Digitális képfeldolgozás
Kontúrvonal-keresés és elemzés

Horváth András, SZE GIVK

v 0.95

1. Alapok
2. A háttér és a tárgyak szétválasztása
3. Az egybefüggő tartományok megkeresése
4. Az egybefüggő tartományok vizsgálata

Alapok

Gyakori képfeldolgozási probléma: a tárgyakat külön kell választani a háttértől, majd megmérni valamilyen tulajdonságukat.

- adott színű labdák vagy adott alakú jelek helyének megállapítása
- különböző foltok geometriai paramétereinek megmérése
- továbbadás részletes elemzésre (pl. arcfelismerés)

Fő lépések:

1. a háttér és a tárgyak szétválasztása
2. egybefüggő tartományok („foltok”, „kontúrok”) megkeresése
3. az egybefüggő tartományok vizsgálata

A háttér és a tárgyak szétválasztása

Cél: legyen egy „térképünk”, melyen

- 0 van tárolva a „háttér”
- 1 van tárolva a „tárgyak”

pontjainál.

A háttér és a tárgyak szétválasztása

Cél: legyen egy „térképünk”, melyen

- 0 van tárolva a „háttér”
- 1 van tárolva a „tárgyak”

pontjainál.

Nem egyszerű feladat.

Valójában értelmezni kellene a képet: mi az érdekes, mi nem.

Egyszerű esetek: intenzitás vagy szín alapján.

Példa: Sárga és legalább közepesen élénk pixelek maszkja:

```
img_HSV=cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
sarga=(img_HSV[:,:,0]>15) & (img_HSV[:,:,0]<45) & (img_HSV[:,:,1]>100)
sarga=sarga.astype(np.uint8)
```

(Hue értékek felezve vannak!)

Küszöbölés: egyszerű módszerek

Küszöbölés ötlete:

1. kiválasztjuk a számunkra érdekes mennyiséget (pl. intenzitás vagy H-érték)
2. megadunk fix alsó és felső korlátot e mennyiségre: ami nem esik a korlátok közé, az háttér lesz, a többi tárgy.

Küszöbölés: egyszerű módszerek

Küszöbölés ötlete:

1. kiválasztjuk a számunkra érdekes mennyiséget (pl. intenzitás vagy H-érték)
2. megadunk fix alsó és felső korlátot e mennyiségre: ami nem esik a korlátok közé, az háttér lesz, a többi tárgy.

Fő gond: Milyen küszöbértékeket válasszak?

- valaki megmondja? (pl. sárga színek Hue tartománya)
- a lehetséges értékek közepénél? (127)
- az érdekes mennyiség értékeinek átlagát vagy mediánját?
- a hisztogramon középtájról egy értéket, ami csúcsokat vág ketté?
- valami speciális, matematikai eljárásból számoltat?

Küszöbölés OpenCV-ben fix küszöbbel

Példák: sötét háttér előtt világos tárgyakat tételezünk fel.
1. módszer: logikai műveletekkel Pl.:

```
gray_img=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
targyak=(gray_im>127).astype(np.uint8)
```

Küszöbölés OpenCV-ben fix küszöbbel

Példák: sötét háttér előtt világos tárgyakat tételezünk fel.
1. módszer: logikai műveletekkel Pl.:

```
gray_img=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
targyak=(gray_im>127).astype(np.uint8)
```

2. módszer: cv2.threshold() parancs, megadott küszöb

```
lim, targyak=cv2.threshold(gray_img, 127, 255, cv2.THRESH_BINARY)
```

Ez `lim`-be beírja a 127-es értéket és az egészet 255-tel szorozza.

Küszöbölés OpenCV-ben fix küszöbbel

Példák: sötét háttér előtt világos tárgyakat tételezünk fel.
1. módszer: logikai műveletekkel Pl.:

```
gray_img=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
targyak=(gray_im>127).astype(np.uint8)
```

2. módszer: cv2.threshold() parancs, megadott küszöb

```
lim, targyak=cv2.threshold(gray_img, 127, 255, cv2.THRESH_BINARY)
```

Ez `lim`-be beírja a 127-es értéket és az egészet 255-tel szorozza.
3. módszer: cv2.threshold() parancs, számolt küszöb

```
lim, targyak=cv2.threshold(gray_img, 0, 255, cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```

Ez az 'Otsu-módszerrel' számolja a küszöbértéket, amit visszaír `lim`-be.

Küszöbölés OpenCV-ben fix küszöbbel

Példák: sötét háttér előtt világos tárgyakat tételezünk fel.
1. módszer: logikai műveletekkel Pl.:

```
gray_img=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
targyak=(gray_im>127).astype(np.uint8)
```

2. módszer: cv2.threshold() parancs, megadott küszöb

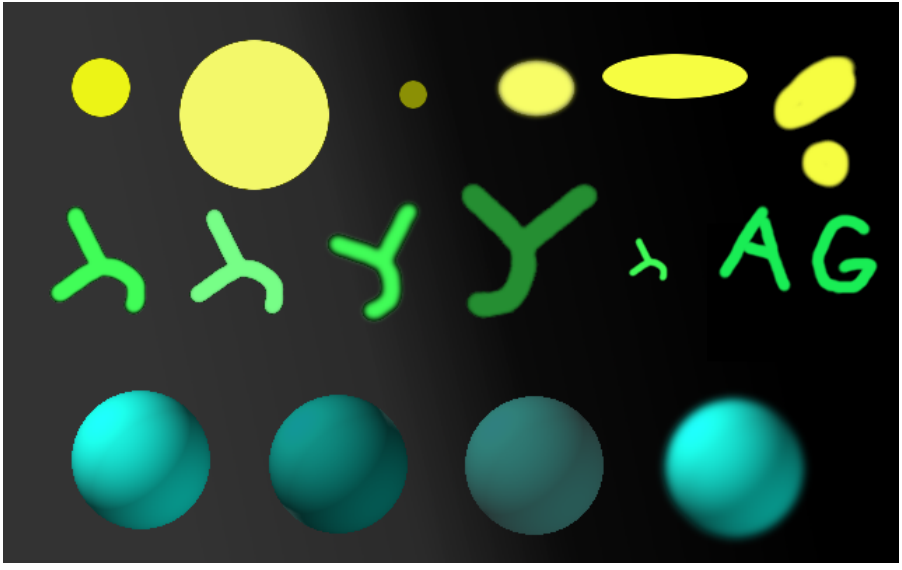
```
lim, targyak=cv2.threshold(gray_img, 127, 255, cv2.THRESH_BINARY)
```

Ez lim-be beírja a 127-es értéket és az egészet 255-tel szorozza.
3. módszer: cv2.threshold() parancs, számolt küszöb

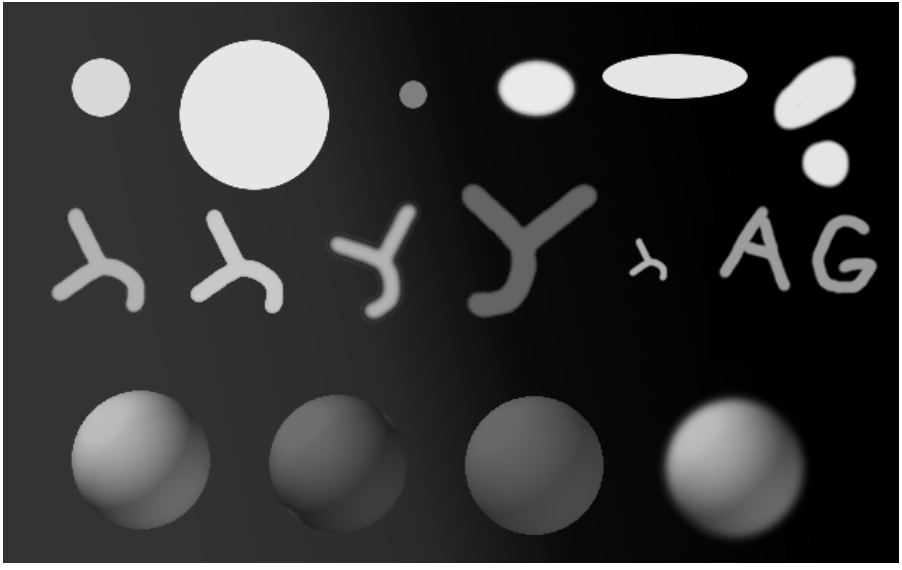
```
lim, targyak=cv2.threshold(gray_img, 0, 255, cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```

Ez az 'Otsu-módszerrel' számolja a küszöbértéket, amit visszaír lim-be.
(cv2.THRESH_BINARY: irány megadás, ha cv2.THRESH_BINARY_INV-re cseréljük, akkor a sötét pixeleket veszi előtérnek.)

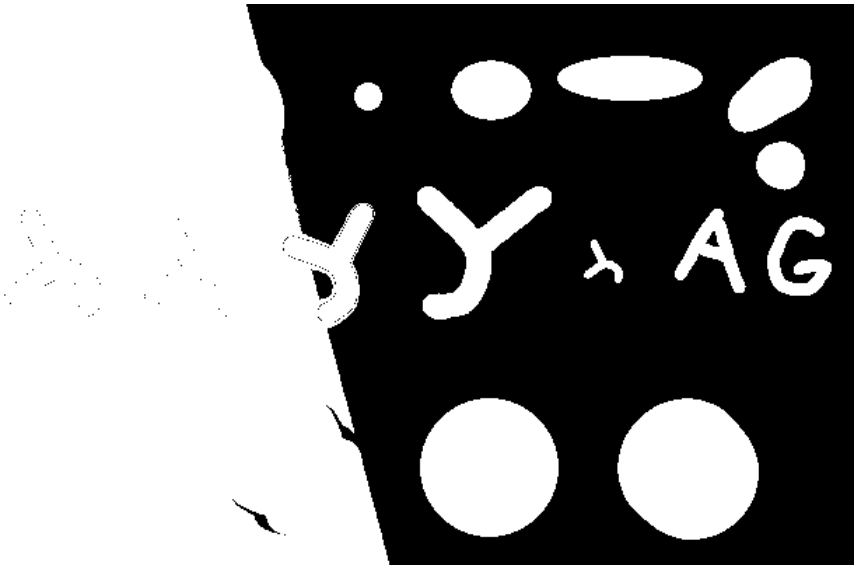
Küszöbölési eljárás (thresholding), mintakép (ezen demonstrálunk)



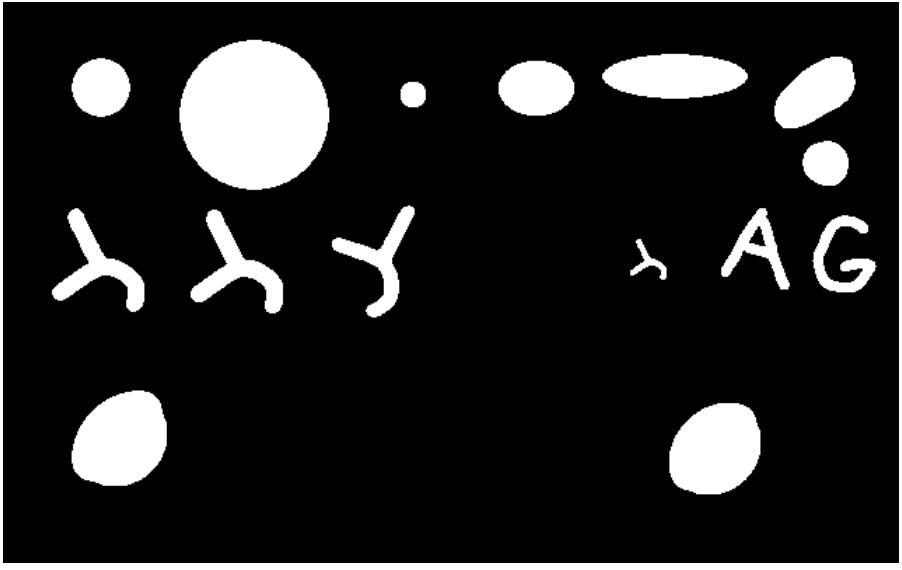
Küszöbölési eljárás: szürkeárnyalatos kép (csak 1 komponens maradt)



