíciója): az x_C, y_C súlypont
- **Orientációja**: a főtengely és az X tengely θ szöge

$$\theta = \frac{1}{2} \arctan \frac{\mu_{02} - \mu_{20}}{2\mu_{11}} + (\operatorname{sign} \mu_{11}) \cdot \frac{\pi}{4}$$

  - ugyanez kétargumentumú arkusztangenssel, $\mu_{11} = 0$-ra is érvényesen: $\theta = \frac{1}{2} \operatorname{atan2}(2\mu_{11}, \mu_{20} - \mu_{02})$
- θ cirkuláris mennyiség: csak $\bmod \pi$ határozható meg
- **Hosszúkás** alakzatokra jól használható: a hosszabb ellipszis pontosabb orientációbecslést tesz lehetővé
- **Kompakt**, körhöz közeli alakzatokra pontatlan: a képlet numerikusan instabil, ha $\mu_{02} - \mu_{20}$ és μ_{11} is kicsi
- A főtengely **nem definiált**, és a képlet nem alkalmazható, ha az alakzatnak három vagy több szimmetriatengelye van
  - ilyenkor $\mu_{20} = \mu_{02}$ és $\mu_{11} = 0$, pl. négyzetre, szabályos háromszögre

Sajátértékek és az ellipszis

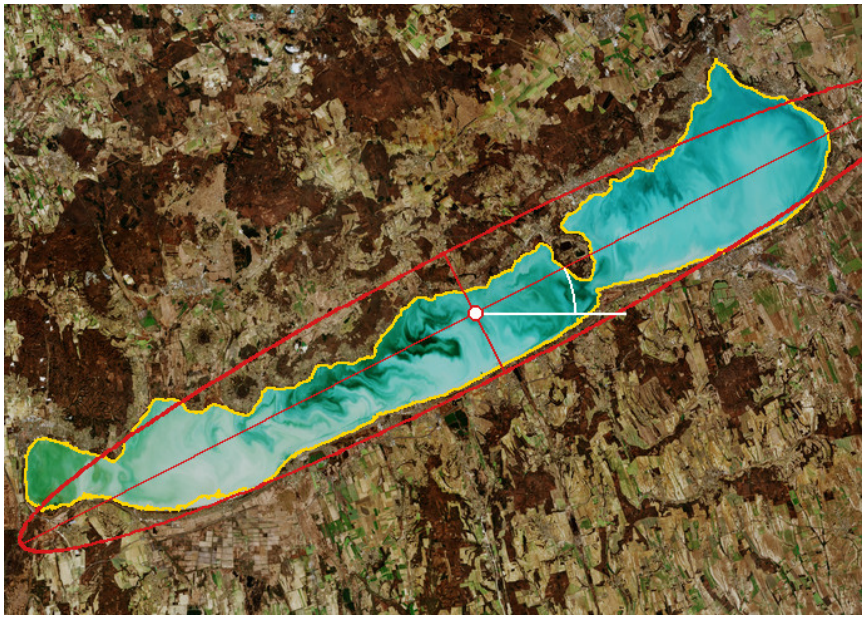
- A másodrendű nyomatékok mátrixa és sajátértékei ($\lambda_1 \geq \lambda_2$):

$$\begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{pmatrix}, \quad \lambda_{1,2} = \frac{\mu_{20} + \mu_{02}}{2} \pm \frac{1}{2}\sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}$$

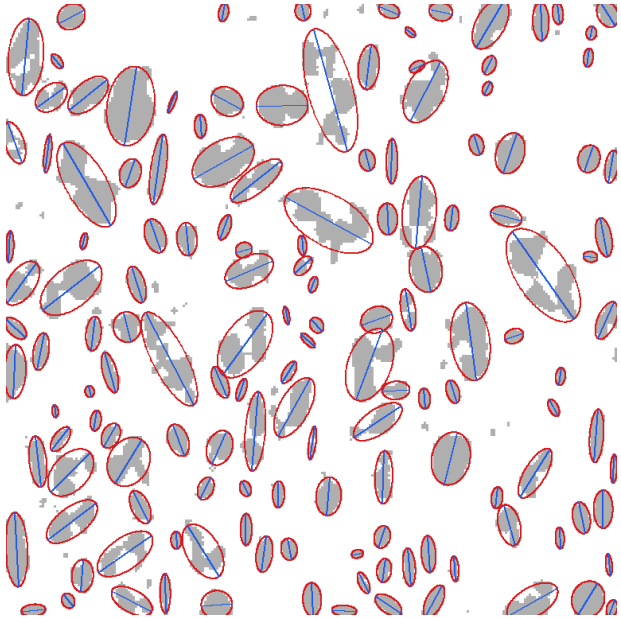
- λ_1: a pontok szórásnégyzete a főtengely mentén, λ_2: rá merőlegesen $\Rightarrow I_{\min} \propto \lambda_2$, $I_{\max} \propto \lambda_1$
  - $M_{ect} = \dfrac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}, \quad M_{cmp} = \dfrac{S}{2\pi(\lambda_1 + \lambda_2)}$
  - a λ_1-hez tartozó sajátvektor iránya éppen θ
- Az a, b féltengelyű ellipszisre $\mu_{20} = a^2/4$, $\mu_{02} = b^2/4 \Rightarrow$ az alakzattal **azonos nyomatékú ellipszis** féltengelyei

$$a = 2\sqrt{\lambda_1}, \qquad b = 2\sqrt{\lambda_2}$$

Nyomatékok valós képen



Balaton: súlypont, tengelyek, ellipszis



szemcsék ($S \geq 12$)

