

TDK-DOLGOZAT

Varjas Viktor

2013

**Szegedi Tudományegyetem
Természettudományi és Informatikai Kar
Képfeldolgozás és Számítógépes Grafika Tanszék**

**Gépkocsi felismerés
előlnézeti kép alapján**

TDK-dolgozat

Készítette:

Varjas Viktor
programtervező
informatikus MSc szakos
hallgató, II. félév

Témavezető:

Tanács Attila
egyetemi adjunktus

Szeged
2013

Tartalomjegyzék

Bevezetés.....	1
1. Gépkocsi felismerés szemből készült fényképek alapján, ismert rendszámtábla sarokpontokkal.....	4
1.1. A teszt adatbázis.....	4
1.2. A lényeges terület kijelölése a képen.....	7
1.3. Jellemző kinyerés.....	8
1.4. A jellemző vektorok osztályozása	9
1.5. Eredmények	10
2. Az osztályozó és a képjellemző vizsgálata	11
2.1. A távolságmetrika vizsgálata	11
2.2. Az SMG kép méretének vizsgálata	12
2.3. A tanító halmaz vizsgálata, “leave-one-out” keresztvalidáció	13
2.4. Osztályozás a 3 és 5 legközelebbi szomszéd alapján	15
2.5. Fourier képjellemző	16
3. A módszer robusztusságának vizsgálata	19
3.1. Perturbációs tesztek	19
3.2. Outlier kocsik detektálása	21
3.3. A Suzuki Swiftek szétválasztása, az évjárat megállapítása	23
4. Félautomata gépkocsi felismerés	24
4.1. A rendszámtábla seed pont alapú szegmentálása	24
4.2. A sarokpontok kinyerése.....	26
5. Teljesen automatikus gépkocsi felismerés	31
5.1. A ROI kijelölése.....	31
5.2. A rendszámtábla detektálása	31
5.3. A megfelelő seed pont kiválasztása	33
6. Az eredmények összefoglalása.....	36
6.1. Gyakorlati alkalmazások	36
Irodalomjegyzék	38

Bevezetés

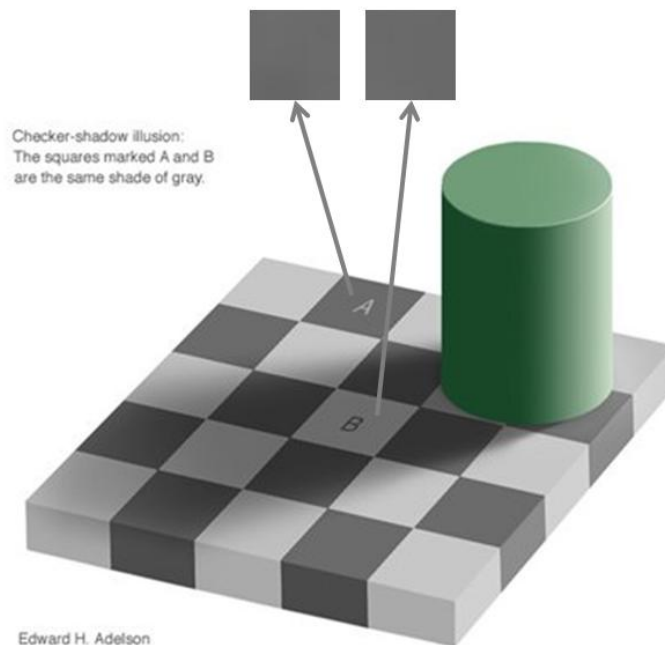
A fényképezés elterjedésével felfoghatatlan mennyiségű fénykép árasztotta el a világhálót. Az interneten elérhető szöveges tartalmak keresése már remekül megoldott probléma, amelynek a Google az úttörője. Természetesen mi nem csak a honlapok szöveges információi között szeretnénk keresni, hanem a fényképek között is. A Google rendelkezik ilyen funkcióval is, de alapértelmezetten szöveg/címke alapján keresik a képeket, így nem garantálható, hogy valóban az van az eredményképeken, amit mi szeretnénk látni rajta. Ennek több oka is lehet:

1. Félrecímkézett képek
2. Szavak többértelműsége
3. Nem megfelelő kereső kulcsszó

Szeretnénk képeket a képi tartalom alapján keresni valamint azt, hogy a lekérdezés ne szöveg alapján történjen, hanem egy másik kép legyen a keresés kulcsa. A Google ebben is remekül teljesít, hiszen lehetőségünk van adott képhez hasonló képek lekérésére, valamint feltöltött fénykép alapján keresni.