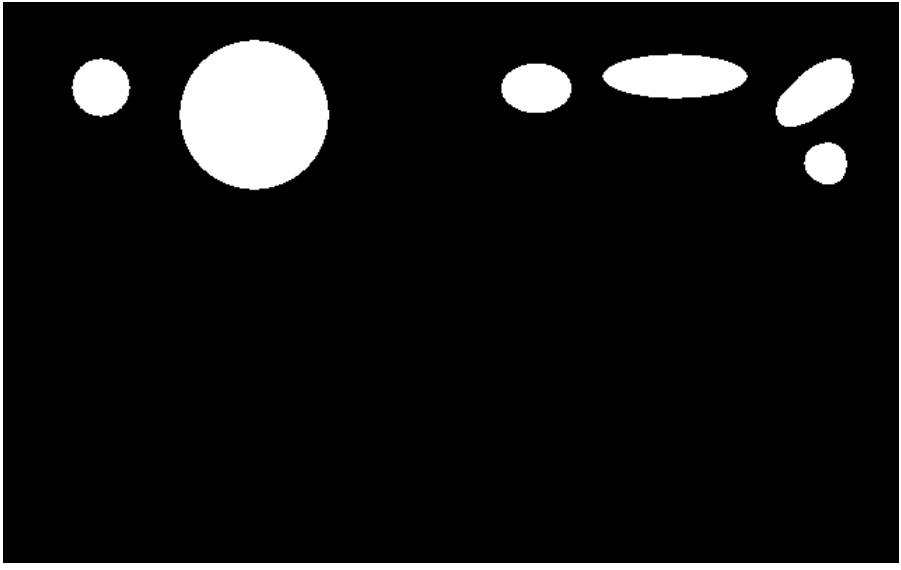
Küszöbölési eljárás: küszöb=40 (a szürkeárnyalatos képen)



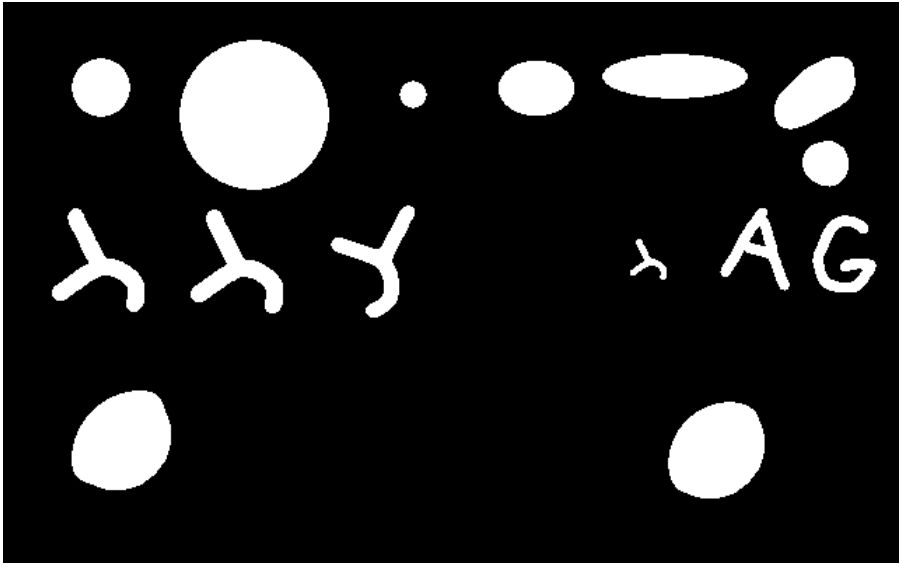
Küszöbölési eljárás: küszöb=120



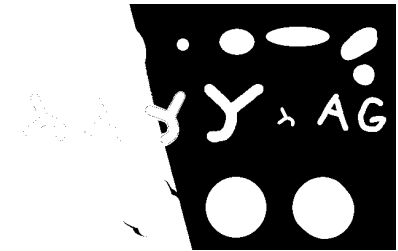
Küszöbölési eljárás: küszöb=200



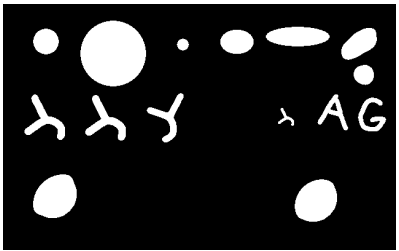
Küszöbölési eljárás: Otsu-módszer (automatikus küszöbérték számolás)



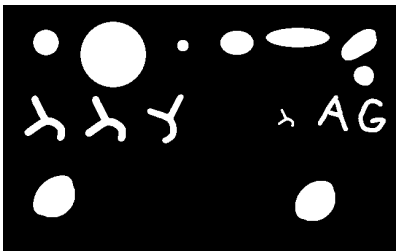
Küszöbölési eljárás: összevetés (nem adaptív módszerek)



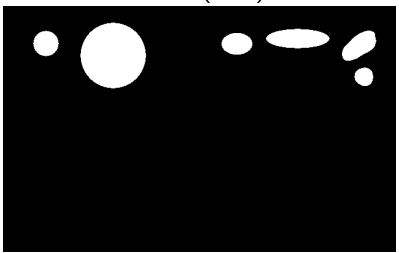
40



Otsu (113)



120



200

Adaptív küszöbérték-választás

A fix küszöbértékek sokszor nem működnek jól.
Pl. egyenetlen háttér-világítás a fenti példákban.
Megoldás: adaptív vágási érték választása.

Adaptív küszöbérték-választás

A fix küszöbértékek sokszor nem működnek jól.
Pl. egyenetlen háttér-világítás a fenti példákban.
Megoldás: adaptív vágási érték választása.

