

# **Gépilátás beadandó**

*2019/2020/2*

**Feladat: Rubik kocka állapotának  
felismerése**

**Készítette: Gvárdián Levente**

Neptunkód:U7A44W

# Bevezetés

Rubik kocka felismerő alkalmazást választottam féléves feladatnak. Ez az alkalmazás 6 fényképet olvas be, amin a rubik kocka minden egyes oldala külön-külön található meg, és ezután átkonvertálja a képet HSV színtérbe, ez alapján készít maszkot, és minden maszkra külön kontúrkereséssel, majd megrajzolással készít az eredetileg beadott képre kontúrokat, azok köré virtuálisan egy kör alakú kontúrt képez, és ennek a körnek a középpontját olvassa be. Ekkor egy koordinátát kapunk, illetve egyből a színt is elmenti egy kétdimenziós tömbbe, majd ezután ezt az x,y,szín koordinátát sorbarendezzük y majd x szerint, és így megkapjuk a kockákat sorrendben balról jobbra haladva. Ezt már csak kiíratással a végén kiíratam a konzolba.

# Elméleti háttér

Elméleti háttére a feladatomnak, hogy először is ugye a képeket az opencv segítségével beolvastatom:

```
cv2.imread("kép.jpg")
```

A beolvasott kép formátumát átalakítom szintén az opencv segítségével:

```
cv2.resize(kep,(1920,1080))
```

Mivel a képet RGB, színrendszerben kapom meg(Ez a programban BGR),ezért az egyeztetett első konzultáció alapján, másik színrendszert kerestem, és kis internetes kutatás után HSV színrendszert találtam a legmegfelelőbbnek.

```
hsv = cv2.cvtColor(kep,cv2.COLOR_BGR2HSV)
```

Ez sikerült, és átkonvertálta a képet a program a számunkra megfelelő színrendszerbe, ekkor következett a maszkolás problémája. Hiszen szükséges tudni a kívánt szín maszkolásához a szín HSV-ben megadott határértékeit, amit lower, illetve upper előtaggal adtam meg. Ehhez egy kis interneten talált program általi segítséget kellett alkalmaznom, hogy ezt a lower,és upper határokat megtaláljam minden színhez HSV szerint.(A dokumentáció legvégén írom le, hogy hol található ez a program)

```
95 lowup =np.array ([[42,55,60],  
96                  [93, 255, 255],  
97                  [100,151,200],  
98                  [106,255,255],  
99                  [171,130,0],  
100                 [255,255,255],  
101                 [21,121,144],  
102                 [255,255,196],  
103                 [0,86,169],  
104                 [63,255,255],  
105                 [62,0,130],  
106                 [255,255,255]])
```

-Ez két soronként értelmezendő. Két soronként változik, hogy melyik színt nézi, például az első két sor az a zöld szín lower és upper értékei, a következő két sor a kék lower és upper értékei, stb. stb. egészen a fehér színnel bezárólag.

Maszkolás pedig színenként történik ezzel a kódsorral:

```
mask = cv2.inRange(hsv,lowup[i],lowup[i+1])
```

Ezt követően jött a következő probléma, hogy hogyan határozzam meg hol van melyik kocka, illetve milyen színű az a kocka. Ehhez ismételten a konzultáció alapján használtam a kontúrozást.

```
contours, _ = cv2.findContours(mask, cv2.RETR_LIST,  
cv2.CHAIN_APPROX_NONE)
```

A koordinátákat pedig úgy kapom meg, hogy mivel a kontúrrajzolás az abban merül ki, hogy a képen látható 9 kockára a maszk alapján rajzolja a négyzet kontúrokat.

```
x, y, w, h = cv2.boundingRect(cnt)
```

```
cv2.rectangle(kep, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

Eköré egy függvény segítségével a lehető legkisebb területű kört “rajzoltattam” virtuálisan:

```
circles = [cv2.minEnclosingCircle(cnt)]
```

Ennek a körnek megnézem a középpontját, és az a középpont visszaadott értéke lesz az x és y koordináta. Ehhez tömböt használtam, ami egy 9x3-mas mátrix, hiszen mind a 9 kockának van x,y koordinátája, és egy színe, amit szintén eltároltam az alapján, hogy a maszkolás for ciklusa hányszor futott le, hiszen ahányszor lefutott már a for ciklus, annyiadik szín kerül maszkolásra, ami alapján a kontúrt rajzolja a program.

```
141 M2 = cv2.moments(cnt)  
142 #print("Momentum2:", M2)  
143 center2 = (int(M2["m10"] / M2["m00"]), int(M2["m01"] / M2["m00"]))  
144 #print("circlecenter:", center2)  
145 cx= int(center2[0])  
146 cy = int(center2[1])  
147 cxcyj=np.empty(3)  
148 cxcyj[0] = cx  
149 cxcyj[1] = cy  
150 cxcyj[2] = j  
151 #print("cxcyj:", cxcyj)  
152 cszintomb = np.append(cszintomb, cxcyj)  
153 print("cszintomb:", cszintomb)
```

Amint ez megvan, és ezeket eltudom tárolni, akkor szükségem van egy sorrendbe rendező algoritmusra, amely a tömb elemeit az x,y koordináta alapján rendezi el, úgy hogy a kockákat balról jobbra kapjuk meg.

