Pénzérme számláló alkalmazás

Fejlesztői és felhasználói dokumentáció

Bognár Milán Károly

Neptun kód: S3PH8C

Gépi látás (GKNB_INTM038)

2020/2021. tanév, őszi félév

Tartalomjegyzék

1. Fejlesztői dokumentáció

1.1. Feladat bemutatása, elvárt funkciók

Féléves feladatomban egy olyan alkalmazás elkészítése volt a cél, amely képes felismerni pénzérméket az általunk készített képekről. A programnak fel kell ismernie a pénzérmék típusait és meg kell határoznia, hogy a képen látható érméknek mennyi az összértékük. A pénzérmék jellemzőit előre definiáljuk, csak az ilyen tulajdonságokkal rendelkező érméket kell felismernie az alkalmazásnak. A program a magyar forint típusú érméket képes felismerni (5 Ft, 10 Ft, 20 Ft, 50 Ft, 100 Ft, 200 Ft).

A programot Python nyelven készítettem el, kihasználva ennek a programozási nyelvnek az előnyeit, mint például a programozási nyelvhez készített csomagok, amelyek megkönnyítik a képfeldolgozást és a kép elemeinek elemzését.

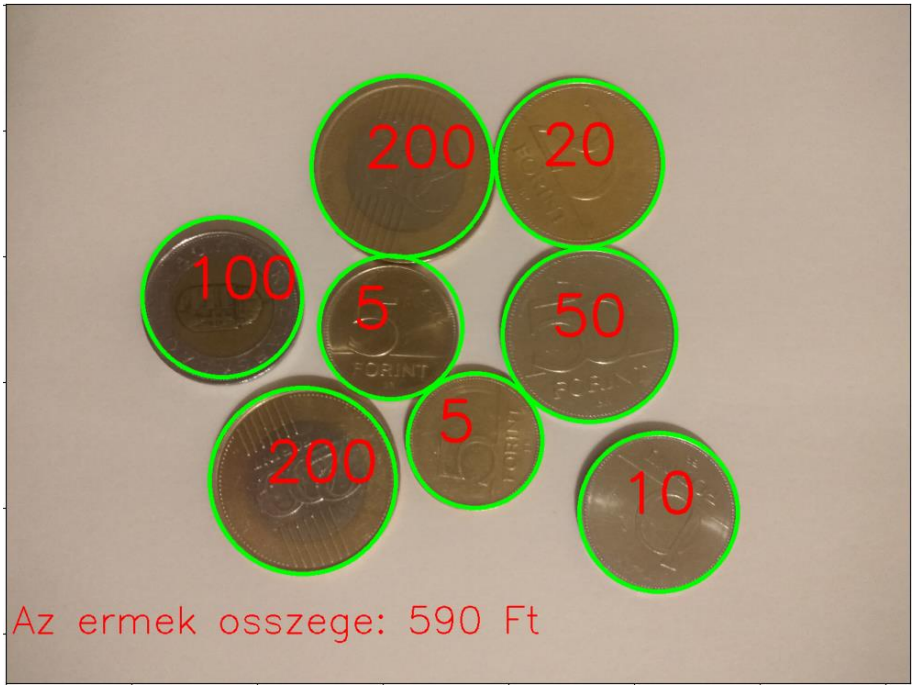
Bemenet:

Általunk készített kép, amely magyar forint típusú pénzérméket tartalmaz, illetve fehér háttér előtt készült. A bemeneti kép lehetőleg minél kevesebb árnyékot tartalmazzon, mert az ronthat a program sikerességén.

1.ábra: Bemeneti kép illusztrációja

Kimenet:

A kimeneti képen jelenjen meg a felismert pénzérmék értéke külön-külön, illetve legyen kiírva a képre a megtalált pénzérmék összértéke forintban. A kimeneti képeken zöld körökkel láthatjuk azt is, hogy az adott pénzérméket milyen pozícióban találta meg a program.



2.ábra: Kimeneti kép illusztrációja

1.2. Fejlesztői környezet, Python könyvtárak

A programot Visual Studio 2019 fejlesztői környezetben, Python programozási nyelven készítettem el.

