



Gépi Látás

TKNB\_INTM038

Rendszámtábla azonosítás

Paluska Máté

HOOXOH

Tata, 2022

**Tartalomjegyzék**

[1 Bevezetés 2](#_Toc507262155)

[2 Feladat elméleti háttere 3](#_Toc507262156)

[2.1 Kép szürkeárnyalatossá alakítása 3](#_Toc507262157)

[2.1.1 Címsor 3 3](#_Toc507262158)

[3 Címsor 1 4](#_Toc507262159)

[4 Címsor 1 5](#_Toc507262160)

[5 Felhasznált irodalom 6](#_Toc507262161)

[6 Mellékletek 7](#_Toc507262162)

# Bevezetés

Napjainkban rohamosan növekedik a forgalomban levő gépjárművek száma, aminek következtében egyre fontosabb szerepet kapnak a különböző gépjármű azonosító informatikai megoldások is. Az ilyen megoldások szükségességét indokolja az is, hogy a modern világban a monoton, gépies munkákat, feladatokat hatékony módon próbálunk automatizálni. Erre kiváló a példa a rendszámok azonosításának esete például egy parkolóházba való belépéskor, autópályákon matricavásárlás ellenőrzésére, lopott vagy körözés alatt álló autók szűrésére, figyelésére, stb. Ilyen esetekben könnyen belátható, hogy nagy szükség van ezekre a rendszerekre, mivel az emberi munkaerő számára ez pusztán csak olvasásból áll és rengeteg energia megspórolható vele. Sokkal könnyebbé és gyorsabbá válik az azonosítás ezen módszerekkel. Mindezek mellett a hibázás lehetőségének csökkentése sem utolsó szempont, mivel emberi munkavégzés esetén sokkal nehezebb a hibák valószínűségét alacsonyan tartani, míg szoftveres megoldással viszonylag magas pontosság is elérhető. A dolgozatomban egy ilyen rendszámtábla azonosító megoldást fogok bemutatni, amely képes egy bemeneti kép alapján a rendszámtábla felismerésére és azonosítására. Fontos tisztázni, hogy a rendszámtábla felismerés nem egyenlő az azonosítással, mivel a felismerés esetében csak a táblát és annak pozícióját határozzuk meg a képen, a tábla szöveges tartalmát már nem olvassuk le róla, míg az azonosítás esetében ez is megtörténik. Ezek mellett nem csak a magyar rendszámtípusok leolvasása a cél, hanem bármilyen olyan rendszám, amely a latin ABC betűit és/vagy számok kombinációit tartalmazza.

# Feladat elméleti háttere

## Kép szürkeárnyalatossá alakítása

Első lépésként a kép BGR színkódjából egy szürkeárnyalatos képet állítunk elő.

## Bilaterális szűrő használata

A szürkeárnyalatos képen ezután egy bilaterális (kétoldalú) szűrést végzünk. Ez a szűrés egy alapvető művelet a képfeldolgozásban, zajcsökkentés mellett az éleket jobban kivehetően tartja.

A szűrés jelentése ebben a kontextusban, hogy a kép értéke egy megadott helyen a környező képpontok súlyozása alapján lesz kiszámítva. A súlyozás a Gauss féle eloszlást követi, ami azt jelenti, hogy a képponthoz közelebb álló pixelek nagyobb súlyozással számítanak bele a súlyozott átlag kiszámításába. Ez azért van, mivel a képek jellemzően lassan változnak a térben, ezért a középponthoz közelebb álló pixelek nagyobb eséllyel fognak hasonló értékekkel rendelkezni.[1]

A bilaterális szűrés alapötlete az, hogy a kép tartományában ugyanaz hajtódik végre, ami a hagyományos szűrők esetén a saját területükön. Két pixel lehet közel egymáshoz (térben) vagy lehet egymáshoz hasonló (hasonló értékeik vannak).[1]

A térbeli eltérések feltérképezése meghiúsul az éleknél, aminek következtében az élek elmosódnak a képen. Ennek a problémának a megoldására lehet alkalmas az anizotrop diffúzió, ahol parciális differenciálegyenletek megoldásával átlagolnak a problémás területekre.[1]

