Hallgatói segédlet

Numpy alapok

Létrehozás, inicializálás

Numpy tömb inicializálásra mutat példákat a következő kódrészlet:

```
import numpy as np
cols = int(10)
rows = int(10)
value = int(5)

name = np.asarray((rows,cols),dtype=np.uint8)    #numpy tömb létrehozása
name = np.full((rows,cols),value,dtype=np.uint8) #numpy tömb létrehozása kezdőértékkel
```

Műveletek

Numpy tömb adott elemeinek elérése:

```
import numpy as np

cols = int(10)
rows = int(10)
dim = int(3)

name = np.asarray((rows,cols),dtype=np.uint8)    #numpy tömb létrehozása
name2= np.asarray((rows,cols,dim),dtype=np.uint8)    #numpy színes tömb létrehozása

name.shape # a tömb méretének lekérdezése
name[0,0] #adott elem elerese
name[0,:] #adott sor eseten az osszes oszlopelem elerese
name[:,0] #adott oszlop eseten az osszes sorelem elerese
name[:,:] # az osszes elem elerese a tombben

name2[:,:,0] # az elso csatorna elerese színes kép esetében
```

Képekkel kapcsolatos műveletek

Kép beolvasása:

```
import cv2

open_mode = 1; // -1: modositatlan ertekek, 0: fekete-fehér mód, 1:színes mód
Image = cv2.imread("kep_eleresi_utvonala",open_mode);
```

Kép kiírása:

```
import cv2

formatParameters_png = list()
formatParameters_jpg = list()
formatParameters_png.insert(cv2.IMWRITE_PNG_COMPRESSION)
```

```
formatParameters_png.insert(5)

formatParameters_jpg.insert(cv2.IMWRITE_JPEG_QUALITY)
formatParameters_jpg.insert(90)

cv2.imwrite("out.png", I, formatParameters_png)
cv2.imwrite("out.jpg", I2, formatParameters_jpg)
```

Kiíráskor szükséges egy listában felsorolni a kimeneti formátum jellemzőit.

Színterek

RGB

Három szín alapján keveri ki a végső színt. A három alapszín inexekkel együtt: Kék (0), Zöld(1), Piros(2)

HSV

Itt is három csatorna van, csak a jelentésük más:

- HUE: színérték
- SATURATION: mennyi a szürke aránya a színkomponensben
- VALUE: világosságérték

Konverziók

```
cv2.cvtColor(I, gray, cv2.COLOR_RGB2GRAY);
cv2.cvtColor(I, hsv, cv2.COLOR_RGB2HSV);
cv2.cvtColor(I, hsv, cv2.COLOR_HSV2RGB);
```

Képjavító módszerek

Szűrők,eljárások

Életlenítő szűrők (box,gauss,median):

```
value = 3 #always odd
sigma_value = 2
output = cv2.boxFilter(input, -1, (value, value));
output = cv2.GaussianBlur(input, (value, value),sigma_value);
output = cv2.medianBlur(input, value);
```

A box átlagértéket, a gauss a görbe által meghatározott értéket, a medián középértéket számol.

Hisztogram-kiegyenlítés:

```
cv2.equalizeHist(input, output);
```

Gyenge kontrasztú képek esetében alkalmazzuk.

Élesítés:

```
image = cv2.imread("test.jpg", 0);

blurred = cv2.GaussianBlur(image, (sizeofKernel, sizeofKernel), 1);
unsharped = cv2.addWeighted(image, 1.5, blurred, -0.5, 0,unsharped);
```

Itt a lényeg, hogy először tetszőleges szűrővel kiszámoljuk az eredeti kép homályosabb verzióját, majd kivonjuk az eredeti képből, ezzel megkapjuk az élesebb változatot. Az összegzés során a súlyok előjelei ellentétesek, és nagyon fontos, hogy az előjeles összegük egyenlő legyen **1-gyel**.

Küszöbölés

Sima globális küszöbölés:

```
thresh = cv2.threshold(Img, thresh_value, 255, cv2.THRESH_BINARY);
thresh = cv2.threshold(Img, thresh_value, 255, cv2.THRESH_BINARY_INV);
```

Globális küszöbölés algoritmussal (Otsu):

```
thresh = cv2.threshold(Img, 0,255,cv2.THRESH_BINARY | cv2.THRESH_OTSU);
```

Adaptív küszöbölés:

```
blockSize = 5;
thresh = cv2.adaptiveThreshold(Img, 255, cv2.ADAPTIVE_THRESH_MEAN_C,
cv2.THRESH_BINARY,blockSize,2);
```

Morfológia

Alapműveletek

Dilatáció, erózió:

```
Img = cv2.imread("morp_test.png", 0);
kernel = np.ones((5, 5), dtype=np.uint8);

dilated = cv2.dilate(Img, kernel);
eroded = cv2.erode(Img, kernel);
```

Nyitás:

```
Img = cv2.imread("morp_test_open.png", 0);
kernel = np.ones((3, 3), dtype=np.uint8);

eroded = cv2.erode(Img, kernel)
opened = cv2.dilate(eroded, kernel)
```

Zárás:

```
Img = cv2.imread("morp_test_closed.png", 0);
kernel = np.ones((7, 7), dtype=uint8);
```

```
dilated = cv2.dilate(Img, kernel);
closed = cv2.erode(dilated, kernel);
```

Struktúrális elem(kernel)

Deklarációk:

```
kernel = np.ones((7, 7), dtype=np.uint8); // sima 7x7es négyzet alakú
kernel = cv2.getStructuringElement(1, (sizeofKernel,sizeofKernel));
```

Lehet használni a beépített metódust is, az első paraméter megadja a struktúrális elem típusát:

- 0: háromszög, cv2.MORPH_RECT
- 1: kereszt, cv2.MORPH_CROSS
- 2: ellipszis, cv2.MORPH_ELLIPSE

Összetettebb műveletek

Kontúr:

```
image = cv2.imread("test.jpg", 0);
kernel = cv2.getStructuringElement(1, (sizeofKernel,sizeofKernel));

dilated = cv2.dilate(image, kernel);
eroded = cv2.erode(image, kernel);
contures = dilated - eroded;
```

Tophat (csúcsok detektálása):

```
sizeofKernel = 10;
image = cv2.imread("test.jpg", 0);
kernel = cv2.getStructuringElement(1, (sizeofKernel, sizeofKernel));

eroded = cv2.erode(image, kernel);
opened = cv2.dilate(eroded, kernel);
tophat = image - opened;
```

Blackhat (völgyrészek detektálása):

```
image = cv2.imread("test.jpg", 0);
kernel = cv2.getStructuringElement(1, (sizeofKernel, sizeofKernel));

dilated = cv2.dilate(image, kernel);
closed = cv2.erode(dilated, kernel);
blackhat = closed - image;
```