A Python nyelvhez találhatunk csomagokat, amelyek megkönnyítik a képfeldolgozás lépéseit számunkra. Ezek közül az alábbiakat használtam:

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
```

3.ábra: A programban felhasznált Python csomagok listája

1.3. Feladat elméleti háttere

A pénzérme felismerés feladat első részében az érmék pozícióját kell meghatároznunk. A pénzérmék kör alakú objektumok, tehát a képen meg kell keresnünk a nekünk megfelelő méretű kör alakzatokat. Ezt megtehetjük többféleképpen is. Amennyiben szeretnénk minden egyes lépést magunk megcsinálni, és előre megírt, beépített függvény nélkül megoldani a feladatot, akkor a lépések a következők lennének:

- Szürkeárnyalatossá alakítás
- Gauss szűrő alkalmazása a bemeneti képen
- Éldetektálás: Canny éldetektor segítségével
- Megadott sugarú körök készítése, amelyekkel végig iterálunk a kapott képen, és megkeressük, hogy vannak-e ilyen sugarú körök
- Amennyiben találunk egyezést, az adott kör középpontjának koordinátáit eltároljuk, illetve a sugár méretét is elmentjük

Ez a módszer nem biztos, hogy a leghatékonyabb, illetve elég sok helyen hiba kerülhet a kódba. Az előbb felvázolt módszer mintájára találhatunk az OpenCV könyvtárban egy függvényt, amely megkeresi nekünk a köröket a képen. A függvény neve: cv2.HoughCircles

1.3.1. Körök detektálása Hough transzformációval:

A körök detektáláshoz szükségünk van egy transzformációra a képtérből a Hough térbe. Ez a transzformáció megadja az egy ponton átmenő összes egyenes egyenletét. A képtérben egy pontot x és y koordinátákkal jelölhetünk. Ehhez a képponthoz tartozó görbe a Hough térben a következő: $r = x * \cos(\varphi) + y * \sin(\varphi)$

A kör detektálása kétféleképpen történhet:

- Ismert sugarú köröket keresünk (például ipari környezetben használnak ilyeneket)
- Ismeretlen sugarú köröket keresünk (Egyéb, például mobilos felismerő applikációk esetén)

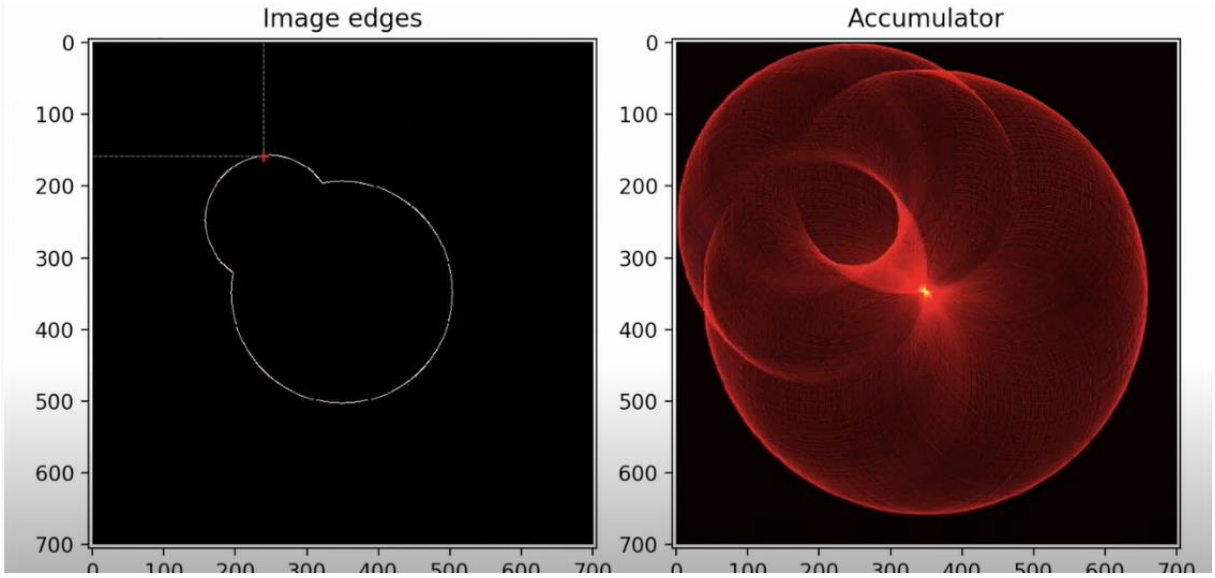
Általános körök esetén a Hough tér 3 dimenziós lesz (amikor nem ismert a keresett kör sugara). Amennyiben ismert a sugár, akkor 2 dimenziós térről beszélünk.

Kör egyenlete a képtérben: $ax^2 + by^2 = r^2$ ahol (a,b) a kör középpontja, r a sugár

Abban az esetben, ha a sugár rögzített, akkor a Hough tér 2 dimenziósra csökken. Az eredeti kör minden egyes pontjára (x, y) meghatározhat egy (x, y) középpontú kört r méretű sugárral. Az összes ilyen kör metszéspontja a paramétertérben megegyezik az eredeti kör középpontjával. A 4. ábrán ezt a keresési folyamatot láthatjuk. A bal oldali képen látjuk az eredeti képet éldetektálás után. A forrásként megjelölt videóban nagyon jól szemléltetik, hogy a megtalált él mindegy egyes képpontján végigmegy az algoritmus. Ha a Hough térben minden ugyanazon pontokból(x,y) készítünk r sugárral köröket, akkor egy olyan ábrát kapunk, ami a jobboldali képen látható.