- Balaton: $S = 192\,895$ pixel, $M_{cmp} = 0{,}18$, $M_{ect} = 0{,}97$, $\theta = 27°$, a féltengelyek aránya 7,6 : 1
- Az ellipszis másodrendű nyomatékai egyeznek az alakzatéval, az alakja nem: a tó inkább sáv, ezért az ellipszis túlnyúlik rajta

Kép: ESA, Copernicus Sentinel-2 (2019), CC BY-SA 3.0 IGO

Az alaknyomatékok összefoglalása

- A nyomatékok kiszámíthatók
  - a szegmentált képen, a definíció szerint
  - a futam-hossz kódból (kicsit bonyolultabb)
  - a lánckódból (még bonyolultabb)
- $p + q = 3$: öt invariáns nyomaték-kombináció
  - az alakzat aszimmetriáját tükrözik; zaj- és torzításérzékenyek, ezért ritkábban használják, mint M_{cmp}-t és M_{ect}-t
- Kiterjesztések: szürke és színes képekre, affin-invariáns változatok
- Alkalmazások
  - viszonylag kis számú (10–20), egymástól jól elkülönülő alakzat felismerése
  - előszelektálás sok alakzat esetén: a szóba jöhető jelöltek gyors kiválogatása kontúrillesztés vagy más számításigényes eljárás előtt
  - más alakjellemzőkkel kombinálva
  - robotlátás, karakterfelismerés, affin-invariáns megfeleltetés

Az invariáns nyomatékok előnyei és hátrányai

Előnyök

- elforgatás-, eltolás- és nagyításinvariánsak
- viszonylag egyszerűen kiszámíthatók
- különböző adatstruktúrákban meghatározhatók
- M_{cmp} robusztus zajra és kisebb torzításra
- elég jól megkülönböztetik az alakzatokat
- az orientáció is meghatározható

Hátrányok

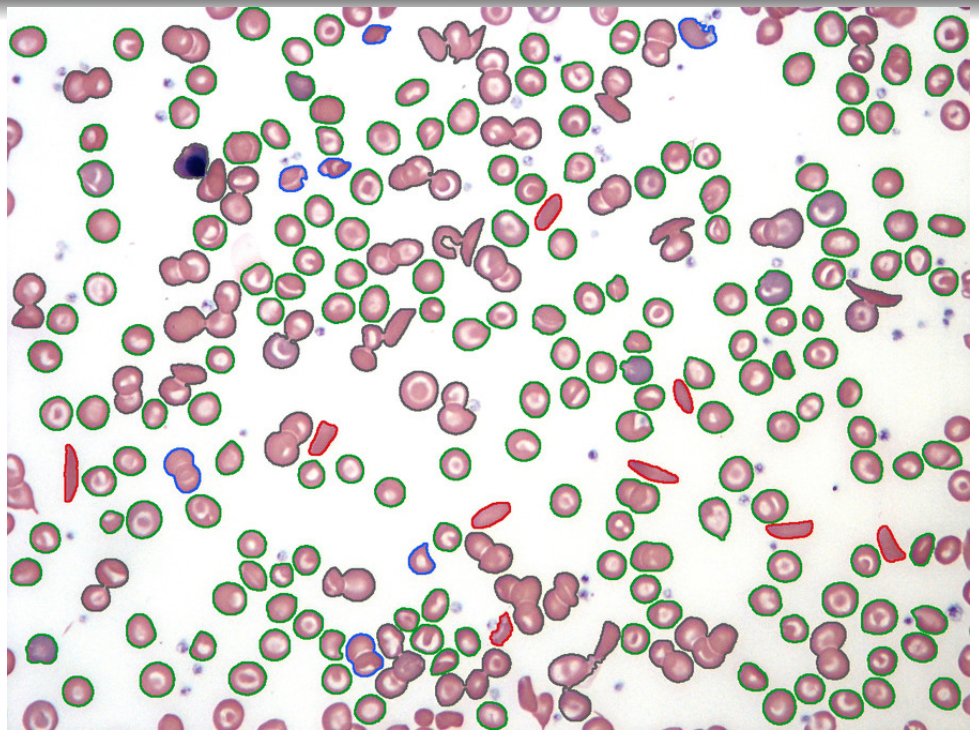
- nem tükröznek lokális kontúrsajátságokat
- M_{ect} és a magasabb rendű nyomatékok kevésbé robusztusok
- az orientáció pontatlan vagy definiálatlan lehet

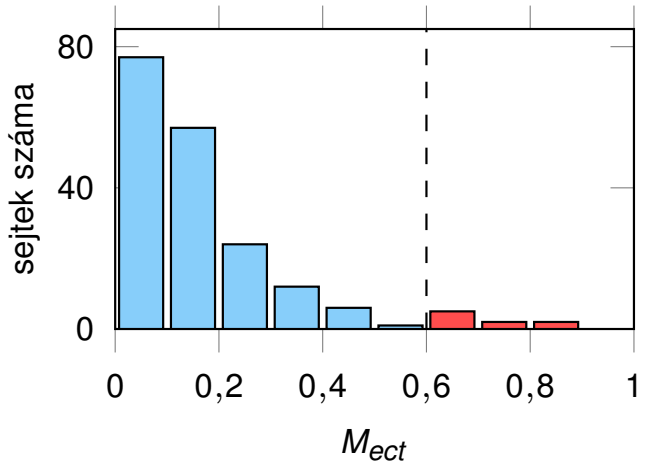
Hol használják a nyomatékokat?