A feladat, hogy képeket az ábrázolt tartalom alapján keressünk, a számítógépeknek egy rendkívül bonyolult feladat, hiszen gyakran a kép magas szintű értelmezését követeli meg. Egy kép csak számok halmaza, amelyekből adott esetekben nagyon bonyolult eljárásokkal lehet olyan információkat kivonni, amire az emberi agy pillanatok alatt képes. Például szeretnénk olyan képeket keresni, amiken autó, elefánt, asztal, stb. található. Az emberek számára ez egy triviális feladat, hiszen elképesztően nagy a priori tudással rendelkezünk és egy még lenyűgözőbb „számítógéppel”, amit agynak hívnak. A szem a látás folyamatának csak egy apró szegmense, mindössze a külvilágból jövő információ összegyűjtése a feladata, a látás érzetét az agyunk állítja elő. Érdekes módon az agy nem csak előállít egy képet, de manipulálja is azt a sok év alatt megtanult előzetes tudása alapján, így nem mindig azt látjuk, amit a szemünk érzékel. Egy remek példa erre az 1. ábra. Bemutatja, hogy az agyunk milyen mértékben képes manipulálni a látottakat, annak érdekében, hogy olyan látvány táruljon elénk, amire „számítunk”. Az ábrán látható jelenség magyarázata, hogy az agyunk tisztában van vele, hogy sakktábla mintát lát, azaz minden mező 4-szomszédságában ellenkező színű mező van. Továbbá, hogy a hasáb árnyékot vet a táblára. Ez már elég is ahhoz, hogy az agyunk egy automatikus kompenzációt hajtson végre a látottakon és kivilágosítsa a „B” jelű mezőt. Érdeemes megfigyelni, hogy az árnyék miatt a „B”

mezőből mutató nyíl alsó felén is érzékelhető a világosítás. Az agyunk a képi tartalom elemzésében is kimagasló eredményekre képes.



1. ábra. Optikai csalódás¹. Az „A” és „B” négyzetek intenzitása pontosan megegyezik egymással.

A fent említettekből is érezhető, hogy a számítógépek jelenlegi tudása nem veheti fel a versenyt az emberi agy rendkívüli képességeivel, ha képi tartalom felismerése a cél, de megpróbáljuk közelíteni azt. A dolgozatomban egy, az ember számára triviális problémára kínál megoldást: gépkocsi gyártó és modell felismerés szemből. Természetesen az algoritmus általánosnak tekinthető olyan szempontból, hogy nem használok ki az algoritmus lényegi részénél, hogy itt kocsikról van szó, csak a ROI kijelölésénél, így számos egyéb alkalmazása is lehet módszernek.

Az algoritmus tárgyalása 3 fő részre bontható:

1. Manuális sarokpont megadás: Az algoritmus futásához szükséges rendszámtábla-sarokpont előre adottak. Tisztán a módszer hatékonyságát méri, nem terheli szegmentálási hiba az eredményeket. A gyakorlatban nem igazán alkalmazható.

¹ Edward H. Adelson, http://web.mit.edu/persci/people/adelson/checkershadow_illusion.html

2. Félautomata rendszámtábla szegmentálás: A sarokpontok nem ismertek, csak egy pontja a rendszámtáblának. Ez a megközelítés már közelít a gyakorlati alkalmazhatóság felé, de még felhasználói interakciót igényel.
3. Automata rendszámtábla detektálás és szegmentálás: A sarokpontok kinyerésének egy teljesen automatikus módja, nem igényel felhasználói beavatkozást. A gyakorlatban is alkalmazhatóvá teszi az autófelismerő módszert. Természetesen romlik az eredményesség, ami a rendszámtábla detektáló és szegmentáló eljárás fejlesztésével javítható.

A legnagyobb hangsúlyt a tesztek során a manuális sarokpontokkal adott tesztek kapják, hiszen azok jellemzik a módszer hatásosságát, robusztusságát. A dolgozat végén röviden összefoglalom az elért eredményeket és levonom a legfontosabb tanulságokat.

A teszteket egy AMD Phenom II X4 Black Edition (3.8 GHz) CPU-n futattam. A program párhuzamosítás nélkül, egy szálon futott.

1. Gépkocsi felismerés szemből készült fényképek alapján, ismert rendszámtábla sarokpontokkal

Ebben a fejezetben egy olyan algoritmust mutatok be, amely gépkocsik típusának és modelljének felismerésére képes, előlnézeti felvételek alapján. Első lépésben bemutatom a teszt adatbázist, majd egy, a szakirodalomban megtalálható algoritmust és annak implementációját [1, 2]. Ebben a fejezetben csak a szakirodalomban említett módszert fogom vizsgálni. Ez az algoritmus feltételezi a rendszámtábla sarokpontjainak pontos ismeretét, így azokat előre lementettem minden kocsinhoz. A következő fejezetben pedig az általam végzett módosításokat és a hozzájuk tartozó tesztelési eredményeket mutatom be.

1.1. A teszt adatbázis

Az adatbázist 827 szemből készült kép alkotja, és 11 osztály, amelyeket egy 3-4 hónapos periódus során gyűjtöttem Szegeden, az utcán parkoló autókról. Ezekből a képekből 727 kép azokról az autókról készült, amelyek a felismerés 10 osztályát képzik, a 11. osztály egy speciális „outlier” osztály, azaz olyan autókról is tartalmaz 100 képet az adatbázis, amelyek nem részei a felismerni kívánt 10 típusnak. Ezzel azt fogom mérni, hogy mennyire képes az algoritmus megmondani egy autóról, hogy része-e a felismerni kívánt modelleknek. A 10 osztályt az 1.1. ábrán lévő modellek alkotják.