```
160     print("cszintomb a vegen:", cszintomb)
161     #segedtömb segítségével csinállok ebből egy 9x3 mátrixot
162     segedt = np.ndarray(shape=(9),dtype=[('x','f4'),('y','f4'),('c','f4')])
163     print("segedt", segedt)
164     l = 0
165     for t in range(0,9):
166         for u in range(0,3):
167             segedt[t][u] = cszintomb[l]
168             l = l+1
169
170     print("segedt:", segedt)
171     #y koordináta szerint sorbarendezelem (axis=0)
172     asd = sorted(segedt, key=lambda segedt_entry: (segedt_entry[1],segedt_entry[0]))
173     print("sorted:", asd)
174
```

Ha ezt is megkaptuk, akkor már csak kiíratjuk sorrendben a színeket, és kész a program.

```
175     o=1
176     for z in range(9):
177         for q in range(3):
178             if (szin(asd[z][q])=="G"):
179                 print('G',end='')
180             elif (szin(asd[z][q])=="B"):
181                 print('B',end='')
182             elif (szin(asd[z][q])=="R"):
183                 print('R',end='')
184             elif (szin(asd[z][q])=="Y"):
185                 print('Y',end='')
186             elif (szin(asd[z][q])=="O"):
187                 print('O',end='')
188             elif (szin(asd[z][q])=="W"):
189                 print('W',end='')
190         if (o%3)==0:
191             print("\n")
192         o=o+1
```

## Megvalósítás terve és kivitelezése

Megvalósítás első része a kép beolvasása. Először is ugye szükségem volt a numpy és opencv plugin segítségével. Ezeket beimportáltam a program elején. A képeket beolvastatom a program elején, majd HSV-be konvertálom a cv2.cvtColor segítségével. Ha ezzel kész van, akkor következik egy forciklus ami azt fogja jelenteni, hogy mind a hat maszkra (mivel hat szín van a rubikkockán, és színenként szeretnénk maszkolni) a cv2.inRange segítségével maskot készítünk, ami után egyből következik a kontúrokkal való foglalkozásnak a része. Itt először is a kontúrokat megkeresi a program, és szépen végig rajzolja, de egy if segítségével a kis területű kontúrokat ki tudjuk szűrni, tehát aminek területe (pixelben) nagyobb mint 10000, akkor megrajzolja nekünk a kontúrt. Eköré a kontúr köré készítünk egy kör alakú kontúr, ami pontosan ráilleszkedik a rajzolt kontúr négyzetre, és ennek a körnek a közepe fogja nekünk megadni a koordinátákat, tehát a kör középpontja alapján határozom meg a kockákat. Ez azért is fontos, mivel különben nem tudnánk a végén a kiíratásnál, hogy melyik kockát szeretnénk kiíratni. Ezt követően egy kétdimenziós tömbbe tárolom el, aminek 9 sora és 3 oszlópa van, hiszen minden kockának van egy koordinátája, amit a kör alapján határoztunk meg, illetve minden kockának van egy színe, amit egy "j" változó segítségével határozok meg. Ehhez szükséges tudni, hogy hányadik for ciklusban járunk a maszkolásnak, így a "j" változót minden maszk forciklus végén növeltem eggyel. Tehát ez alapján tudjuk, mivel a színek így vannak sorrendben: zöld, kék, piros, sárga, narancs, fehér. Hiszen ezeket én adtam meg a lowup tömbnél, aminek a célja ugye a színek kiszűrése HSV segítségével, ezt használja a maszk is. Szóval ez is eltárolódik a cszintomb-ben tehát 9x3 mátrix szükséges, hogy mind a megtalált kilenc kockánknak x,y koordinátáját, és színének értékét(ami egy int, de mivel bekerül a tömbbe ezáltal float típusú lesz a koordinátákkal együtt). Ezt követően jön a segéd tömb bevezetése, ami ahhoz kell, hogy a cszintomb-ből külön szedjem a float típusokat külön értékekké, hiszen a cszintomb nem teljesen 9x3-mas mátrix, hanem csak egy float tömb ami 1x27, hiszen az np.append segítségével adtam hozzá a dolgokat. Visszatérve, ebből egy mátrix lesz a segéd tömb, azaz a programomban segédet alapján. Viszont ez így nem maradhat, hiszen koordináták alapján szeretnénk ezt sort-olni, tehát rendezni, és ezt is úgy, hogy először az y, majd x koordináták alapján. Így megkapva mind a 9 kockát balról jobbra haladva, fentről lefelé. Ezt követően már csak annyi a feladat, hogy ezt konzolba kiírjuk, tehát megadja, hogy melyik szín, hol van. Ezt leegyszerűsítettem: G,B,R,Y,O,W betűkre, hiszen így is tökéletesen érzékelhető, hogy a programnak mi az eredménye. Sajnos nem sikerült megoldani, hogy mind a 6 képet amit beadunk neki, azt automatikusan magától egyesével végig menjen rajtuk, de még utólag próbálkozom vele, hátha azt is sikerül, és akkor majd feltöltöm külön kódfájlként.

# Tesztelés működő képekkel

Tesztelésem során megadtam a következő képeket:



1.ábra: zold.jpg, 2.ábra: kek.jpg, 3.ábra: piros.jpg, 4.ábra: sarga.jpg, 5.ábra: narancs.jpg, 6.ábra: fehér.jpg

## **Eredménye a tesztelésnek:**

### **Jelentése az eredményen látható betűknek:**

G->Green, ami magyarul a zöld színt jelenti

B->Blue, ami magyarul a kék színt jelenti

R->Red, ami magyarul a piros színt jelenti

Y->Yellow, ami magyarul a sárga színt jelenti

O->Orange, ami magyarul a narancs színt jelenti

W->White, ami magyarul a fehér színt jelenti

Fontos, hogy minden eredmény egy 3x3-mas mátrix, amiben 9db betűjelet fogunk látni, attól függően természetesen, hogy mi az eredmény.

## 1. Kép tesztelése: (zold.jpg)