leíró	alkalmazás
nyomaték-invariánsok	**Repülőgépek felismerése** sziluettből: 6 típus, 132 kép, az emberi megfigyelőknél kisebb hibával (Dudani et al., 1977, Hu invariánsaival)
súlypont, orientáció	**Ipari robotlátás**: a véletlenszerűen elhelyezett alkatrészek helye és szöge a megfogáshoz (SRI Vision Module: Gleason, Agin, 1979)
súlypont, orientáció	**Fejkövetés webkamerával**: a bőrszín-valószínűségi kép nyomatékaiból a fej helye, mérete, dőlése; játékok vezérlése (CamShift: Bradski, 1998)
ellipszis, orientáció	**Muslicák viselkedése**: legfeljebb 50 egyed videón; minden komponensre ellipszis a pixelek átlagából és kovarianciájából (Ctrax: Branson et al., 2009)
kompaktság	**Földrajzi körzetek** tervezése: $C_{MI} = A^2/(2\pi I)$ (I: a súlypontra vett tehetetlenségi nyomaték) éppen M_{cmp} (Li et al., 2013)
excentricitás, orientáció	**Sejtbiológia**: a CellProfiler minden sejtre megadja; excentricitása az ellipszisé: $e = \sqrt{1 - b^2/a^2}$

- A θ csak $\bmod \pi$ ismert: azt, hogy a légynek melyik vége a feje, a Ctrax külön lépésben, a mozgásirányból dönti el

Vörösvértestek: sarlósejtes vérszegénység

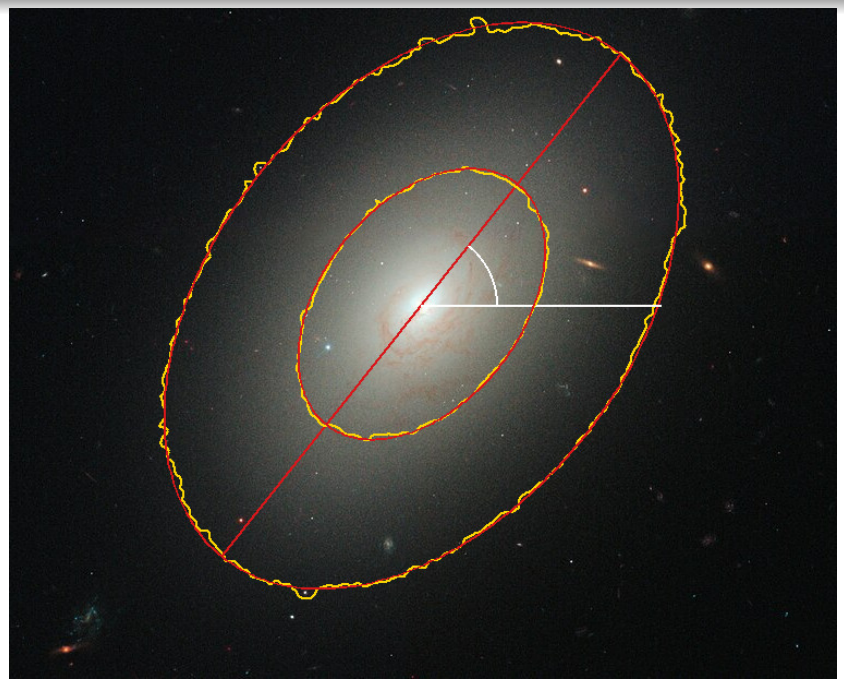




- Vérkenet automatikus kiértékelése (Delgado-Font et al., 2020): **elnyúlt**, ha $b/a \leq 0{,}5$; ez a mi jelölésünkkel $M_{ect} \geq 0{,}6$
- 186 különálló sejtből 9 elnyúlt; a többi normál (170) vagy egyéb deformáció (7, ehhez az alaktényező kell); a 39 összetapadt csoport kimarad

Kép: K. Chambers, Wikimedia Commons, CC BY-SA 3.0, módosítva

Galaxisok alakja



T	S	b/a	θ
30	176 835	0,652	51,5°
50	109 423	0,639	51,5°
70	76 039	0,637	51,0°
100	40 484	0,651	51,0°

NGC 4150; sárga: az azonos fényességű kontúr (izofóta) $T = 30$ és 100 mellett, piros: az azonos nyomatékú ellipszis

- Az ellipszis szinte fedi az izofótát, b/a és θ alig függ a küszöbtől $\Rightarrow$ robusztus mérés
- A csillagászati katalógusok ugyanígy mérnek: a SExtractor program a másodrendű nyomatékokból adja a tengelyeket és a szöget
- Elliptikus galaxisok Hubble-osztálya: En, $n = 10\,(1 - b/a)$

Kép: NASA, ESA, R. M. Crockett et al. (Hubble, közkincs), módosítva

Tartalom

1 Bináris képek adatstruktúrái

2 Terület alapú alakleírás

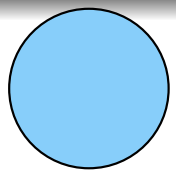
3 Kontúr alapú alakleírás
- Alaktényező
- Görbület
- Kontúrsarkok
- Alkalmazások