## Éldetektálás

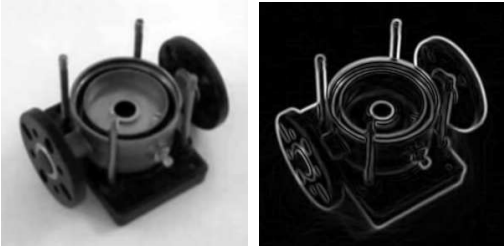
Az éldetektálást célja, hogy jelentősen csökkentsük az adatmennyiséget a képen, miközben a strukturális jellemzőket megőrizzük. Sokféle megoldás létezik éldetektálásra, én a projektemben a Canny algoritmust használtam, melyet John F. Canny dolgozott ki a 80-as években.[2]

A Canny algoritmus 5 fő lépésből áll:

* **Simítás**: *A kép elhomályosítása zajcsökkentés céljából.*
* **Színátmenetek keresése**: *Az élek megjelölése, ahol a kép színátmenetei nagyok.*
* **Non-maximum suppression (nem-maximális elnyomás)***: Csak a lokális maximumokat kell élként megjelölni.*
* **Kettős küszöb***: A lehetséges éleket egy küszöbérték határozza meg.*
* **Élkövetés hisztérzissel***: A legvégső élek meghatározás olyan módon, hogy eltüntetjük az összes olyan élt, amely nem kapcsolódik egy erősebb élhez.*

Minden kamerával készített képen található valamennyi zaj. Ezért **simításkor** egy Gauss szűrő alkalmazásával kicsit homályosabbá tesszük a képet.[2]

Az algoritmus ott talál éleket, ahol a szürkeárnyalat intenzitása leginkább változik. Ezeket a területeket a **színátmenetek keresésével** lehet meghatározni. Ezeket az átmeneteket a Sobel-operátor segítségével lehet meghatározni. Első lépésként a színátmenet x és y irányú közelítését kell végrehajtani egy kernel segítségével. Az átmenet nagyságait ezután euklideszi távolságmértékként tudjuk meg határozni a Pitagoras tétel alkalmazásával. A színátmenetek keresése utáni állapot az **1. ábrán** látható.[2]



1. **ábra**: simítás után, majd a színátmenetek keresése után.

*Forrás:[2] 3. o. 2. ábra*

A **non-maximum suppression** lényege, hogy a képen látható vastag éleket vékonyabbá alakítsuk. Ez úgy történik, hogy az algoritmus az összes helyi maximumot megőrzi a képen a többit pedig törli. Ezáltal vékony, élesebb éleket kapunk.[2]

Az élek vékonyítása után még mindig lehetnek nem valós élek a képen, melyeket okozhatnak zajok vagy színeltérések a durva felület miatt. Hogy ezeket megkülönböztessük egymástól, egy küszöböt kell használnunk, így csak egy bizonyos értéknél erősebb élek maradnak meg. A **kettős köszöb** az jelenti a Canny algoritmus esetében, hogy a felső küszöbnél erősebb pixelek értékeit „erősnek” jelöli meg, az alsó küszöb alatti értékeket elhagyja, a két küszöb közötti értékeket pedig „gyengének” jelöli.[2]

Az erősnek jelölt élek esetében biztosak lehetünk, hogy azok valódi élek, a gyengék esetében csak akkor lehetünk biztosak benne, ha valamelyik erős élhez kapcsolódnak, mivel a zaj és más apró eltérések valószínűleg nem eredményeznek erős éleket. Az **élkövetést** BLOB (Binary Large Object) analízissel lehet megvalósítani. Az élek pixeleit szét kell bontani összekapcsolt BLOB-okra, egy BLOB tartalmazza a középső pixelt és a 8 darab őt körülvevő pixelt is. Azok a BLOB-ok amelyek legalább 1 erős jelölést tartalmaznak megmaradnak, a többi nem.[2]

## Zárt téglalapok keresése

A sikeres éldetektálás után, a zárt téglalapok keresésének módszerével ki tudjuk használni a rendszámtábla alakját, mivel szinte minden esetben téglalap alakúak. A bináris képen meg tudjuk vizsgálni az élek alakzatát, hány oldaluk van, hol helyezkednek el. Ezekből az információkból ki tudjuk számolni a négyszögek magasságának és szélességének arányát, amiből tudunk következtetni hogy megfelelő méretű téglalap lehet e.