```
Thonny - C:\Users\event\Desktop\lin\Segitaski\beadando\verzi02.py @ 188:41
File Edit View Run Device Tools Help

beadando\verzi02.py
125 print("jelejen:",j)
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 100000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL MÓDOSÍTANI rectangle(IDE ÍRNI A KÉP NEVÉT)!!!!!!!!!!!!!!!!!!!!
131         cv2.rectangle(kepvalasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         #m = cv2.moments(cnt)
133         #print("Moments:", m)
134         #cx = (m['m10']/m['m00'])
135         #print("cx:", cx)
136         #cy = (m['m01']/m['m00'])
137         #print("cy:", cy)
138         #center = np.array([cx, cy])
139         #print("momentscenter:", center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #print("center:", center)

Shell
jvegen: 6
cazintomb a vegen: [1485. 671. 0. 942. 669. 0. 394. 668. 0. 1468. 506. 0.
949. 502. 0. 425. 502. 0. 1456. 354. 0. 957. 351. 0.
450. 345. 0. 1499. 671. 3. 939. 669. 3. 397. 668. 3.
949. 501. 3. 1476. 505. 3. 426. 501. 3. 1464. 352. 3.
959. 351. 3. 473. 346. 3. 407. 667. 5. 959. 669. 5.
950. 501. 5. 427. 501. 5. 960. 351. 5. 1491. 520. 5.
474. 346. 5.]
segedt [(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)]
segedt: [(1485., 671., 0.) ( 942., 669., 0.) ( 394., 668., 0.) (1468., 506., 0.)
( 949., 502., 0.) ( 425., 502., 0.) (1456., 354., 0.) ( 957., 351., 0.)
( 450., 345., 0.)]
sorted: [(450., 345., 0.), (957., 351., 0.), (1456., 354., 0.), (425., 502., 0.), (949., 502., 0.), (1468., 506., 0.), (394., 668., 0.), (942., 669., 0.), (1485., 671., 0.)]

GGG
GGG
GGG

r elejen: 2
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
cazintomb: [461. 629. 1.]
ki: 1
cazintomb: [4.610e+02 6.290e+02 1.000e+00 1.455e+03 6.240e+02 1.000e+00]
ki: 2
cazintomb: [4.610e+02 6.290e+02 1.000e+00 1.455e+03 6.240e+02 1.000e+00 9.590e+02
6.250e+02 1.000e+00]
```

Nagyítva az eredmény:

```
Shell x
949. 502. 0. 425. 502. 0. 1456. 354. 0. 957. 351. 0.
450. 345. 0. 1499. 671. 3. 939. 669. 3. 397. 668. 3.
949. 501. 3. 1476. 505. 3. 426. 501. 3. 1464. 352. 3.
959. 351. 3. 473. 346. 3. 407. 667. 5. 959. 669. 5.
950. 501. 5. 427. 501. 5. 960. 351. 5. 1491. 520. 5.
474. 346. 5.]
segedt [(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)]
segedt: [(1485., 671., 0.) ( 942., 669., 0.) ( 394., 668., 0.) (1468., 506., 0.)
( 949., 502., 0.) ( 425., 502., 0.) (1456., 354., 0.) ( 957., 351., 0.)
( 450., 345., 0.)]
sorted: [(450., 345., 0.), (957., 351., 0.), (1456., 354., 0.), (425., 502., 0.), (949., 502., 0.), (1468., 506., 0.), (394., 668., 0.), (942., 669., 0.), (1485., 671., 0.)]

GGG
GGG
GGG

r elejen: 2
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
cazintomb: [461. 629. 1.]
```



## 2. Kép tesztelése: (kek.jpg)

```
Thonny - C:\Users\Levente\Desktop\Un\Gépiatan\beadandoversio2.py @ 188:41
File Edit View Run Device Tools Help

beadandoversio2.py
125 print("jelejen:",j)
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 10000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL HÖDÖSÍTANI rectangle(IDE IRNI A KÉP NEVÉT)!!!!!!!!!!!!!!!!!!!!!!
131         cv2.rectangle(kepvalasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         M = cv2.moments(cnt)
133         #print("moment:", M)
134         #cx = (M["m10"] / M["m00"])
135         #print("cx:", cx)
136         #cy = (M["m01"] / M["m00"])
137         #print("cy:", cy)
138         #center = np.array([cx, cy])
139         #print("momentcenter:", center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #...

Shell
1.000e+00 1.428e+02 4.760e+02 1.000e+00 1.405e+03 3.370e+02 1.000e+00
9.590e+02 3.280e+02 1.000e+00 4.890e+02 3.360e+02 1.000e+00 4.700e+02
6.200e+02 5.000e+00 9.710e+02 6.240e+02 5.000e+00 9.530e+02 4.750e+02
5.000e+00 4.880e+02 4.800e+02 5.000e+00 4.890e+02 3.360e+02 5.000e+00
1.443e+03 4.860e+02 5.000e+00 9.590e+02 3.360e+02 5.000e+00]
segedt [(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
segedt: [( 461., 629., 1.) (1455., 624., 1.) ( 959., 625., 1.) ( 478., 480., 1.)
( 950., 476., 1.) (1428., 476., 1.) (1405., 337., 1.) ( 959., 338., 1.)
( 489., 336., 1.)]
sorted: [(489., 336., 1.), (1405., 337., 1.), (959., 338., 1.), (950., 476., 1.), (1428., 476., 1.), (478., 480., 1.), (1455., 624., 1.), (959., 625., 1.), (461., 629., 1.)]