Az eredeti kör középpontját úgy kapjuk meg, hogy egy összegzőtömbben nyilvántartja az algoritmus, hogy mely ponton megy át a legtöbb körív, és ahol ennek a tömbnek a maximuma van, ott lesz a kör középpontja.

Eredeti kép éldetektálás után: Középpont keresése a Hough térben:



4.ábra: kör detektálása a Hough térben, abban az esetben, ha ismert a keresett kör sugara

Megjegyzés: A jobb oldali ábrán láthatjuk, hogy a Hough térben a keresett középpontot ott találhatjuk meg, ahol a legtöbb kör metszi egymást. Az ábrán ennek a helye jól kivehető.

forrás: https://www.youtube.com/watch?v=Ltqt24SQQoI

A cv2.HoughCircles függvény használatakor nagyon fontos, hogy a minRadius, maxRadius paramétereket megfelelően állítsuk be, mert ha rosszul vannak ezek megadva, akkor számunkra felesleges -túl kicsi vagy túl nagy- köröket is találni fog a függvény.

Ha a keresett körök sugara nem ismert, akkor 3D-s Hough térben kell dolgoznunk. Ebben az esetben a módszer ugyanaz, de a sugár mindig növekedni fog.

1.4. Feladat megoldása, program felépítése

A feladat első felében a képen meg kell keresnünk a kör alakzatokat, amelyekről később eldöntjük, hogy milyen pénzérme található az adott pozícióban.

Ehhez a következő lépéseket használtam:

- **Kép átméretezése**

A legelső művelet a beolvasott képpel az volt, hogy átméreteztem. Erre azért van szükség, mert a mai technológiával már nagy felbontású képeket vagyunk képesek készíteni, de ennél a feladatnál nem kell, hogy a képek felbontása túl nagy legyen. Ha túl nagy a kép felbontása, az még akár hátráltathatja is a munkánkat, mivel sokkal részletesebb és ezáltal a feldolgozása lassabb lehet. A képeket 1440 képpont szélesre méreteztem át.

- **Kép szürkeárnyalatossá alakítása**

Az OpenCV a színcsatornákat BGR sorrendben tárolja, ezt először átalakítom RGB sorrendre. Ezután, elkészítem a kép szürkeárnyalatos másolatát. Erre azért van szükség, mert a cv2.HoughCircles függvény egy szürkeárnyalatos képet vár paraméterként.

5.ábra: szürkeárnyalatossá alakított kép

- **Gauss szűrő alkalmazása**

A harmadik műveletben az átméretezett és szürkeárnyalatos képen Gauss-szűrőt alkalmazok. A körök megkeresésénél nem fontos a részletesség, az éleket szeretnénk detektálni.

6.ábra: Gauss-szűrő alkalmazása

- **Kör alakzatok keresése a képen, HoughCircles függvény segítségével**

A kör alakzatokat a feljebb már ismertetett Hough transzformáció segítségével fogjuk megkeresni. Erre az OpenCV-ben a cv.HoughCircles függvény van a segítségünkre. A függvény visszaad egy eredménytömböt, amelyben eltárolta a keresett körök középpontjainak x és y koordinátáit, és a sugár hosszát. ezt láthatjuk a konzolon kiírva.