Modell	Darabszám
Peugeot 206	55
Peugeot 307	47
Opel Astra	100
Skoda Fabia	100
Skoda Octavia	43
Volkswagen Golf	93
Volkswagen Polo	45
Ford Focus	47
Renault Megane	97
Suzuki Swift	100

1.1. táblázat. Modellek.

Az adatbázis gyűjtésének gyorsítása érdekében minden autóról azonnal 2 darab kép készült, de a második más szögből és/vagy távolságból. Ennek a módszernek az is előnye, hogy így jól megfigyelhetőek az esetleges osztályozási hibák távolság/szög összefüggései, hiszen előfordulhatnak olyan esetek, hogy az egyik képen szereplő autót sikerül felismerni, de a párját már nem. Egy modellen belül ugyanabból az évjáratból gyűjtöttem képeket a Suzuki Swift kivételével, ahol 2 különböző modelltől készültek

felvételek. Ezzel azt fogom vizsgálni, hogy mennyire képes az osztályozó kicsit eltérő autók között különbséget tenni.

Az adatbázis rendkívül változatos minden szempontból. Megvilágítás szempontjából előfordulnak éjszakai, hajnali, esős, árnyékos, tűző napos, valamint olyan képek is, ahol az autó egyik fele erősen megvilágított a másik pedig árnyékban van. Tehát sokféle, a gyakorlatban előforduló megvilágítási lehetőséget igyekeztem belevinni az adatbázisba. A kocsitól mért távolság is roppant változatos, előfordulnak közel 1 méterről készült képek, valamint 10 métert is meghaladó távolságok. A kamera szöge is hasonlóan változatos. A képek döntő része 2MP-en készült háromféle kamera felhasználásával:

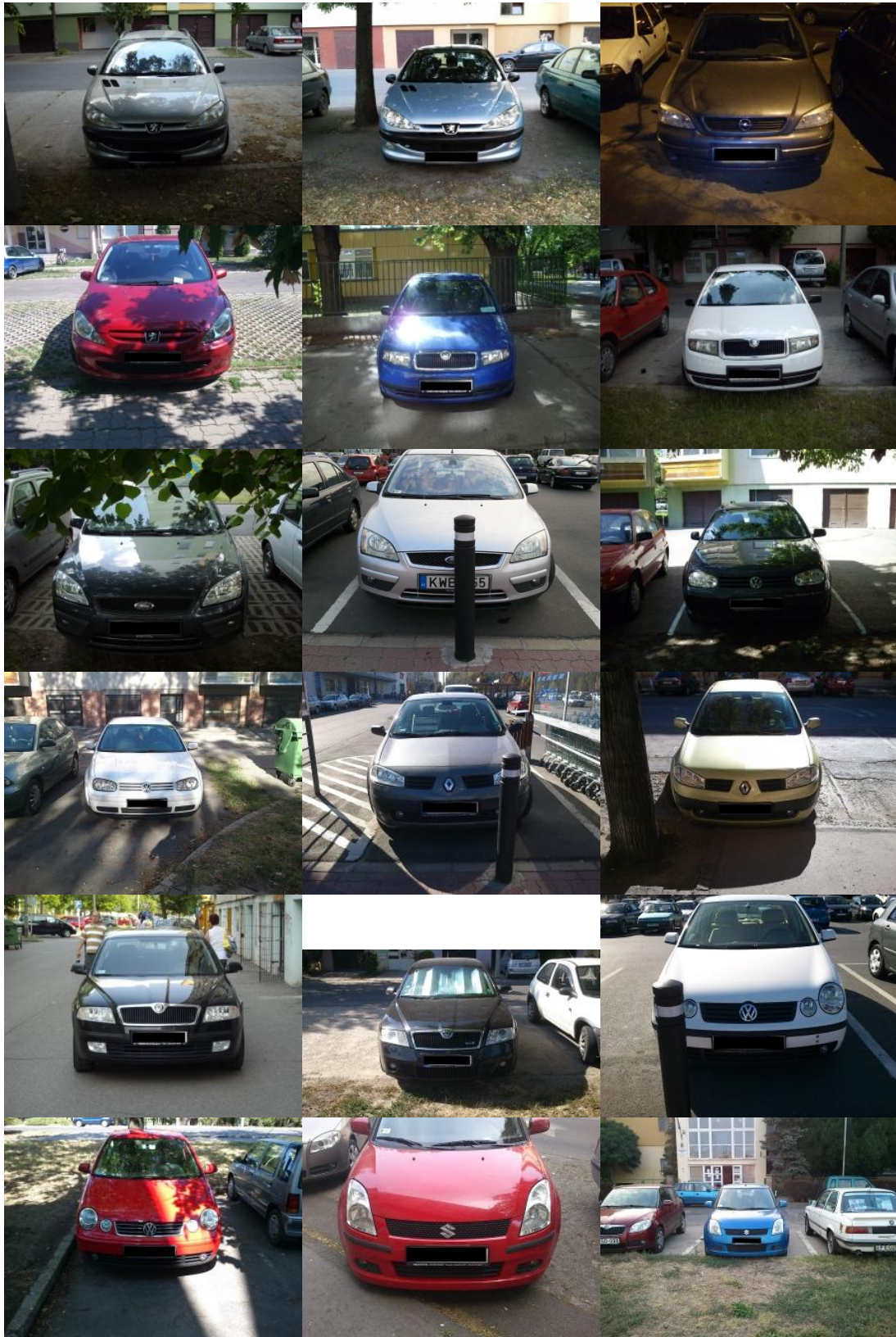
1. Pentax Optio (kompakt gép, 2MP, 4:3-as képarány)
2. Sony Ericsson Xperia Mini (2MP, 4:3-as és 16:9-es képarány)
3. Samsung Galaxy S (2MP, 4:3-as képarány)