BBB
BBB
BBB

← Kék oldal(kek.jpg) képnék
az eredménye

r elejen: 3
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
cszintomb: [1499. 639. 2.]
k: 1
cszintomb: [1499. 639. 2. 996. 636. 2.]
k: 2
cszintomb: [1499. 639. 2. 996. 636. 2. 480. 636. 2.]
k: 3
```

## Nagyítva az eredmény:

```
Shell
cszintomb a vege: [4.610e+02 6.290e+02 1.000e+00 1.455e+03 6.240e+02 1.000e+00 9.590e+02
6.250e+02 1.000e+00 4.780e+02 4.800e+02 1.000e+00 9.500e+02 4.760e+02
1.000e+00 1.428e+03 4.760e+02 1.000e+00 1.405e+03 3.370e+02 1.000e+00
9.590e+02 3.380e+02 1.000e+00 4.890e+02 3.360e+02 1.000e+00 4.700e+02
6.200e+02 5.000e+00 9.710e+02 6.240e+02 5.000e+00 9.530e+02 4.750e+02
5.000e+00 4.880e+02 4.800e+02 5.000e+00 4.890e+02 3.360e+02 5.000e+00
1.443e+03 4.860e+02 5.000e+00 9.590e+02 3.360e+02 5.000e+00]
segedt [(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
(0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.) (0., 0., 0.)
segedt: [( 461., 629., 1.) (1455., 624., 1.) ( 959., 625., 1.) ( 478., 480., 1.)
( 950., 476., 1.) (1428., 476., 1.) (1405., 337., 1.) ( 959., 338., 1.)
( 489., 336., 1.)]
sorted: [(489., 336., 1.), (1405., 337., 1.), (959., 338., 1.), (950., 476., 1.), (1428., 476., 1.)

BBB
BBB
BBB

← BBB
BBB
BBB

r elejen: 3
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
cszintomb: [1499. 639. 2.]
k: 1
cszintomb: [1499. 639. 2. 996. 636. 2.]
k: 2
```

### 3. Kép tesztelése: (piros.jpg)

```
Thonny - C:\Users\Levente\Desktop\Unif\Gépieltés\beadandovero2.py @ 188:41
File Edit View Run Device Tools Help

beadandovero2.py
125 print("jelejen:", i)
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 10000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL MEGADOTNI A RECTANGELNEK TRNT A KÉP NEVÉT!!!!!!
131         cv2.rectangle(kepvallasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         #M = cv2.moments(cnt)
133         #print("Mmoments:", M)
134         #cx = (M['m10'] / M['m00'])
135         #print("cx:", cx)
136         #cy = (M['m01'] / M['m00'])
137         #print("cy:", cy)
138         #center = np.array([cx, cy])
139         #print("momentscenter:", center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #print("center/circle")

Shell
1000. 986. 0. 000. 901. 0. 1477. 343. 0. 1012. 340. 0.
561. 339. 5.]
segedt: [(1485., 671., 0.) ( 942., 669., 0.) ( 394., 668., 0.) (1468., 506., 0.)
( 949., 502., 0.) ( 425., 502., 0.) (1456., 354., 0.) ( 957., 351., 0.)
( 450., 345., 0.)]
segedt: [(1499., 639., 2.) ( 996., 636., 2.) ( 480., 636., 2.) (1486., 488., 2.)
( 993., 482., 2.) ( 510., 480., 2.) (1477., 343., 2.) (1009., 337., 2.)
( 528., 336., 2.)]
segedt: [(528., 336., 2.), (1009., 337., 2.), (1477., 343., 2.), (510., 480., 2.), (993., 482., 2.), (1486., 488., 2.), (480., 636., 2.), (996., 636., 2.), (1499., 639., 2.)]

RRR
RRR
RRR
← Piros oldal(piros.jpg)
képek az eredménye

r elejen: 4
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
jvegen: 3
i: 6
jelejen: 3
cszintomb: [1604. 704. 3.]
ki: 1
cszintomb: [1604. 704. 3. 1052. 698. 3.]
ki: 2
cszintomb: [1604. 704. 3. 1052. 698. 3. 492. 696. 3.]
```

Nagyítva az eredmény:

```
Shell
cszintomb a vege: [1499. 639. 2. 996. 636. 2. 480. 636. 2. 1486. 488.
993. 482. 2. 510. 480. 2. 1477. 343. 2. 1009. 337. 2.
528. 336. 2. 494. 656. 4. 1477. 343. 4. 1478. 343. 4.
493. 632. 5. 1509. 637. 5. 1020. 637. 5. 995. 479. 5.
1500. 486. 5. 508. 481. 5. 1477. 343. 5. 1012. 340. 5.
561. 339. 5.]
segedt: [(1485., 671., 0.) ( 942., 669., 0.) ( 394., 668., 0.) (1468., 506., 0.)
( 949., 502., 0.) ( 425., 502., 0.) (1456., 354., 0.) ( 957., 351., 0.)
( 450., 345., 0.)]
segedt: [(1499., 639., 2.) ( 996., 636., 2.) ( 480., 636., 2.) (1486., 488., 2.)
( 993., 482., 2.) ( 510., 480., 2.) (1477., 343., 2.) (1009., 337., 2.)
( 528., 336., 2.)]
sorted: [(528., 336., 2.), (1009., 337., 2.), (1477., 343., 2.), (510., 480., 2.), (993., 482., 2.), (1486., 488., 2.), (480., 636., 2.), (996., 636., 2.), (1499., 639., 2.)]