Lehetséges módszerek:

- a képet nagyméretű kernellel elmosni, ezt levonni a képből, és ezen alkalmazni fix vágási értéket
- a képre egyszerű, lassan változó függvényt (pl. polinomot) illeszteni, ezt levonni, és ezen alkalmazni fix vágási értéket
- helyfüggő vágási érték: minden képpont adott sugarú környezetéből számolni pl. Otsu-küszöböt

Adaptív küszöbérték-választás OpenCV-ben

OpenCV módszer:

`cv2.adaptiveThreshold(chan, MAX, SIMÍTÁS, IRÁNY, MÉRET, LIMIT)`

A `chan` csatorna másolatát elsimítja a `SIMÍTÁS` módszerrel `MÉRET` nagyságú kernellel, kivonja belőle az eredeti csatornát és küszöböli a fix `LIMIT` küszöbbel. Aztán az egészet `MAX`-szal szorozza.

Adaptív küszöbérték-választás OpenCV-ben

OpenCV módszer:

`cv2.adaptiveThreshold(chan, MAX, SIMÍTÁS, IRÁNY, MÉRET, LIMIT)`

A `chan` csatorna másolatát elsimítja a `SIMÍTÁS` módszerrel `MÉRET` nagyságú kernellel, kivonja belőle az eredeti csatornát és küszöböli a fix `LIMIT` küszöbbel. Aztán az egészet `MAX`-szal szorozza.

`SIMÍTÁS` lehetséges értékei:

- `cv2.ADAPTIVE_THRESH_MEAN_C`: egyenletes kernel
- `cv2.ADAPTIVE_THRESH_GAUSSIAN_C`: Gauss-kernel

Adaptív küszöbérték-választás OpenCV-ben

OpenCV módszer:

`cv2.adaptiveThreshold(chan, MAX, SIMÍTÁS, IRÁNY, MÉRET, LIMIT)`

A `chan` csatorna másolatát elsimítja a `SIMÍTÁS` módszerrel `MÉRET` nagyságú kernellel, kivonja belőle az eredeti csatornát és küszöböli a fix `LIMIT` küszöbbel. Aztán az egészet `MAX`-szal szorozza.

`SIMÍTÁS` lehetséges értékei:

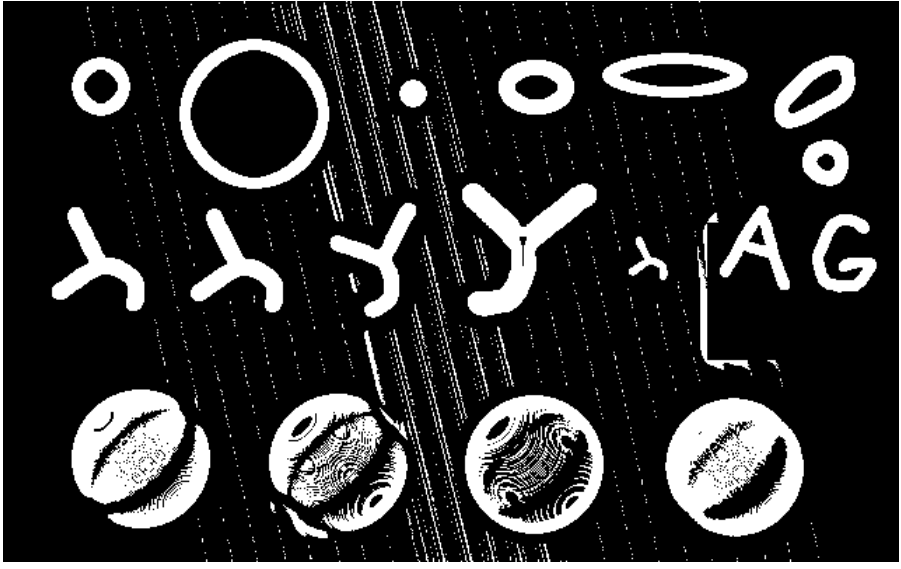
- `cv2.ADAPTIVE_THRESH_MEAN_C`: egyenletes kernel
- `cv2.ADAPTIVE_THRESH_GAUSSIAN_C`: Gauss-kernel

A paraméterek beállítása problémafüggő!

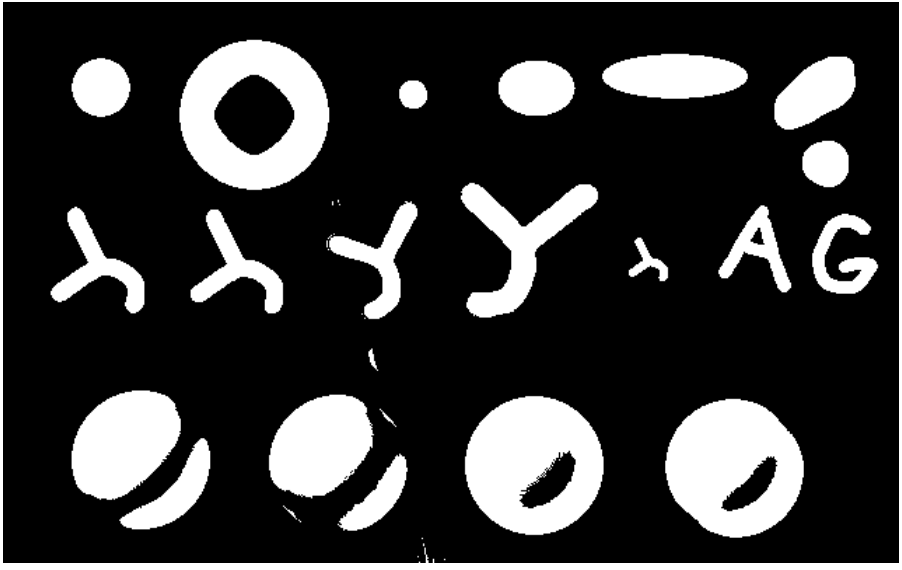
Általában a Gauss-kernel ad jobb eredményt, olyan `MÉRET`-tel, mely kicsivel nagyobb a vizsgált tárgyak méreténél. `LIMIT` értékét próbálgatni kell.

(A módszer buta: a `MÉRET` paraméter kell neki, hogy tudja, mekkora tárgyak felbukkanására számítunk.)