4 Összefoglalás

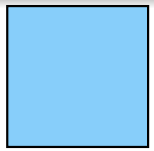
Az alaktényező

- Az **alaktényező** (*shape factor*), ahol S a terület, L a kerület:
$$F = \frac{4\pi S}{L^2}, \qquad 0 \leq F \leq 1$$
- Más néven **körszerűség** (*circularity*)
  - lemezre $F = 1$ ($S = \pi R^2$, $L = 2\pi R$), hosszú, keskeny alakzatra $F \ll 1$
  - a kontúr simaságát is tükrözi
- Folytonos esetben, mint a nyomatékok, invariáns minden hasonlósági transzformációra
- Diszkrét esetben a rácson való eltolásra invariáns; az elforgatás- és skálázásinvariancia csak közelítő
  - függ a felbontástól: új kontúrrészletek megjelenésével a kerület lényegesen változik
- Viszonylag robusztus; zajos kontúrt simítani, közelíteni kell

Alaktényező és kompaktság



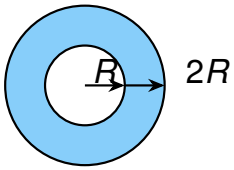
$F = 1$



$F = \frac{\pi}{4} \approx 0{,}79$



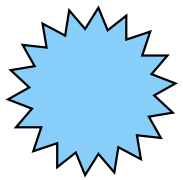
$F = \frac{4\pi}{25} \approx 0{,}5$



$F = \frac{1}{3} \approx 0{,}33$



$F \ll 1$



$F \ll 1$

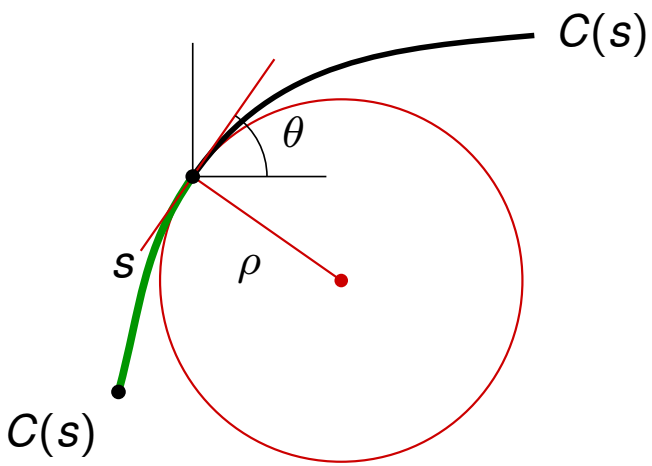
- Lemezre a kompaktság és az alaktényező is maximális (1)
- Cikcakkos határú lemez: M_{cmp} csak kissé csökken, F a kerület növekedése miatt nagyot zuhan
- Gyűrű (R, $2R$): $F = 1/3$, de $M_{cmp} = 0{,}6 \Rightarrow$ közel kétszeres eltérés
- Hosszú téglalap ($a \times 4a$): $F \approx 0{,}5$, $M_{cmp} \approx 0{,}45 \Rightarrow$ alig különböznek

Kompaktság és alaktényező digitális alakzatokon

	M_{cmp}			F		
alakzat	elmélet	$r = 50$	$r = 5$	elmélet	$r = 50$	$r = 5$
kör (r)	1	1,000	0,993	1	0,907	0,936
négyzet ($2r$), 0°	0,955	0,955	0,967	0,785	0,802	0,994
négyzet ($2r$), 45°	0,955	0,955	0,955	0,785	0,797	0,906
téglalap ($4r \times r$)	0,449	0,443	0,472	0,503	0,506	0,617
gyűrű ($r/2$, r)	0,600	0,600	0,582	0,333	0,296	0,276

- S: pixelszám; L: a kontúr(ok) lánckód-kerülete; M_{cmp} még $r = 5$-re is $\pm 0{,}03$-on belül marad
- F érzékenyebb
  - a digitális kör F-je kb. 0,9: a lánckód-kerület nem a valódi kerület
  - kis méretnél erősen változik, és az elforgatástól is függ: a 9×9-es négyzetre $F = 0{,}994$, majdnem mint a körre
- Balaton: $F = 0{,}139$, $M_{cmp} = 0{,}182$

A görbület definíciója



- $C(p) = \{x(p), y(p)\}$: p paraméterű görbe, pl.
  - $p = x$: $C(x) = \{x, y(x)\}$
  - $p = s$ ívhossz: $C(s) = \{x(s), y(s)\}$
- A **görbület** definíciója

$$\kappa = \frac{d\theta}{ds} = \frac{1}{\rho}$$

- θ: az érintő iránya, $\tan\theta = \frac{dy}{dx}$ a meredekség; s: az ívhossz, $ds^2 = dx^2 + dy^2$
- ρ: az érintőkör (simulókör) sugara, a **görbületi sugár**

Angolul: görbe *curve*, görbület *curvature*, meredekség *slope*, ívhossz *arc length*, érintőkör *osculating circle*, görbületi sugár *radius of curvature*.

A görbület kiszámítása

- Tetszőleges $\{x(p), y(p)\}$ paraméterezésre

$$\kappa = \frac{x_p\, y_{pp} - y_p\, x_{pp}}{\left(x_p^2 + y_p^2\right)^{3/2}}, \qquad \text{ahol } x_p = \frac{\partial x}{\partial p}$$

- Az $\{x, y(x)\}$ paraméterezésre ($x_p = 1$, $x_{pp} = 0$)

$$\kappa = \frac{y_{xx}}{\left(1 + y_x^2\right)^{3/2}}$$

  - gond lehet: $y(x)$ többértékű lehet; függőleges érintőnél $|y_x| \to \infty$
- Az s ívhosszal paraméterezett görbére nincs ilyen gond
  - $x(s)$ és $y(s)$ egyértékű, x_s és y_s korlátos: $x_s^2 + y_s^2 = 1 \Rightarrow \kappa = x_s\, y_{ss} - y_s\, x_{ss}$
- A görbület **előjeles**: az érintőkör a görbétől balra ($\kappa > 0$) vagy jobbra ($\kappa < 0$) eshet