RRR
RRR
RRR
← RRR
RRR
RRR

r elejen: 4
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
jvegen: 3
i: 6
jelejen: 3
cszintomb: [1604. 704. 3. 1
```

## 4. Kép tesztelése: (sarga.jpg)

```
Thonny - C:\Users\Levente\Desktop\Unif\Gépieltás\beadando02.py @ 188 / 41
File Edit View Run Device Tools Help

beadando02.py
125 print("jelejen:")
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 10000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL MÓDOSÍTANI rectangle[DOE IRNI A KÉP NEVÉT]!!!!!!!!!!!!!!!!!!!!
131         cv2.rectangle(kepalasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         m = cv2.moments(cnt)
133         #print("Moment:", m)
134         #cx = (M["m10"] / M["m00"])
135         #print("cx:", cx)
136         #cy = (M["m01"] / M["m00"])
137         #print("cy:", cy)
138         #center = np.array([cx, cy])
139         #print("momentscenter:", center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #cv2.circle(kepalasztas(r), center, radius, (255, 0, 0), 2)

Shell
csintomb a vegen: [1604. 704. 3. 1052. 698. 3. 492. 696. 3. 1579. 537. 3.
1056. 530. 3. 536. 526. 3. 1561. 386. 3. 1065. 383. 3.
560. 375. 3. 1606. 704. 4. 1058. 698. 4. 540. 676. 4.
1581. 537. 4. 1057. 530. 4. 543. 526. 4. 1561. 386. 4.
1066. 382. 4. 566. 375. 4.]
segedt [( 461., 629., 1.) (1455., 624., 1.) ( 959., 625., 1.) ( 478., 480., 1.)
( 950., 476., 1.) (1428., 476., 1.) (1405., 337., 1.) ( 959., 338., 1.)
( 489., 336., 1.)]
segedt: [(1604., 704., 3.) (1052., 698., 3.) ( 492., 696., 3.) (1579., 537., 3.)
(1056., 530., 3.) ( 536., 526., 3.) (1561., 386., 3.) (1065., 383., 3.)
( 560., 375., 3.)]
sorted: [(560., 375., 3.), (1065., 383., 3.), (1561., 386., 3.), (536., 526., 3.), (1056.,
530., 3.), (1579., 537., 3.), (492., 696., 3.), (1052., 698., 3.), (1604., 704., 3.)]

YYY
YYY
YYY
r elejen: 5
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
jvegen: 3
i: 6
jelejen: 3
jvegen: 4
i: 8
```

## Nagyítva az eredmény:

```
Shell
csintomb a vegen: [1604. 704. 3. 1052. 698. 3. 492. 696. 3. 1579. 537. 3.
1056. 530. 3. 536. 526. 3. 1561. 386. 3. 1065. 383. 3.
560. 375. 3. 1606. 704. 4. 1058. 698. 4. 540. 676. 4.
1581. 537. 4. 1057. 530. 4. 543. 526. 4. 1561. 386. 4.
1066. 382. 4. 566. 375. 4.]
segedt [( 461., 629., 1.) (1455., 624., 1.) ( 959., 625., 1.) ( 478., 480., 1.)
( 950., 476., 1.) (1428., 476., 1.) (1405., 337., 1.) ( 959., 338., 1.)
( 489., 336., 1.)]
segedt: [(1604., 704., 3.) (1052., 698., 3.) ( 492., 696., 3.) (1579., 537., 3.)
(1056., 530., 3.) ( 536., 526., 3.) (1561., 386., 3.) (1065., 383., 3.)
( 560., 375., 3.)]
sorted: [(560., 375., 3.), (1065., 383., 3.), (1561., 386., 3.), (536., 526., 3.), (1056.,
530., 3.), (1579., 537., 3.), (492., 696., 3.), (1052., 698., 3.), (1604., 704., 3.)]

YYY
YYY
YYY
r elejen: 5
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
jvegen: 3
i: 6
jelejen: 3
jvegen: 4
i: 8
```

## 5. Kép tesztelése: (narancs.jpg)

```
Thonny - C:\Users\Levente\Desktop\Unif\Gépi\test\beadando2.py @ 188:41
File Edit View Run Device Tools Help

beadando2.py
125 print("jelejen:", i)
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 10000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL MEGADOTNI A KÉP NEVÉT!!!!!!!!!!!!!!!!!!!!!!
131         cv2.rectangle(kepvalasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         #M = cv2.moments(cnt)
133         #print("Moments:", M)
134         #cx = (M["m10"] / M["m00"])
135         #print("cx:", cx)
136         #cy = (M["m01"] / M["m00"])
137         #print("cy:", cy)
138         #center = np.array([cx, cy])
139         #print("momentscenter:", center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #M = cv2.moments(cnt)