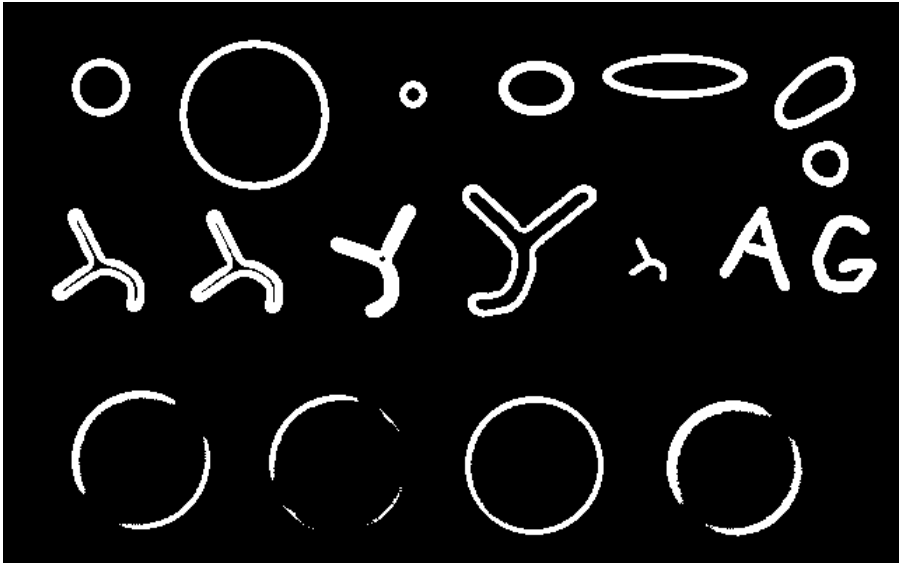
Egyenletes kernel, MÉRET=15, LIMIT=0



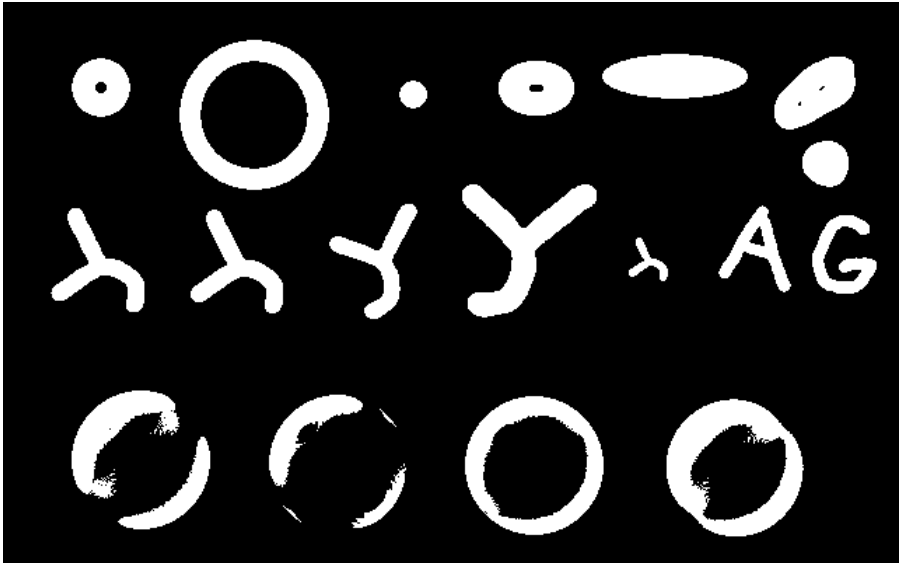
Egyenletes kernel, MÉRET=45, LIMIT=-1



Gauss-kernel, MÉRET=15, LIMIT=-3



Gauss-kernel, MÉRET=45, LIMIT=-3



Küszöbölés tanulság

Nem is olyan egyszerű jól küszöbölni!

Fejlesztési lehetőségek:

- zajszűrés (konvolúcióval)
- a megtalált kép javítása pl. morfológiai operátorokkal
- színek figyelembe vétele
- fejlettebb módszerek, melyek figyelik a közeli tárgyak összetartozását (pl. Canny, WaterShed)
- jobb felvételek készítése

E tárgyban a fentebb bemutatott fix értékű, Otsu és adaptív értékválasztással megoldható problémákkal találkozunk csak.

Tartományok keresése

Többféle stratégia lehet:

- Belső pontok követése:
  Keresünk egy pontot, ami egy tárgyhoz tartozik, majd hozzávesszük a szomszédjait, annak a szomszédjait, ...
  Ezeket a pontokat kizárni a további keresésből és megnézni, van-e még tárgyhoz nem rendelt pont és arra megismételni.
- Kontúrok követése:
  Keresni egy határpontot, majd ennek szomszédos határpontját, majd ennek szomszédját, ...
  Ezt ismételni, amíg van határpont, amit nem rendeltünk kontúrvonalhoz.

Elvileg egyenértékű eredményre jutunk a két módszerrel.

Tartomány-keresési algoritmusok

Megpróbálkozhatunk saját algoritmus írásával, de igen lassú lesz!
Gyors algoritmushoz szükséges:

- fejlett matematikai módszerek alkalmazása
- hatékony implementáció pl. C++-ban

OpenCV kontúr-keresés: `cv2.findContours()`
Sok lehetőség:

- Külső és belső kontúrok is érdekelnek?
- Érdekel, melyik kontúr melyikbe van ágyazva?
- Pontosan követem a kontúrokat vagy csak egy toleranciával?

Időhiány: csak a legegyszerűbb esetekkel foglalkozunk.

Az OpenCV kontúr-keresője

`conts, hier=cv2.findContours(mask, TÍPUS, KÖZELÍTÉS)`

Visszaadott értékek:

- `conts`: a kontúrok listája
  Elemei: pixel-listák a határpontokkal.
- `hier`: kontúr-hierarchia adatbázis; melyik kontúr melyik belsejébe van ágyazva
  Nem foglalkozunk vele ebben a kurzusban.