## Optikai karakterfelismerés

Az optikai karakterfelismerés célja, hogy a nyomtatott szöveget és képeket digitalizálja, azokat gépileg kezelhetővé tegye. Ez egy összetett probléma a nyelvek, betűtípusok sokfélesége miatt.[3]

Az optikai karakterfelismerés általános lépési:

* **Képbeszerzés** (Image acquisition)
* **Előfeldolgozás** (Preprocessing)
* **Karakter szegmentáció** (Character segmentation)
* **Jellemzők kinyerése** (Feature extraction)
* **Karakterek besorolása** (Character classification)
* **Utófeldolgozás** (Post processing)

A **képbeszerzés** után következő **előfeldolgozás** a képminőség javítását célozza. Itt különböző típusú szűrők alkalmazhatók, például átlagoló, minimum és maximum szűrők, ezek mellett különböző morfológiai műveletek is végezhetők (pl. erózió, tágítás, nyitás, zárás). Fontos része az előfeldolgozásnak a dokumentum ferdeségének kiderítése. Erre is többféle technológia létezik, például Hough-transzformáció, legközelebbi szomszédsági algoritmusok. Az előfeldolgozás végére már ismerjük a szövegsorok pozícióját, a pixelek vetületei vagy klaszterezése alapján.[3]

A **karakter szegmentációs** lépésnél a képet karakterekre kell bontani, ami megtörténhet explicit vagy implicit módon az osztályozási fázis közben is. Ezen kívül a többi fázis is segíthet a hasznos kontextuális információk megszerzésében a kép szegmentálásához.[3]

A **Feature extraction** szakaszban a karakterek különféle jellemzőit kell kivonni, amik képesek egyedileg azonosítani az adott karaktert, mint például a geometriai jellemzők (vonások, hurkok). Végső soron a kép méretét is csökkenteni kell bizonyos esetekben, melyekre szintén különböző technikák alkalmasak (pl. főkomponens-elemzés).[3]

Ezután a **karakterek besorolása**, osztályozása következik, amelynek a megvalósítása a képkomponensekben található kapcsolatokon alapul. Vannak statisztikai megközelítések, melyek egy osztályozó diszkriminációs függvény használatán alapulnak, ilyen pl. a Bayes-osztályozó, döntési fa-osztályozó, neurális hálózatokkal végzett osztályozás vagy a legközelebbi szomszédság módszerén alapuló osztályozók. Ezek mellet vannak olyan egyéb olyan osztályozó megoldások is, melyek szintaktikai megközelítsen alapulnak, ezek nyelvtani megközelítést feltételeznek.[3]

Az utófeldolgozás főbb célja a karakterfelismerés eredményének javítása. Itt lehet használni különböző megoldásokat, az egy ilyen megközelítés lehet az, hogy egynél több osztályozót használunk és összevetjük az eredményeket. Az eredmények javítása érdekében kontextuális elemzés is végezhető, ezek mellett a Markov-modelleken és szótáron alapuló lexikális feldolgozás is hasznos lehet.[3]

# Címsor 1

Abra

1. ábra: AbraAlairas
2. táblázat: TablazatCim

|  |  |
| --- | --- |
| TablazatFej | TablazatFej |
| TablazatCella | TablazatCella |
| TablazatCella | TablazatCella |
| TablazatCella | TablazatCella |

matematikaiFormula (1)

1. KodreszletCim

Kodreszlet

# Címsor 1

# Felhasznált irodalom

1. C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images", *Proceedings of the 1998 IEEE International Conference on Computer Vision*, Bombay, India.
2. Moeslund, T. (2009, március 23.), Canny Edge Detection.
3. Noman Islam, Zeeshan Islam, Nazia Noor, "A Survey on Optical Character Recognition System", *Journal of Information & Communication Technology-JICT Vol. 10 Issue. 2, December 2016*

# Mellékletek

1. Melleklet