Shell
978. 477. 4. 484. 472. 4. 1460. 337. 4. 987. 332. 4.
512. 330. 4.]
ki: 9
jvegen: 5
i: 10
jelejen: 5
jvegen: 6
cszintomb a vegen: [1493. 632. 4. 444. 630. 4. 967. 627. 4. 1469. 479. 4.
978. 477. 4. 484. 472. 4. 1460. 337. 4. 987. 332. 4.
512. 330. 4.]
segedt [(1499., 639., 2.) ( 996., 636., 2.) ( 480., 636., 2.) (1486., 488., 2.)
( 993., 482., 2.) ( 510., 480., 2.) (1477., 343., 2.) (1009., 337., 2.)
( 528., 336., 2.)]
segedt: [(1493., 632., 4.) ( 444., 630., 4.) ( 967., 627., 4.) (1469., 479., 4.)
( 978., 477., 4.) ( 484., 472., 4.) (1460., 337., 4.) ( 987., 332., 4.)
( 512., 330., 4.)]
sorted: [(512., 330., 4.), (987., 332., 4.), (1460., 337., 4.), (484., 472., 4.), (978., 477., 4.), (1469., 479., 4.), (967., 627., 4.), (444., 630., 4.), (1493., 632., 4.)]

ooo
ooo
ooo
ooo
r elejen: 6
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 5
```

## Nagyítva az eredmény:

```
Shell
i: 10
jelejen: 5
jvegen: 6
cszintomb a vegen: [1493. 632. 4. 444. 630. 4. 967. 627. 4. 1469. 479. 4.
978. 477. 4. 484. 472. 4. 1460. 337. 4. 987. 332. 4.
512. 330. 4.]
segedt [(1499., 639., 2.) ( 996., 636., 2.) ( 480., 636., 2.) (1486., 488., 2.)
( 993., 482., 2.) ( 510., 480., 2.) (1477., 343., 2.) (1009., 337., 2.)
( 528., 336., 2.)]
segedt: [(1493., 632., 4.) ( 444., 630., 4.) ( 967., 627., 4.) (1469., 479., 4.)
( 978., 477., 4.) ( 484., 472., 4.) (1460., 337., 4.) ( 987., 332., 4.)
( 512., 330., 4.)]
sorted: [(512., 330., 4.), (987., 332., 4.), (1460., 337., 4.), (484., 472., 4.), (978., 477.,

ooo
ooo
ooo
ooo
r elejen: 6
i: 0
jelejen: 0
jvegen: 1
i: 2
jelejen: 1
jvegen: 2
i: 4
jelejen: 2
jvegen: 3
i: 6
jelejen: 3
jvegen: 4
```

## 6. Kép tesztelése: (feher.jpg)

```
Thorny - C:\Users\Levente\Desktop\Unigépi\fej\beadandovero02.py @ 188:41
File Edit View Run Device Tools Help

beadandovero02.py
125 print("jelejen:",j)
126 contours, _ = cv2.findContours(mask, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE)
127 for cnt in contours:
128     if cv2.contourArea(cnt) > 10000:
129         x, y, w, h = cv2.boundingRect(cnt)
130         # ITT KELL MÓDOSÍTANI rectangle(IDE ÉRTI A KÉP NEVÉT)!!!!!!!!!!!!!!!!!!!!
131         cv2.rectangle(kepalasztas(r), (x, y), (x + w, y + h), (0, 255, 0), 2)
132         #M = cv2.moments(cnt)
133         #print("Mmoment:",M)
134         #cx = (M["m10"] / M["m00"])
135         #print("cx:",cx)
136         #cy = (M["m01"] / M["m00"])
137         #print("cy:",cy)
138         #center = np.array([cx,cy])
139         #print("momentscenter:",center)
140         circles = [cv2.minEnclosingCircle(cnt)]
141         #M = cv2.moments(cnt)

Shell
k: 8
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5.]
k: 9
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5.]
k: 10
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5. 513. 349. 5.]
k: 11
jvegen: 6
cszintomb a vegen: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5. 513. 349. 5.]
segedt [(1604., 704., 3.) (1052., 698., 3.) ( 492., 696., 3.) (1579., 537., 3.)
(1056., 530., 3.) ( 536., 526., 3.) (1561., 386., 3.) (1065., 383., 3.)
( 560., 375., 3.)]
segedt [(1507., 663., 5.) ( 980., 659., 5.) ( 453., 655., 5.) (1023., 506., 5.)
(1006., 502., 5.) (1496., 503., 5.) (1003., 504., 5.) ( 491., 497., 5.)
(1479., 356., 5.)]
sorted: [(1479., 356., 5.), (491., 497., 5.), (1006., 502., 5.), (1496., 503., 5.), (1003., 504., 5.), (1023., 506., 5.), (453., 655., 5.), (980., 659., 5.), (1507., 663., 5.)]

WWW
WWW
WWW
Fehér oldal(fehér.jpg)
képnek az eredménye
>>>
```

## Nagyítva az eredmény:

```
Shell
k: 8
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5.]
k: 9
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5.]
k: 10
cszintomb: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5. 513. 349. 5.]
k: 11
jvegen: 6
cszintomb a vegen: [1507. 663. 5. 980. 659. 5. 453. 655. 5. 1023. 506. 5.
1006. 502. 5. 1496. 503. 5. 1003. 504. 5. 491. 497. 5.
1479. 356. 5. 997. 353. 5. 513. 349. 5.]
segedt [(1604., 704., 3.) (1052., 698., 3.) ( 492., 696., 3.) (1579., 537., 3.)
(1056., 530., 3.) ( 536., 526., 3.) (1561., 386., 3.) (1065., 383., 3.)
( 560., 375., 3.)]
segedt [(1507., 663., 5.) ( 980., 659., 5.) ( 453., 655., 5.) (1023., 506., 5.)
(1006., 502., 5.) (1496., 503., 5.) (1003., 504., 5.) ( 491., 497., 5.)
(1479., 356., 5.)]
sorted: [(1479., 356., 5.), (491., 497., 5.), (1006., 502., 5.), (1496., 503., 5.), (1003., 504., 5.), (1023., 506., 5.), (453., 655., 5.), (980., 659., 5.), (1507., 663., 5.)]