A 3 különböző kamera használatát az indokolja, hogy így a módszer különböző minőségű képeken is kipróbálásra kerül. A 3 kamera eltérő minőségű képet produkál, ami különösen Xperia mini esetében vehető észre. A legjobb minőséget a Pentax produkálta, a legrosszabbat az Xperia.

A képkészítési protokoll megköveteli, hogy lehetőleg csak 1 kocsi szerepeljen a képeken, ami a kép középső részén helyezkedik el. Számos olyan kép is készült, ahol az autó egy lényeges részét egy oszlop vagy más hétköznapi akadály eltakarja, amelyek nagyon jól jellemzik majd az általam módosított módszer robusztusságát.

Minden képhez tartozik egy txt fájl is, amely tartalmazza az ideális rendszámtábla sarokpontokat, melyeket én, manuálisan jelöltem ki. Az [1, 2] cikkek által tárgyalt módszerek feltételezik, hogy ismerjük az ideális sarokpontokat. Az ideális sarokpontok ismeretében további, a robusztusságot vizsgáló tesztek is elvégezhetőek. A cikkekben 74 és 77 osztállyal dolgoznak és közel 1000 mintával, ami azt jelenti, hogy sokkal kevesebb minta jut egy adott modellre. Olyan eset is előfordul, hogy 2-3 képpel tesztelnek egy adott típust, ami nem mérvadó. Nálam kevesebb osztály, de sokkal több minta található osztályonként. Ezen az adatbázison jó szereplést várhatnánk a cikkben említett módszertől, de látni fogjuk, hogy a rugalmasabb protokoll alapján készült, valóság-hű képeket tartalmazó adatbázison nem szerepel elfogadhatóan, még 10 osztály esetén sem.

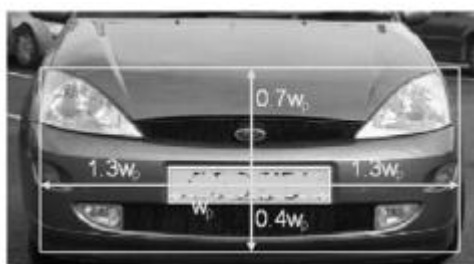
Összegzésként elmondható, hogy az adatbázis nem laboratóriumi körülmények között készült, hanem nagyon is valóságos, a hétköznapi életben várható képek fordulnak elő benne, amit rendkívül fontosnak tartok.



1.2. ábra. Mintaképek a teszt adatbázisból.

1.2. A lényeges terület kijelölése a képen

Az algoritmus kulcsfontosságú lépése egy olyan tartomány meghatározása a képen, amely egy állandó, modelleken belül nem változó részt határoz meg (ROI – Region Of Interest). Kézenfekvő az ötlet, hogy ezt a ROI-t például a rendszámtábla nagyon jól meghatározza, mert modellenként jó közelítéssel az autó elejének ugyanazon régiójára van rögzítve (1.3. ábra).



1.3. ábra. Egy lehetséges ROI kijelölés [1].

A kocsifelismerő eljárás csak a ROI-t használja fel az autó modelljének azonosítására, a többi képpont eldobható. Ha adott a rendszámtábla 3 sarokpontja, abból már megmondható a rendszámtábla hossza, magassága, középpontja, továbbá az a szög, amellyel a kép x-tengelyéhez képest el van fordulva a képen, így korrigálható az autó esetleges elforgása a képen. A ROI kijelölést elegendő az eredeti kép szürkeárnyaltos változatán elvégezni, hiszen a színinformáció semmilyen információt sem hordoz a képen látható kocsitípusról, így eldobható.