```
    X    Y    R
[[[ 472  758  147]
  [ 628  256  143]
  [ 342  474  131]
  [ 912  254  133]
  [1038  808  126]
  [ 930  516  147]
  [ 744  696  109]
  [ 608  512  109]]]
Megtalaltam 8 db penzermet!
```

7. ábra: A cv2.HoughCircles segítségével eltároltuk a körök pozícióját és sugarát

Miután megtaláltuk a pénzérméket (köröket) a képen, és ezeknek a pozícióját eltároltuk, ki kell elemeznünk, hogy a kép adott szeletében milyen típusú érme található.

- **Pénzérme típusának meghatározása**

Körök területének átlagos színintenzitása

A pénzérmék egyik meghatározó tulajdonsága, az összetételük, amelyek különböző színeket eredményeznek. Például a 10 forintos és az 50 forintos érme is 75% rézből és 25% Nikkelből épül fel. Ennek eredményeképpen mind a két érme ezüstös/ szürkés színű. A képeknek van egy olyan tulajdonságuk, hogy leírhatók mátrixok segítségével és minden egyes képpontot egy számhármassal tudunk jellemezni, amelyek megadják a színcsatornák értékeit. Ezt kihasználva a pénzérmék területét kivágtam a képből, majd ennek a területnek vettem az összes képpontját és ezek színintenzitását átlagoltam. Így megkaptam minden érmére egy értéket, amely jellemzi az adott pénzérmét. Ezzel a módszerrel jól el lehet különíteni a 10 Ft-os és az 50 Ft-os érméket a többitől, mert azok színei sötétebbek ezekhez viszonyítva.

8. ábra: 10 forintos érme területe a képen

A pénzérmék területének átlagos színintenzitását az alábbi függvénnyel számoltam ki:

A függvényben minden egyes oszlopban megnézzük az átlag intenzitást, ezeket átlagoljuk az np.mean() függvény segítségével.

```
def averageIntensity(img, circles):
    av_values = []
    for coordinates in circles[0,:]:
        r=coordinates[2]
        column = np.mean(img[coordinates[1]-r:coordinates[1]+r, coordinates[0]-r:coordinates[0]+r])
        av_values.append(np.around(column))
    print (av_values)
    return av_values
```

9. ábra: átlagos színintenzitás kiszámolása

Kör sugara

A pénzérmék meghatározásához újra elő kell vennünk a megtalált körök sugarát, hiszen a másik fontos tényező, amely meghatároz egy pénzérmét, az a mérete. Az 5 és 200 forintos érmék mérete kiugró a többi közül, hiszen az első a legkisebb érme, a második pedig a legnagyobb érme mind közül.

A többi érme esetében a méretet és az átlagos színintenzitást is figyelembe kell venni, csak így tudjuk meghatározni az érmék értékét.

Kiíratás

Miután meghatároztuk a pénzérmék értékét, kiírjuk az adott érme fölé az értéket, és a kép aljára az összértéket. A kimeneti képet feljebb már bemutattam.

1.5. Tesztelés

1.6. Fejlesztési lehetőségek

2. Felhasználói dokumentáció