WWW
WWW
WWW
WWW
WWW
WWW
>>>
```

# Tesztelés zavaró tényezős képpel

Ezen az oldalon fogom bemutatni azokat a teszteket, amelyek tartalmaznak kirakott, és összekevert kockát egyaránt, de olyan környezeti háttérrel, és olyan zavaró tényezőkkel, vagy esetleg helytelen fényviszonyok között, hogy azt a program már nem tudja rendesen feldolgozni.

## **Jelentése az eredményen látható betűknek:**

G->Green, ami magyarul a zöld színt jelenti

B->Blue, ami magyarul a kék színt jelenti

R->Red, ami magyarul a piros színt jelenti

Y->Yellow, ami magyarul a sárga színt jelenti

O->Orange, ami magyarul a narancs színt jelenti

W->White, ami magyarul a fehér színt jelenti



1. **Tesztelés(Test (1).jpg):** Ez a kép egy kirakott kockát tartalmaz, de az benne a zavaró tényező, hogy több oldalát is látjuk, illetve a fényviszonyok sem megfelelőek, hiszen árnyék tűnik fel mögötte:



## Eredménye a tesztnek:

```

1118. 1042. 5. 1084. 576. 5. 1069. 518. 5. 738. 517. 5.
2305. 453. 5. 453. 449. 5. 1091. 393. 5.]
seged: [1.7875762e+39, 6.4449359e+39, 5.3245527e+39]
(7.8681461e+39, 8.2755463e+39, 1.0581240e+39)
(6.7796226e+39, 1.0838773e+39, 1.0120046e+39)
(9.2755477e+39, 6.2449427e+39, 1.0561224e+39)
(1.0635067e+39, 1.0285754e+39, 7.8081461e+39)
(9.4408813e+39, 6.5204481e+39, 1.0285754e+39)
(9.5103007e+39, 1.0562233e+39, 1.0561397e+39)
(7.7431016e+39, 1.0561224e+39, 2.9389996e+39)
(4.4899956e+39, 6.2264903e+39, 1.0285754e+39)]
segeds: [[541., 535., 0.] (743., 515., 0.) (453., 449., 0.) (312., 451., 4.)
(242., 386., 4.) (603., 384., 4.) (309., 328., 4.) (926., 327., 4.)
(355., 328., 4.)
(11555., 326., 4.), (526., 327., 4.), (309., 328., 4.), (603., 384., 4.), (242., 386., 4.), (453., 449., 0.), (312., 451., 4.), (743., 518., 0.), (541., 535., 0.)]]

```

**Test (1).jpg eredménye**