Ha adott a bal felső, jobb felső és jobb alsó sarokpont, melyeket rendre (x_1, y_1) , (x_2, y_2) , (x_3, y_3) jelölnek, akkor a rendszámtábla paraméterei könnyen és gyorsan megkaphatóak:

$$\begin{aligned} \text{középpont} &= \left(\frac{x_1 + x_3}{2}, \frac{y_1 + y_3}{2} \right) \\ \text{szélesség} &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ \text{elfordulás szöge} &= \frac{180}{\pi} \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \end{aligned}$$

Ezekből az adatokból már kijelölhető a ROI, amelyet célszerű a rendszámtábla hosszának függvényében megadni. Mielőtt kivágnám a ROI-t a képből, kompenzálom az esetleges elforgását az autónak. A képet visszaforgatom az elfordulás szögével, az új pixelértékek meghatározásához bilineáris interpolációt használok. Rendkívül fontos még a rendszámtábla maszkolása, hiszen azt semmiképpen sem szeretnénk, hogy a

rendszám táblán található betűk és számok bármilyen irányban befolyásolják az osztályozás kimenetelét, nem hordoznak semmilyen képi információt az autó típusáról.

A szakirodalomban említett ROI kijelöléstől mindössze annyiban tértem el, hogy vertikálisan 0.8 és 0.3 súllyal veszem a rendszám tábla hosszát, mert azt figyeltem meg, hogy így nagyobb eséllyel alkotják a ROI-t csak a kocsi pixelei. Tehát horizontálisan 2.6-szor akkora a ROI, mint a rendszám tábla szélessége, vertikálisan pedig 1.1-szer. A 1.4. ábra egy példát mutat be a ROI kijelölésre. Végül a ROI-t elsimítom egy 7x7-es maszkkal történő medián-szűréssel. A ROI önmagában még nem alkalmas az autók típusazonosítására, szükséges egy a modellekre jellemző képjellemzők kivonása.



1.4. ábra. A ROI kijelölése, a rendszám tábla maszkolása, valamint az esetleges elfordulás kompenzálása.

1.3. Jellemző kinyerés

A feldolgozás ezen fázisában adott egy olyan régió, amin csak az autót egy jól jellemző része van a képnek, nagyrészt az autó elejét alkotó képpontok. Ezen ROI méretek természetesen tetszőlegesen változatosak lehetnek a képmérettől és a kocsi fényképezőtől mért távolságától függően. Szükséges, hogy ezeket a képeket valamilyen módon egymással összevethetővé tegyük.

A ROI képből kiszámítunk egy Square Mapped Gradient (SMG) képnek nevezett jellemzőt, amelyet a vertikális és horizontális Sobel gradiens maszkokkal történő konvolúció után, a következő képlettel számolhatunk ki:

$$g_x = \frac{s_x^2 - s_y^2}{s_x^2 + s_y^2}, \quad g_y = \frac{2s_x s_y}{s_x^2 + s_y^2}$$

ahol s_x az x-irányú Sobel gradiens képet, és s_y pedig az y-irányút jelöli.

Ez a képi jellemző nagyon jól kezeli a megvilágításbeli különbségeket, hiszen csak a vertikális és horizontális gradiensekre támaszkodik. Látszik a képleten, hogy a homogén területek esetén numerikus problémák jöhetnek elő, amit a gradiens képek kiszámítása után kezelni kell. Az eredményképeken a következő esetekben nullázok ki pixelértékeket:

- Ha nem szám jön ki (NaN)
- $\pm\infty$ esetén
- Ha nagyobb, mint 100000 vagy kisebb, mint -100000

Az így megkapott képeket átméretezzük egy előre meghatározott méretre, ami a [2] cikk szerint 162x68 pixel (magasság x szélesség). Ezt követően egy jellemző vektort hozunk létre, amit úgy kapunk meg, hogy az x- és y-irányú SMG képeket egy vektorra konkaténáljuk, ami ebben az esetben egy 22032 dimenziós vektort fog eredményezni, amit jelöljön F . Végül a vektor értékeit átskálázom a $[-1,1]$ intervallumba:

$$F(i) = \frac{F(i) - \text{Min}(F)}{\text{Max}(F) - \text{Min}(F)} \cdot 2 - 1, \text{ ahol } i = 1, 2, \dots, |F|$$



1.5. ábra. A ROI, valamint az x- valamint y-irányú SMG feautere képek.

1.4. A jellemző vektorok osztályozása

Ebben a 22032 dimenziós térben kellene megmondani egy jellemző vektorról, hogy melyik osztályba tartozik. Erre a célra a legközelebbi szomszéd módszert alkalmazzuk. A szakirodalomban leírtak alapján, két vektor távolságán az általuk bezárt szöget fogjuk érteni, tehát a koszinusz távolságmetrikát használjuk. Ez a választás logikusabbnak tűnik az egyszerű Euklideszi távolsággal szemben, de amint később látni fogjuk, messze nem a legjobb választás. Lehetőség van nem csak a legközelebbi szomszéd keresésére, hanem a 3 vagy akár több legközelebbi megkeresésére, majd többségi szavazással eldönteni, hogy az adott vektor melyik osztályba tartozik.