Görbület, simítás, skálatér

- A deriválás miatt a görbület **zajérzékeny** $\Rightarrow$ a görbét előbb simítjuk (átlagoló, tipikusan Gauss-szűrővel)
  - növekvő σ-jú Gauss-simítás utáni görbületszámítás: görbületi **skálatér** (*scale-space*)
  - az élszűréshez hasonlóan fokozatosan csökkenő részletesség
- A nagy görbületű pontok **sarkok**, kontúr-töréspontok
  - jól lokalizálhatók, fontosak az alak- és a mozgáselemzésben, az emberi és a gépi látásban egyaránt
- A görbület nullátmenetei **inflexiós pontok**: kevésbé jól lokalizálhatók, kevésbé fontosak
- Diszkrét esetben kontúr-adatstruktúra kell: gyakran a lánckód
  - zajos képből zajos, „cikcakkos” lánckód lesz
  - diszkretizálási hibák: nagyon keskeny részek, szélsőséges esetben önmagát érintő kontúrszakaszok
  - $\Rightarrow$ pontatlan görbület, hamis sarkok

Görbületszámítás diszkrét kontúron

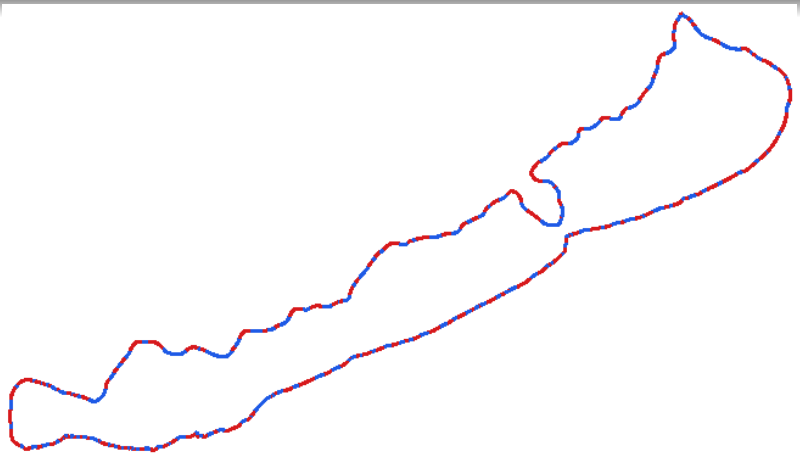
Görbület ívhosszal paraméterezett diszkrét görbén

1. Eltávolítjuk a zajt, elsimítjuk a görbét; közben ügyelünk a potenciális sarkokra, hogy ne tegyük tönkre őket.
2. Az $x(s)$, $y(s)$ görbét szabályos, állandó Δs lépésközzel újramintavételezzük; ehhez interpoláció kell.
3. Kiszámítjuk a görbületet a κ képlettel, a deriváltak közelítésével:

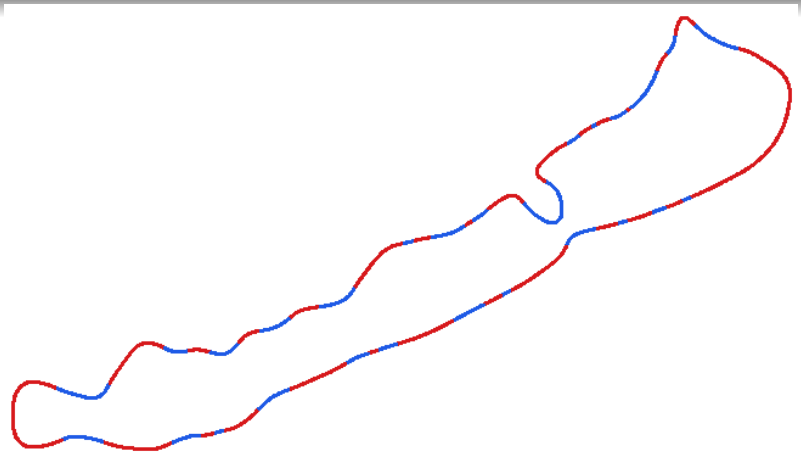
$$x_s \approx \frac{x_{i+1} - x_{i-1}}{2\Delta s}, \qquad x_{ss} \approx \frac{x_{i+1} - 2x_i + x_{i-1}}{(\Delta s)^2}$$

(y-ra ugyanígy).

Görbület valós kontúron



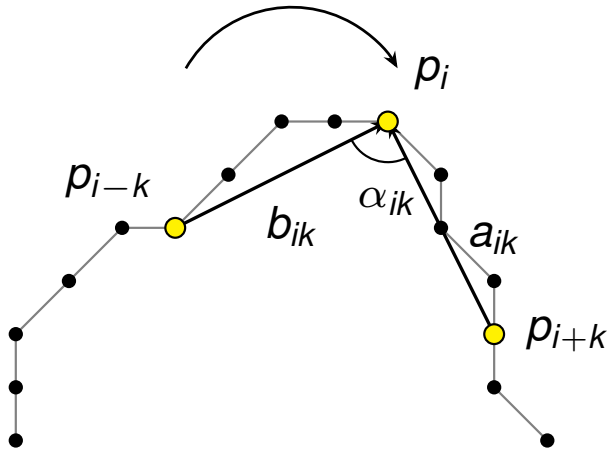
$\sigma = 4$



$\sigma = 16$

- A Balaton kontúrja (3514 lánckód), Gauss-simítás után $\Delta s = 1$ pixellel
  - piros: konvex, kék: konkáv ív
- Az inflexiós pontok (előjelváltások) száma a skála függvényében
  - $\sigma = 2, 4, 8, 16$ pixel: 590, 282, 132, 54
  - kis σ-nál a part „zaja” dominál, nagynál a tó öblei és félszigetei maradnak meg