Az OpenCV kontúr-keresője

`conts, hier=cv2.findContours(mask, TÍPUS, KÖZELÍTÉS)`

Visszaadott értékek:

- `conts`: a kontúrok listája
  Elemei: pixel-listák a határpontokkal.
- `hier`: kontúr-hierarchia adatbázis; melyik kontúr melyik belsejébe van ágyazva
  Nem foglalkozunk vele ebben a kurzusban.

`TÍPUS` értékei:

- `cv2.RETR_LIST`: egyszerű lista
- `cv2.RETR_EXTRENAL`: egyszerű lista csak a külső kontúrokról
- `cv2.RETR_CCOMP`, `cv2.RETR_TREE`: összetett struktúra
  (itt nem tárgyaljuk)

Az OpenCV kontúr-keresője

`conts, hier=cv2.findContours(mask, TÍPUS, KÖZELÍTÉS)`

Visszaadott értékek:

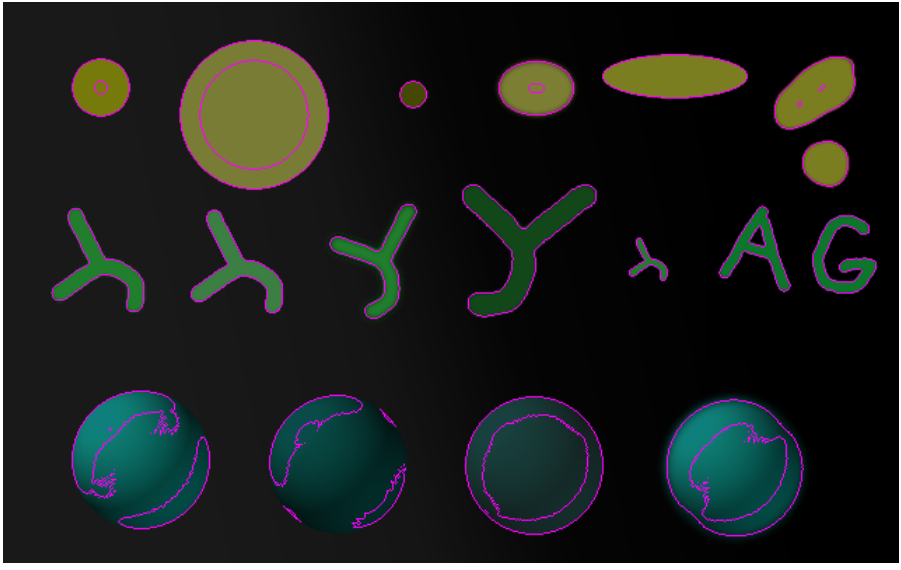
- `conts`: a kontúrok listája
  Elemei: pixel-listák a határpontokkal.
- `hier`: kontúr-hierarchia adatbázis; melyik kontúr melyik belsejébe van ágyazva
  Nem foglalkozunk vele ebben a kurzusban.

`TÍPUS` értékei:

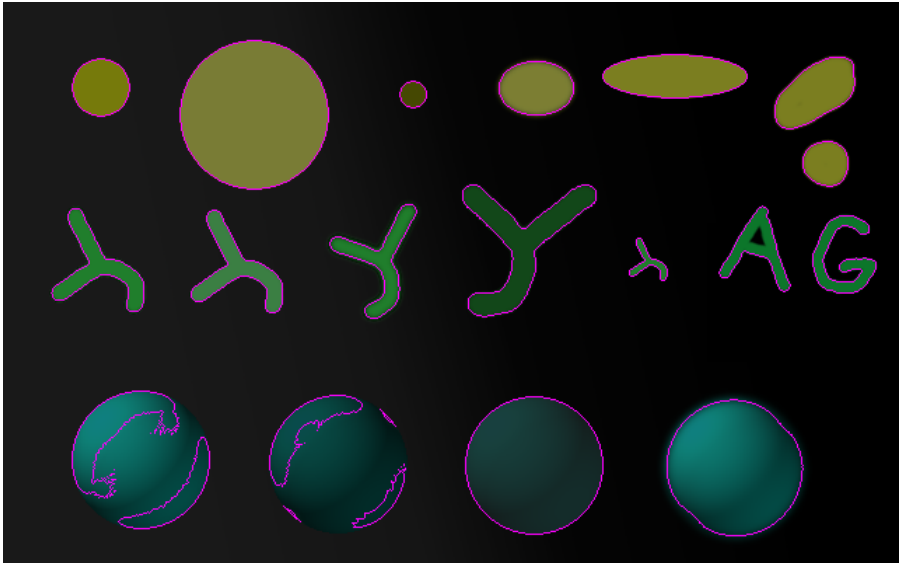
- `cv2.RETR_LIST`: egyszerű lista
- `cv2.RETR_EXTRENAL`: egyszerű lista csak a külső kontúrokról
- `cv2.RETR_CCOMP`, `cv2.RETR_TREE`: összetett struktúra
  (itt nem tárgyaljuk)

`KÖZELÍTÉS`: Ebben a kurzusban legyen `cv2.CHAIN_APPROX_NONE`
Ez azt jelenti, hogy nem alkalmazunk közelítést.

Kontúrok keresése: külső és belső



Kontúrok keresése: csak külső



Mit kezdjünk a megtalált kontúrokkal?

1. Megmérjük bizonyos paramétereit.
   Pl.: terület, kerület, körülírt kör sugara, eloszlás-momentumok, ...
2. Kiválasztjuk a nekünk érdekeseket.
   Pl.: kizárjuk a túl kicsiket vagy nagyokat, amik nem elég kör-szerűek, stb.
3. Az „érdekeseket” megjelenítjük, vagy fontos paramétereit listázzuk.
   Pl.: középpont-koordináták listája, színezett ábrák, ...

Lásd: mintaprogram.