Az osztályozás minden kocsni esetében 3-4 modell alapján történik úgy, hogy a modellek nem részei a tesztelő halmaznak. Ezeket a modelleket a jó minőségű képek közül választottam ki és igyekeztem úgy megadni őket, hogy lefedjék az esetleg rendszámtábla típus eltéréseket, hiszen több típusú rendszámtábla is forgalomban van, ami befolyásolja a ROI kijelölését és így az osztályozást is.

1.5. Eredmények

Az általam készített, a valósághoz nagyon közel álló teszt adatbázison a 162x68 pixelre átméretezett SMG képek legközelebbi szomszéd módszerrel történő osztályozása, a vektorok bezárt szögét figyelembe véve nem szerepelt jól. Az eredményeket az 1.6. táblázat foglalja össze. Egy autó felismerésének futási ideje átlagosan 3.5 másodpercet vett igénybe.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	25	30	45,5%
307	47	21	26	44,7%
Astra	100	61	39	61%
Fabia	100	54	46	54%
Focus	43	15	28	34,9%
Golf	93	26	67	28%
Megane	45	18	27	40%
Octavia	47	29	18	61,7%
Polo	97	65	32	67%
Swift	100	14	86	14%
Összegzés	727	328	399	45,1%

1.6. táblázat. Eredmények: SMG[162x68]_cosine.

A táblázatból kiolvasható, hogy a módszer mindössze 45,1%-os eredmény elérésére képes egy, a gyakorlathoz nagyon közel álló adatbázison, ami alkalmazhatatlanná teszi a módszert még abban az ideális esetben is, amikor pontosan adottak a rendszámtábla sarokpontok, nem terheli őket szegmentálási hiba. A következő fejezetben az osztályozót fogom alaposabban megvizsgálni, mert maga a kivont képi jellemző jó, csak az osztályozási algoritmuson kell változtatni, amint azt látni fogjuk.

2. Az osztályozó és a képjellemző vizsgálata

Ebben a fejezetben bemutatam az általam kitalált módosításait a módszernek. Vizsgálom az osztályozó módszert, továbbá azt, hogy különböző méretű képjellemzők esetén milyen eredményt ér el a módszer. Egy általam jó képjellemzőnek gondolt Fourier transzformáción alapuló jellemző is bemutatásra kerül, ami meglehetősen jól szerepel, de elmarad az SMG jellemzőtől.

2.1. A távolságmetrika vizsgálata

A [2] cikkben a „cosine” távolságmetrikát használják, ami az osztályozni kívánt vektorok által, a jellemzők térben bezárt szöget jelenti. Ebben az esetben képeket szeretnénk osztályozni, így hiába alakítjuk át vektor reprezentációra a háttérben akkor is képek között szeretnénk távolságot meghatározni. Ebből kiindulva célszerű lecserélni a távolságmetrikát a korrelációnak nevezett metrikára. Adott X és Y vektorok között a korrelációs távolságot a következő képlettel számoljuk:

$$1 - \left| \frac{\frac{1}{n} \sum_{i=1}^n X_i Y_i - \mu_X \mu_Y}{\sigma_X \sigma_Y} \right|,$$

ahol σ a szórást, míg μ a várható értéket jelöli, az abszolút értékben lévő korreláció értéke pedig a $[-1, 1]$ intervallumról kerül ki. -1 vagy 1 közelében van, ha korrelál a 2 vektor, 0 felé közeledve pedig egyre kevesebb kapcsolat van a két vektor között. A számlálóban a vektorok kovarianciája található, tehát a korreláció a normalizált kovariancia, aminek nagysága jellemző a vektorok lineáris relációjára. A tört abszolút értékben nagy szám lesz, ha korrelálnak a vektorok, így az $1 - \text{korreláció}$ egy kicsi érték lesz.

A metrika lecserélése a korrelációra rendkívüli javulást eredményezett: 45,1%-ról felugrott a találati arány 96,4%-ra, ami nagymértékű javulásnak mondható egy ilyen „nehéz” képeket tartalmazó adatbázison. Az új távolságmetrika bevezetésével elért eredményeket a 3.1. táblázat foglalja össze. Látható, hogy rendkívüli javulást eredményez, ha statisztikai szempontból vizsgáljuk meg a jellemző vektort (képet) és nem azt figyeljük, hogy mekkora szöget zárnak be egymással a jellemző térben.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	47	0	100%
Astra	100	100	0	100%
Fabia	100	99	1	99%
Focus	43	43	0	100%
Golf	93	88	5	94,6%
Megane	45	45	0	100%
Octavia	47	47	0	100%
Polo	97	85	12	87,6%
Swift	100	92	8	92%
Összegzés	727	701	26	96,4%

2.1. táblázat. Eredmények: SMG[162x68]_correlation.

2.2. Az SMG kép méretének vizsgálata

Felmerült a kérdés, hogy szükséges-e a jól teljesítő korrelációs metrika mellett 162x68-as képekkel dolgozni, amelyekkel egy 22032 dimenziós képjellemző térben kell keresni és olyan módon skálázni a képet, hogy horizontálisan rengeteg információt eldobunk. Ekkora térben keresni időigényes feladat, arról nem is beszélve, hogy az eredményeken sem javít a nagyobb számítást igénylő képméret, ahogy az a következő eredményekből látható.