Sarokerősség: k-koszinusz



$k = 4$: $a_{ik} = (-2, 4)$,
$b_{ik} = (4, 2)$
$\Rightarrow c_{ik} = 0, \ \alpha_{ik} = 90°$

k-koszinuszok

1. A sarok finomságát szabályozó skálaparaméter: $k = 1, 2, \ldots$
2. k-vektorok:

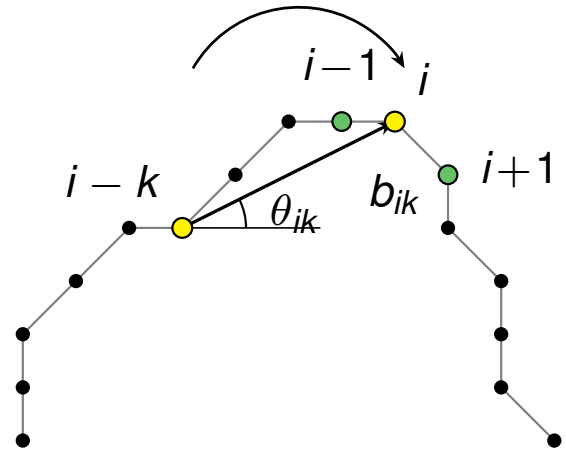
$$a_{ik} = (x_i - x_{i+k}, \ y_i - y_{i+k})$$
$$b_{ik} = (x_i - x_{i-k}, \ y_i - y_{i-k})$$

3. Sarokerősség:

$$c_{ik} = \cos \alpha_{ik} = \frac{(a_{ik} \cdot b_{ik})}{|a_{ik}| \, |b_{ik}|}$$

- Kisebb k: finomabb sarkok, de zajérzékenyebb eljárás
- Egyenesen $c_{ik} = -1$, derékszögben 0, hegyes saroknál $\to 1$

A k-vektor szögváltozása



$\theta_{i-1,k} = 26{,}6°$, $\theta_{i+1,k} = 0°$
$\Rightarrow \Delta\theta_{ik} = -26{,}6°$

- A b_{ik} k-vektor komponensei

$$(X_{ik}^-, Y_{ik}^-) = (x_i - x_{i-k}, y_i - y_{i-k})$$

- Az iránya

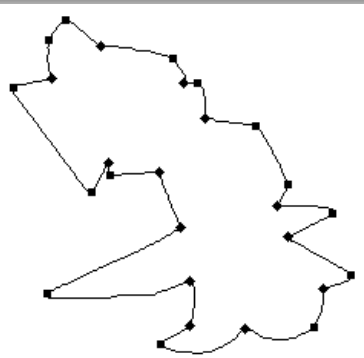
$$\theta_{ik} = \arctan \frac{Y_{ik}^-}{X_{ik}^-}$$

- Alternatív sarokerősség: a k-vektor **szögváltozása**

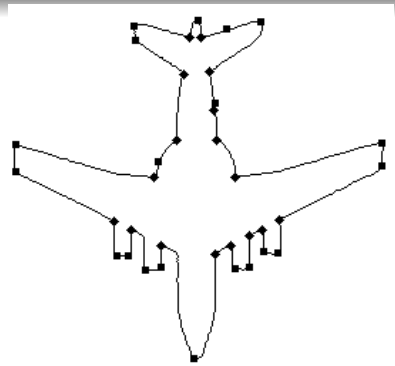
$$\Delta\theta_{ik} = \theta_{i+1,k} - \theta_{i-1,k}$$

- b_{ik} az $i - k$ és az i pont közötti húr $\Rightarrow \Delta\theta_{ik}$ a húr közepe, az $i - k/2$ pont körüli irányváltozást méri: egy sarokra a válasz maximuma $k/2$ ponttal a sarok után van