```

000
001
002
003

```

```

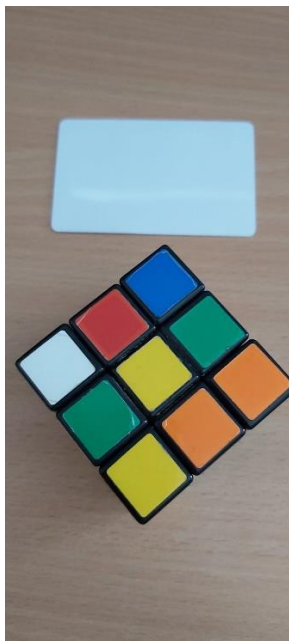
i = i+1; m=0
i=0
j=alejoni 0
j=alejoni 1
i=2
j=alejoni 1
mainintomb: [541. 526. 3.]
k=1
centintomb: [4.610e+02 6.250e+02 1.000e+00 1.455e+03 6.240e+02 1.000e+00]
R1 2
mainintomb: [4. 610e+02 6. 250e+02 1. 000e+00 1. 455e+03 6. 240e+02 1. 000e+00 5. 743e+02

```

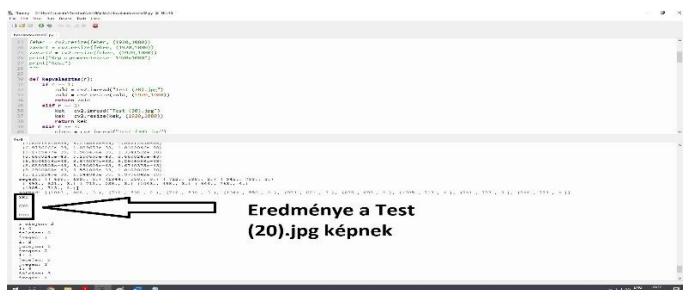
## Nagyítva:

[illegible]

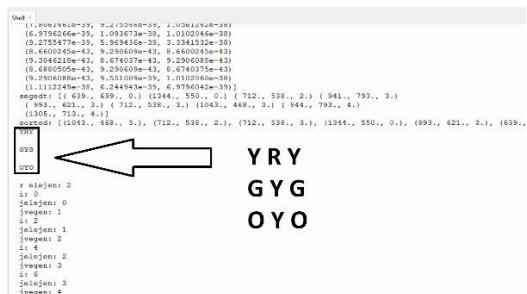
**2. Tesztelés (Test (15).jpg): Ezen a képen egy kevert kockát láthatunk elforgatva, illetve egy zavaró tényezőt. A hátrány, amiért a program hibát fog kidobni az az, hogy zavaró tényezőt raktam a képre, és a fényviszonyok is rosszak (árnyékok a képen).**



**Eredménye a tesztnek:**



**Nagyítva:**





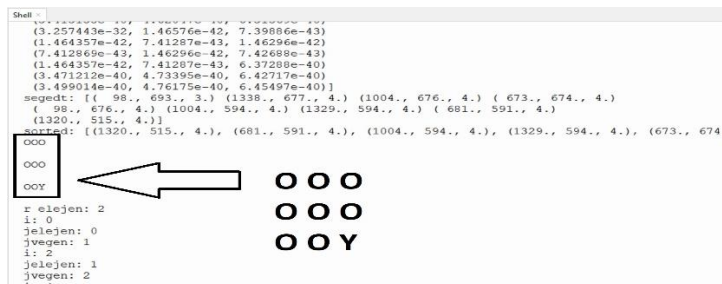
**3. Tesztelés (zavarotenyezok01.jpg): Ezen a képen egy kirakott kocka piros oldalát láthatjuk. Több zavaró tényező tárgy is látható a képen. Fényviszony is más mint az előző képeken.**



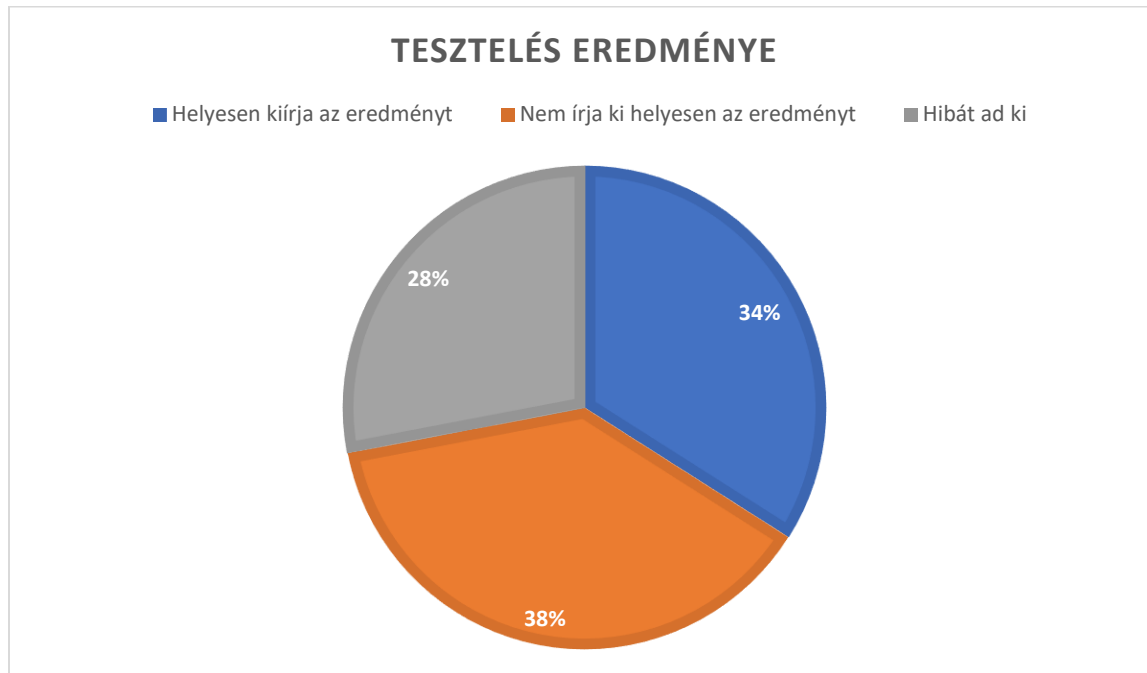
## Eredménye a tesztnek:



## Nagyítva:



**Teszteltem mind a 32 teszt képet, és százalékosan ez az eredmény született:**



**Ezek között a teszt képek között található:**

- -Kirakott állapotú kocka
- -Kevert kocka
- -Homogén háttérrel rendelkező kocka
- -Zavaró tényezővel rendelkező háttér
- -Zavaró tárgyak, zavaró színekkel
- -Fényviszonyok különböző formája
- -Elforgatott kocka
- -Mind a három oldalt látható kocka pozíció

## **Felhasználói leírás**

Először is 6 darab képre van szükségünk a program helyes futásához(Ezeket úgy kell elkészíteni, hogy a kocka mind a 6 oldalát lefotózzuk). Ezt a hat képet ugyan abba a mappába kell tenni, ahol a program kódja található. Olyan néven nevezzük el, ahogy a képen látható kocka oldalának a közepén lévő szín van. Tehát például ha lefotóztuk 6 külön oldalát a kockának, és az adott képen a fehér van középen, akkor azt érdemes fehér.jpg névre átnevezni, és így tovább. Ezt a 6 képet ha bemásoltuk, akkor további teendők nincs. A programnak működnie kell.

**Felhasznált program:**

Ide inkább egy kódot szeretnék megadni, amit feltöltöttem a githubra, ami a hsv detektálásában segített, hogy a színeket jól tudja maszkolni a program, és ennek a fájlnek a neve:hsv\_detection.py

Természetesen ez nem az én munkám, de nem találtam meg sajnos a linket, amelyről letöltöttem.