Néhány nem triviális paraméter

terület/kerület: „átlagos vastagság”

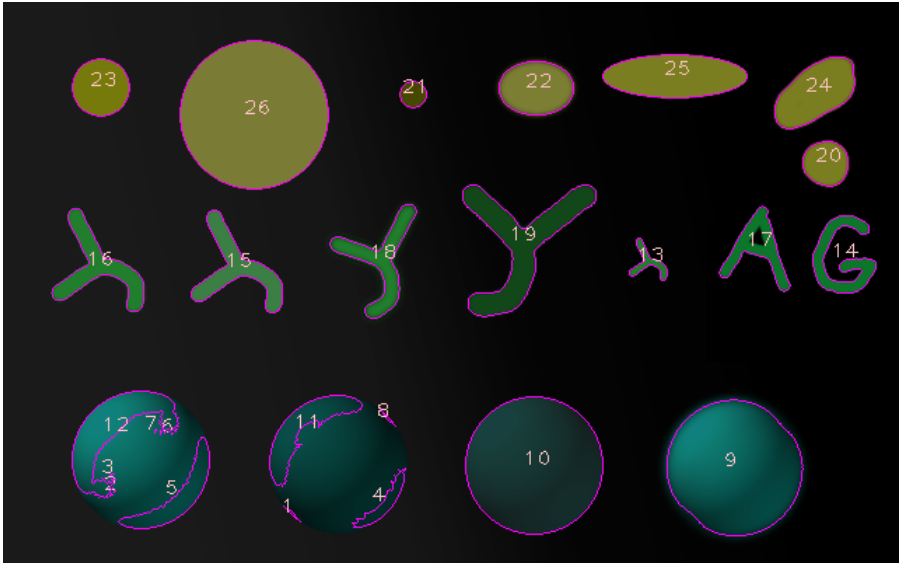
körszerűség: 4π(terület/kerület^2).
Ez csak kör esetén 1, minden más esetben 1-nél kisebb.

tömörség: terület/konvex-burok-területe.
Ez konvex alakzatra 1, más esetekben 1-nél kisebb. (Minél ágas-bogasabb, annál kisebb.)

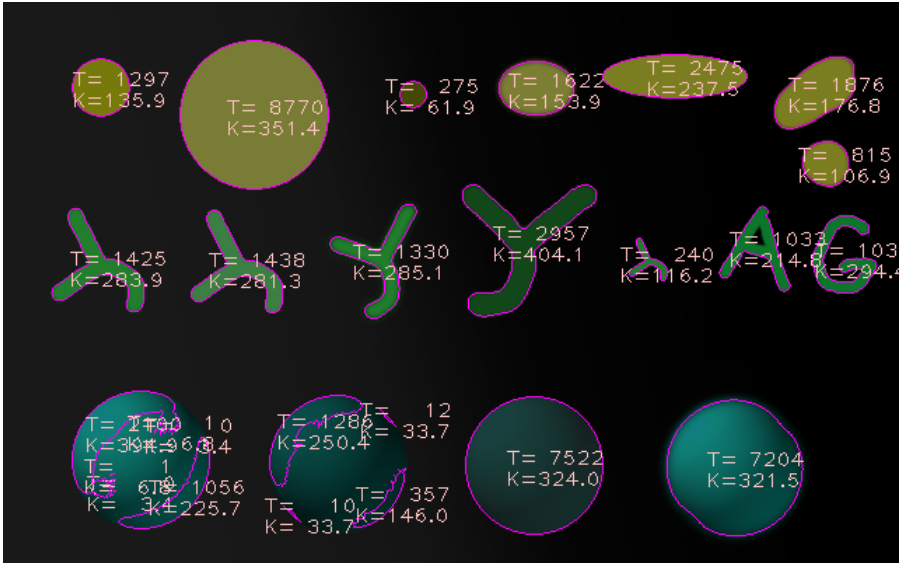
legkisebb körülírt kör sugara:

határhoz illesztett ellipszis méretei:

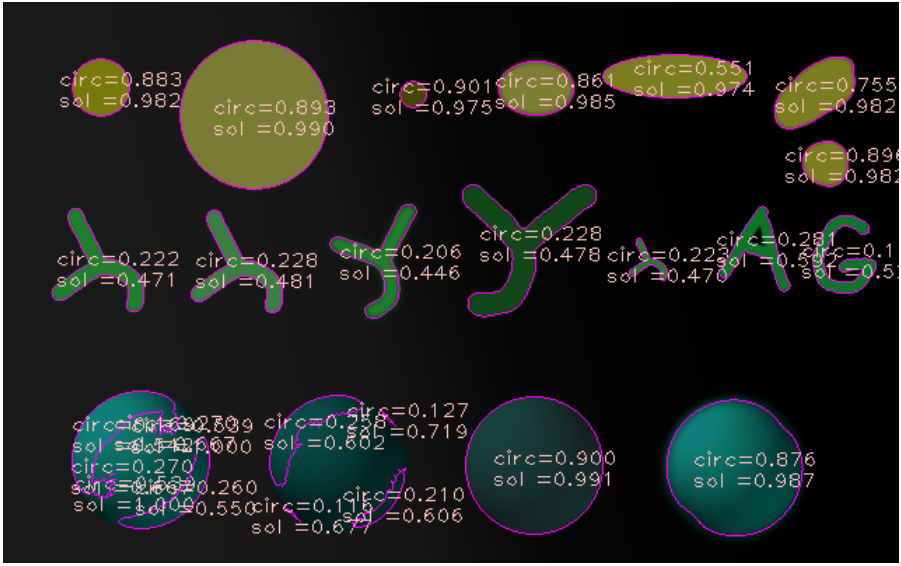
Kontúrok paraméterei: sorszámok



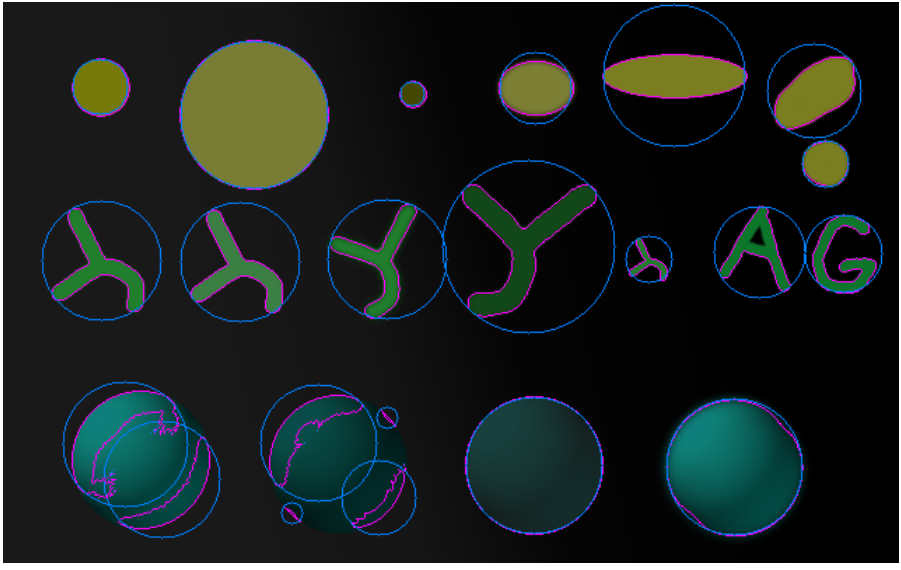
Kontúrok paraméterei: Kerület, terület



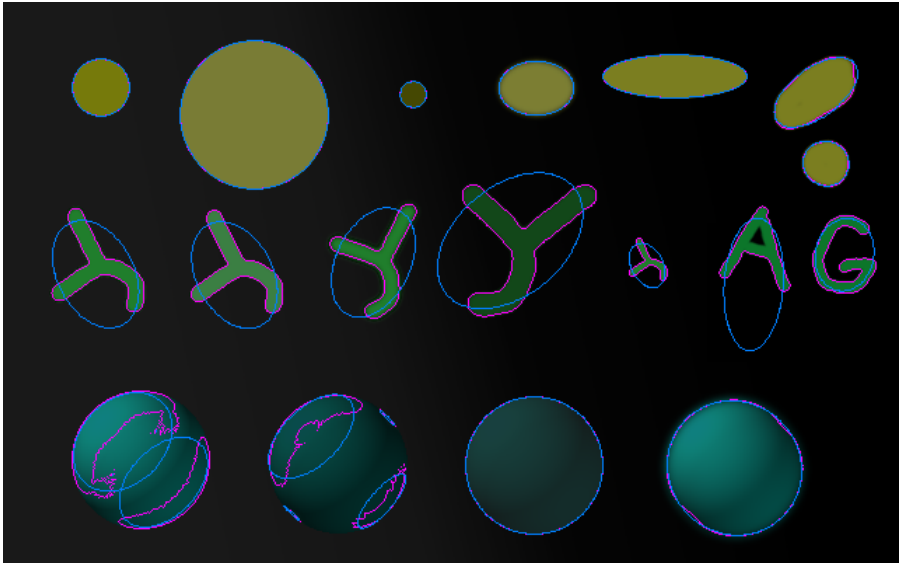
Kontúrok paraméterei: Körszerűség és tömörség



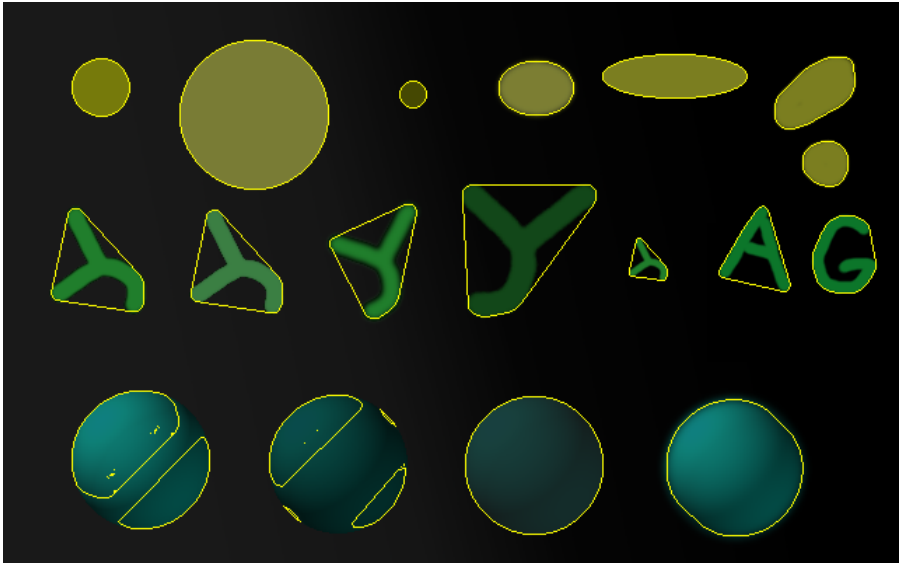
Kontúrok paraméterei: Körülírt legkisebb kör



Kontúrok paraméterei: Illesztett ellipszis



Kontúrok paraméterei: Konvex burok



Momentumok

Az alakzatot jellemző számok, melyek a valószínűségi eloszlások momentumainak felelnek meg.

$$\mathtt{m}_{ji} = \sum_{x,y} \left(\mathtt{image}(x,y) \cdot x^j \cdot y^i \right)$$

Az első néhány szemléletes jelentésű:

- $m_{0,0}$: terület
- $m_{1,0}/m_{0,0}$: tömegközéppont x koordinátája
- $m_{0,1}/m_{0,0}$: tömegközéppont y koordinátája

A többinek bonyolultabb a jelentése, pl. a másodrendűek a szórással kapcsolatosak.
(Nem fogjuk közvetlenül használni őket.)

Hu-momentumok

A fenti momentumokból számolható 7 momentum, melyek nem változnak, ha az alakzatot elforgatjuk, eltoljuk, nagyítjuk.
Számítási módjuk összetett (de az OpenCV tudja).

Felhasználásuk: Két alakzat hasonlónak mondható, ha Hu-momentumaik megegyeznek.

Tanítás: megadunk egy mintát, kiszámoljuk Hu-momentumait és eltároljuk.
Felismerés: a talált alakzatok Hu-momentumait összevetjük a fent eltárolttal. Ha jó az egyezés, akkor hasonló az alak, csak esetleg el van forgatva vagy más méretben szerepel.

Kontúrok paraméterei: Hu-momentumok