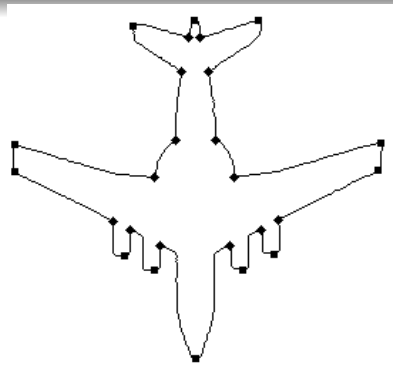
Nem-maximum elnyomás és példák



kis k



kis k

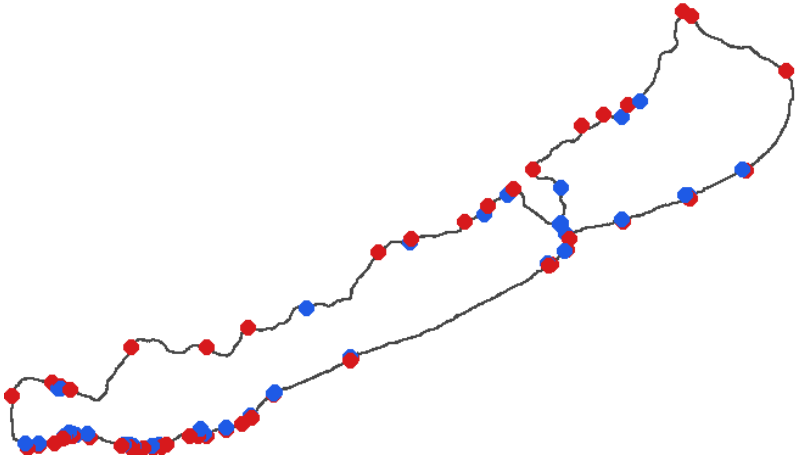


nagyobb k

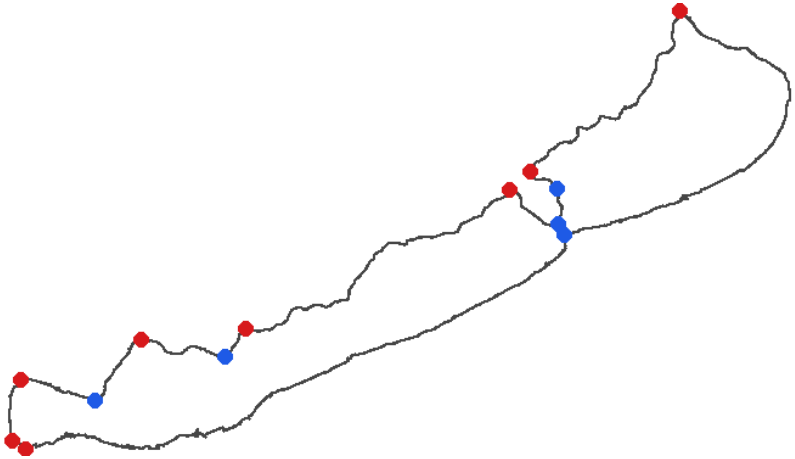
- Az éldetektáláshoz hasonlóan a sarokerősségre **nem-maximum elnyomást** alkalmazunk
  - egyszerűbb, mint a képen: a kontúron könnyű megtalálni a szomszédos pontokat
- Kis k: zajos, hamis detektálás is van, de a repülő kontúrján minden sarkot megtalálunk, az egymáshoz közelieket is
- Nagy k: a nem-maximum elnyomás miatt az egymáshoz közeli sarkok közül az egyik elvész

Kép: Csetverikov D.: jegyzet, 10.15. ábra

Sarokdetektálás valós kontúron



$k = 5$: 93 sarok



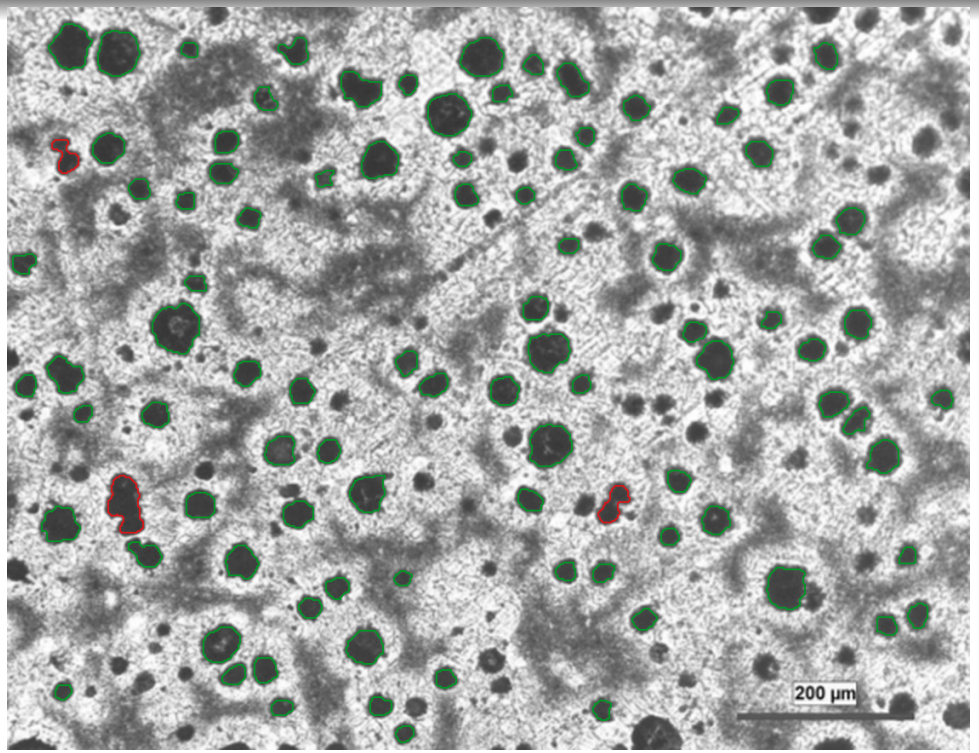
$k = 40$: 13 sarok

- k-koszinusz a Balaton lánckódján; sarok, ha $c_{ik} > -0{,}6$ ($\alpha_{ik} < 127°$), és $\pm k/2$ ponton belül ez a maximum
  - piros: konvex, kék: konkáv sarok
- Kis k: sok hamis sarok a part zajos szakaszain
- $k = 20$: 26, $k = 40$: 13 sarok, köztük a Tihanyi-félsziget, a keleti csúcs és a nyugati medence sarkai

Hol használják az alakténye zőt és a görbületet?

