Dokumentáció a Rubik-kocka állapotának felismeréséhez fényképek alapján

# Bevezetés

A projekt célja a Rubik-kocka állapotának felismerése és rögzítése fényképek alapján. A script képes feldolgozni a képeket, azonosítani a színeket, majd az eredményeket egy Excel táblázatban megjeleníteni. A dokumentáció célja bemutatni a projekt megvalósításának folyamatát és az alkalmazott módszereket.

# Eszközök és technológiák ismertetése

A projekt megvalósításához az alábbi eszközöket és technológiákat használtam:

Python: A projekt fő programozási nyelve.

A Python egy magas szintű, általános célú programozási nyelv, amelyet Guido van Rossum alkotott meg, és először 1991-ben jelent meg. A Python tervezésénél nagy hangsúlyt fektettek a kód olvashatóságára és egyszerű szintaxisára, amely lehetővé teszi a programozók számára, hogy kevesebb kódsorral fejezzék ki az ötleteiket. Ennek köszönhetően a Python egy rendkívül hatékony és könnyen tanulható nyelv, amely széles körben elterjedt mind az iparban, mind az oktatásban.

OpenCV: A képfeldolgozáshoz és színazonosításhoz használt könyvtár.

Az OpenCV (Open Source Computer Vision Library) egy nyílt forráskódú számítógépes látás könyvtár, amelyet az Intel hozott létre 1999-ben, és ma széles körben használják különféle képfeldolgozási és számítógépes látási feladatok megoldására. Az OpenCV C++, Python és Java nyelveken is elérhető, de a Python interfész az egyik legnépszerűbb a könnyű használhatóságának és rugalmasságának köszönhetően.

NumPy: A numerikus számításokhoz és tömbkezeléshez használt python könyvtár.

openpyxl: Az eredmények Excel táblázatba történő írásához használt python könyvtár.

# Műszaki dokumentáció

## Könyvtárak és függvények

* OpenCV (cv2):
  + Képfeldolgozás, gamma korrekció, kontúrok keresése, színek megállapítása/összehasonlítása.
* NumPy:
  + Tömbműveletek, színek átlagolása.
* openpyxl:
  + Excel fájl létrehozása, Rubik-kocka ábrázolása, megállapított értékek megjelenítése

## Főbb függvények

dominant\_color(cell): Megkeresi a domináns színt egy képen.

adjust\_gamma(image, gamma): Gamma korrekciót végez a képen.

find\_closest\_color(color, colors): Megkeresi a legközelebbi színt az előre definiált színek közül.

rgb\_to\_hex(rgb): RGB színkódot hexadecimális formátumba konvertál.

# A script lépései

## Gamma érték beállítása:

A képfeldolgozás során nagyon fontos a megfelelő színek használata, így a scriptet kiegészítettem egy külső ciklussal. Ez a ciklus egy nagyon magas (2,5) gamma értéket állít be kezdőértékként, és a Rubik-kocka 6 oldaláról készült képeket ezzel a gamma korrekciós értékkel próbálja meg feldolgozni. A cél az, hogy minden színből 9 darabot találjon. Ha nem jár sikerrel, elkezdi a gamma értéket csökkenteni, míg nem lesz sikeres a színek felismerése, vagy el nem ér egy túl alacsony gamma értéket. Túl alacsony érték esetén egy „flag” segítségével az alapértelmezett 1.0 értékkel fut le utoljára a színfelismerés, a kimeneti fájl az így megállapított színek alapján készül el.

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Automatikusan generált leírás

1. ábra: Gamma korrekciós ciklus

## Kép beolvasása és előfeldolgozása

A beolvasott képet a script először olyan képpé alakítja, ami csak teljesen fekete vagy fehér képpontokból áll. Ez a lépés azért szükséges, mert így sokkal könnyebb megtalálni a fényképen a Rubik-kocka kontúrjait.

A képen tér, tervezés, rács, fekete-fehér látható

Automatikusan generált leírás

2. ábra: fekete-fehér Rubik-kocka.

A megtalált kontúrok segítségével egy bounding box megrajzolása következik, ami pontosan meghatározza a képen a Rubik-kocka helyzetét, majd ezeknek az értékeknek a felhasználásával az eredeti képből kivágja a hasznos területet, így a kép többi részével nem kell a későbbiekben foglalkozni.

A képen Mechanikus kirakójáték, játék, Rubik-kocka, puzzle látható

Automatikusan generált leírás

3. ábra: Kontúrok és bounding Box.

Miután elkészült a kivágott kép, a script alkalmazza erre a képre a gamma korrekciós függvényt.

A gammakorrekció után a képet 2 vízszintes és 2 függőleges vonal segítségével a script 9 egyenlő részre osztja, majd ezeket a kép részleteket egyesével, külön képekként dolgozza fel.

A képen Színesség, színek, paletta, művészet látható

Automatikusan generált leírás

A képen Színesség, Téglalap, színek, művészet látható

Automatikusan generált leírás

A Rubik-kocka fehér oldalán középen található a termék logója. Ez a színekkel teli logó vezetett ahhoz a megoldáshoz, hogy mind a 9 képből egy kisebb részletet vág ki a script a bal felső részből a cella színének megállapításához. (És ahhoz is, hogy kötött a kockáról készült képek elkészítésének a módja)

………………………..

…………………………

……………………….

Átlagos szín kiszámítása minden cellában.

Legközelebbi szín megtalálása az előre definiált színek közül.

Eredmények tárolása:

Színek tárolása egy numpy tömbben.

Felismert színek tárolása egy külön numpy tömbben.

Excel fájl létrehozása és kitöltése:

Excel munkafüzet létrehozása.

Sorok és oszlopok méretének beállítása.

Cellák kitöltése az átlagos szín értékekkel.

Cellák kitöltése a felismert színek neveivel.

Eredmények kiírása az Excel fájlba.

script leírása után a script használatának bemutatása, majd az eredmények kiértékelése