Több képméret kipróbálása után azt tapasztaltam, hogy elegendő mindössze 34x80 pixel méretű SMG képekkel dolgozni. Ebből az is látszik, hogy a kép magasságához mérten nagymértékű horizontális alulmintavételezést is megszüntettem. Ekkora képekkel egy mindössze 5440 dimenziós vektortérben kell keresni, ami drasztikusan lerövidíti az osztályozási procedúrát. Egy kocsi átlagos osztályozási ideje 0.5 másodpercre csökkent. A futási időn túl az eredmények is javultak, ami annak köszönhető, hogy a kisebb képek képesek elnyelni olyan ROI kijelölési pontatlanságokat, amelyek egy nagyobb képen még látszódnak, így növelve a fedésben lévő pixelek számát. Az is gyakran előfordul, hogy a rendszámtábla alapból ferdén van felszerelve vagy eldeformálódott. A kisebb méretű képek ezeket a hibákat is képesek lehetnek tolerálni.

A 34x80-as SMG képpel kapott eredmények még az eddigi eredményeket is felülmúlták. Mindössze 4 kocsit hibázott el az osztályozó a 727-ből, így 99,4%-os, közel tökéletes találati arányt ért el. Az eredményeket a 2.2. táblázat foglalja össze.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	47	0	100%
Astra	100	100	0	100%
Fabia	100	100	0	100%
Focus	43	43	0	100%
Golf	93	93	0	100%
Megane	45	45	0	100%
Octavia	47	47	0	100%
Polo	97	94	3	96,9%
Swift	100	99	1	99%
Összegzés	727	723	4	99,4%

2.2. táblázat. Eredmények: SMG[34x80]_correlation.

Több képméretet is kipróbáltam és azt tapasztaltam, hogy a SMG képek méretének csökkenése, kevés osztály esetén egyre robusztusabbá teszi a rendszert, de ez természetesen csak egy bizonyos határig igaz. Ha túl kicsi a kép, akkor annyi információ veszik el, hogy drasztikusan leromlanak az eredmények. Sok osztály esetén nem érdemes a fent tárgyalt képméret alá menni.

2.3. A tanító halmaz vizsgálata, “leave-one-out” keresztvalidáció

A továbbiakban a legjobbnak bizonyult 34x80-as SMG jellemzőket fogom használni minden teszt során, egy kivétellel, amikor a Fourier jellemzőt tárgyalom a fejezet végén. A távolságmetrika pedig a korreláció lesz. A Fourier képjellemzőtől eltekintve, egyedül a tanító halmaz különböző módosításait fogom vizsgálni.

Az eddig tárgyalt eredmények úgy születtek, hogy modellenként 3-4 kép került bele a tanító halmazba, aminek jó hatása van a futási időre, hiszen minél kevesebb elemű a tanító halmaz, annál kevesebb elemmel kell összevetni az osztályozandó elemet. Ebben az esetben 42 elemű a tanító halmaz, mert minden típushoz hozzáadtam egy speciális tanító elemet, aminek az automatizálás során lesz jelentősége, így majd később tárgyalom. Felmerül a kérdés, hogy mennyire befolyásolja az osztályozás sikerességét a tanító halmaz mérete, a jellemző tér nagyobb mértékű telítettsége. A kérdés megválaszolása érdekében 2 tesztet hajtottam végre.

Az első teszt során minden osztályra vettem az első 20 képet az adatbázisból és beletettem a tanító halmazba. Így egy 200 elemű halmaz jött létre, amelyet 527 mintával

teszteltem, ahol a minták teljesen függetlenek a tanító halmaztól. Az eredményeket az alábbi 2.3. táblázat foglalja össze.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	35	35	0	100%
307	27	27	0	100%
Astra	80	80	0	100%
Fabia	80	80	0	100%
Focus	23	23	0	100%
Golf	73	73	0	100%
Megane	25	25	0	100%
Octavia	27	27	0	100%
Polo	77	77	0	100%
Swift	80	80	0	100%
Összegzés	527	527	0	100%

2.3. táblázat. Eredmények: SMG[34x80]_correlation. Minden osztály első 20 képe alkotja a tanító halmazt.

Az eredményekből látszik, hogy a tanító halmaz moderált növelése egyértelműen növeli a találati arányt, mégpedig 100%-ra. Egyetlen autót sem osztályozott félre a módszer. A futási idő átlagosan 2.1 másodperc volt.