leíró	**alkalmazás**
alakténye ző	**Sejtek, szemcsék**: az ImageJ (*Circularity*) és a CellProfiler (*FormFactor*) is $4\pi A/P^2$-et számol
alakténye ző	**Vörösvértestek**: $F < 0{,}8 \Rightarrow$ deformált (Delgado-Font et al., 2020)
kompaktság (kontúr)	**Emlődaganatok**: a kontúr kompaktsága 92,3%-ban választotta szét a sima és a tüskés szélű elváltozásokat (Rangayyan et al., 1997)
alakténye ző	**Választókerületek**: a Polsby–Popper-pontszám éppen $4\pi A/P^2$
görbület, skálatér	**Növényhatározó**: a Leafsnap a levélkontúr több skálán mért görbületéből azonosít (184 fafaj; Kumar et al., 2012)
görbület, skálatér	**Képkeresés**: az MPEG-7 kontúrleírója a görbületi skálatér, inflexiós pontok simítással (Bober, 2001)
görbület	**Szemészet**: a retinaerek kanyargósságának mérése görbülettel (Hart et al., 1999)
görbület	**Sávtartás**: az út modellje a klotoid, $\kappa(s) = c_0 + c_1 s$ (Gackstatter et al., 2010)

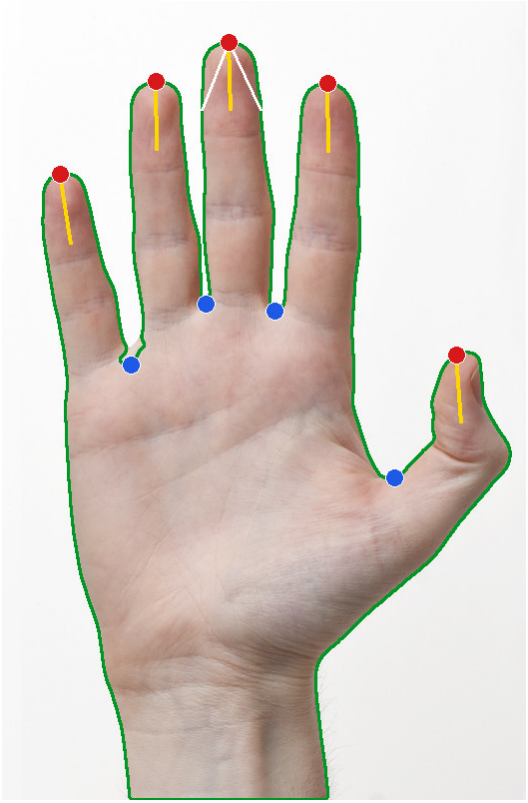
Gömbgrafitos öntöttvas



- Az öntöttvas tulajdonságait nagyban meghatározza a grafit alakja
- Az ASTM E2567 szabvány képelemzéssel méri a **nodularitást**: a gömbszerű szemcsék területaránya
- Itt: 96 szemcse, medián $F = 0{,}97$
- $F \geq 0{,}7$: 93 szemcse, a grafitterület 95%-a; $F < 0{,}7$: 3 szemcse, szabálytalan vagy összeérő gömbök (a küszöb itt szemléltető)
- Hiszterézises küszöbölés (vö. éldetektálás), mert a szürke perlites foltok is sötétek; csak $S \geq 80$ pixel, mert kis alakzatra F megbízhatatlan: itt is 26 szemcsére $F > 1$

Kép: Michelshock, Wikimedia Commons, közkincs, módosítva

Ujjhegyek keresése k-koszinusszal



- Gesztusvezérlés (Visual Touchpad: Malik, Laszlo, 2004): a kéz kontúrján a k-vektorok szögét vizsgálják
  - kis szög $\Rightarrow$ ujjhegy-jelölt
  - a két vektor determinánsa: csúcs vagy völgy
  - nem-maximum elnyomás
  - az ujj iránya: a p_{i-k}, p_{i+k} húr közepétől p_i-ig (sárga)
- Itt: $k = 80$ pont, $\alpha_{ik} < 60°$
  - 5 ujjhegy, 4 völgy, hamis sarok nincs
  - $k = 70, \ldots, 100$ mellett is 5 ujjhegy
  - fehér: k-vektorok (középső ujj)

Kép: Evan-Amos, Wikimedia Commons, CC BY-SA 3.0, módosítva

Tartalom

1 Bináris képek adatstruktúrái

2 Terület alapú alakleírás

3 Kontúr alapú alakleírás

4 Összefoglalás

Az alakleírók összefoglalása

leíró	képlet	jellemzi	megjegyzés
kompaktság	$M_{cmp} = \frac{1}{2\pi}\frac{S}{\mu_{20}+\mu_{02}}$	radiális eloszlás	robusztus
excentricitás	$M_{ect} = \frac{\sqrt{(\mu_{20}-\mu_{02})^2+4\mu_{11}^2}}{\mu_{20}+\mu_{02}}$	elnyújtottság	zajérzékenyebb
orientáció	$\theta = \frac{1}{2}\,\mathrm{atan2}(2\mu_{11}, \mu_{20}-\mu_{02})$	a főtengely iránya	$\bmod \pi$
alaktényező	$F = 4\pi S/L^2$	körszerűség, a kontúr simasága	felbontásfüggő
görbület	$\kappa = d\theta/ds$	lokális alak	simítás kell (skálatér)
***k*-koszinusz**	$c_{ik} = \cos\alpha_{ik}$	kontúrsarkok	k: skála; nem-max. elnyomás

- Adatstruktúrák: képmátrix és futam-hossz kód (terület alapú), lánckód (kontúr alapú); az első lépés a komponens-analízis
- A nyomatékok globális, robusztus leírók; a kontúr alapú leírók a lokális sajátságokat (sarkokat) is megragadják
- Nem tárgyaltuk: Fourier-leírók, harmadrendű invariánsok (Hu), alakzatillesztés