A második teszt egy validációs teszt, amelyet angolul leave-one-out cross-validation néven szoktak említeni [4]. A leave-one-out arra utal, hogy ennél a tesztnél az egész adatbázist tanító halmazként kezeljük, annak a képnek a kivételével, amit éppen osztályozni szeretnénk és így teszteljük az összes képet az adatbázisban. Ennek a tesztnek a futási ideje jelentősen nagyobb, mint az eddigieknek, de rendkívül jól szemlélteti a módszer viselkedését nagyszámú tanító minta esetében. Egy minta kihagyása a tanító halmazból tulajdonképpen a k-szoros keresztvalidáció extrém esete, ahol minden mintapartíció 1 elemű. Az én adatbázisom struktúrájából adódóan nem tehetem meg, hogy minden kocsit csak az aktuálisat veszem ki a tanító halmazból, hiszen a kocsiról 2-2 kép készült eltérő távolságból/szöggel, így a 2. kép az adott kocsiról túl jó tanító minta lenne, ezért minden osztályozás nagy valószínűséggel sikeres kimenetelű lenne. Ezt elkerülendő, mindig kiveszem az aktuális kocsit is a tanító halmazból, így 725 elemű tanítóval történik az osztályozás. A leave-one-out keresztvalidációs teszt eredményeit a 2.4. táblázat szemlélteti. A teszt eredményei nagyon ígéretesek, mindössze 1 kocsit hibázott el, így 99,9%-os eredményt ért el. A futási idő pedig 9.3 másodperc körül mozgott. A 725 elemű tanító halmaz remek szereplése több szempontból is pozitív. Az első, hogy 10 osztály esetén telítettebbnek

mondható jellemző térben is kiválóan szerepel a módszer. A második pedig, hogy olyan modellek is bekerültek ezzel a módszerrel a tanító halmazba, amiknek amúgy nem lenne ott a helyük. Itt olyan képekre gondolok, amik nagyon rossz megvilágítási viszonyok között készültek vagy részben kitakartak, így valójában nem alkalmasak a tanító szerepre. Ebből megfigyelhető a rendszer robusztussága, hiszen a zajosnak mondható tanító halmaz ellenére szinte hibátlan eredményeket sikerült elérni.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	47	0	100%
Astra	100	100	0	100%
Fabia	100	100	0	100%
Focus	43	43	0	100%
Golf	93	93	0	100%
Megane	45	45	0	100%
Octavia	47	47	0	100%
Polo	97	96	1	99%
Swift	100	100	0	100%
Összegzés	727	726	1	99,9%

2.4. táblázat. Eredmények: SMG[34x80]_correlation. Leave-one-out keresztvalidáció teszt.

Összefoglalva a tanító halmaz vizsgálatát elmondható, hogy a módszer nagyon jól skálázható. Az eredmények 20-30 tanító kép esetén is kiválóak, de tovább javíthatóak a halmaz számosságának sokszorozásával. Ebből következően a felhasználási területhez igazítható. Ahol megbízható, de valós idejű futásra van szükség, ott redukálni lehet a modellek számát. Olyan alkalmazások esetén, ahol viszont nem lényeges a futási idő (például offline elemző szoftverek esetén), nagyszámú tanító mintával nagyon stabil eredmény érhető el.

2.4. Osztályozás a 3 és 5 legközelebbi szomszéd alapján

A legközelebbi szomszéd keresésénél nem feltétlenül kell az adott vektorral legjobban korreláló vektor címkéjével azonosítanunk a megoldásunkat. Megtehetjük azt is, hogy a korreláció mértéke alapján növekvő sorrendbe rendezzük a tanító halmazt és vesszük az első n legkisebb elemet. Ha adott az első n legközelebbi szomszéd, akkor több lehetőségünk is van eldönteni, hogy mi legyen az eredmény címke. A következő tesztek során a 3 legközelebbi szomszédot vizsgáltam és azt a címkét választottam, amely többségben van, egyenlőség esetén a legközelebbi vektor címkéje a megoldás.

Elsőként a modellként 3-4 tanító elemet használó osztályozót teszteltem. Az eredmények egy eset kivételével megfelelnek a 2.2. táblázatban ismertetett eredményeknek. Az egyedüli eltérést egy Golf okozta, ami tévesen lett osztályozva. Ebből is látszik, hogy a több szomszéd vizsgálata egy kétélű fegyver, hiszen jó osztályozásokat elronthat, azonban bizonytalanságokat pedig megerősíthet. Ebben az esetben azért nem meglepő a kisarányú romlás, mert nagyon kis elemszámú a tanító halmaz, így könnyedén fölénybe kerülhet egy hibás címke. Nagyméretű tanító halmaz esetén és sok osztály esetén, azonban nagy eséllyel az eredmények javulását eredményezi. Ezért kipróbáltam a korábban 100%-osan teljesítő 20-20 tanító elemet használó osztályozót, hogy megfigyelhető-e elbizonytalanodás, ha nem csak a legközelebbi szomszédot vesszük figyelembe. Ebben az esetben 3 és 5 legközelebbi szomszédot is vizsgáltam. Mind a két esetben 100%-os találati arányt ért el a módszer, tehát a több tanító kép bevezetése, még az 5 legközelebbi szomszéd esetében sem okoz téves osztályozást. Nagy osztályszám és nagy tanító halmaz esetén nagy valószínűséggel robusztusabbá tenné a módszert, ha az első 3, 5 vagy a tanító halmaz méretének függvényében akár több legközelebbi szomszéd alapján hoznánk döntést. A leave-one-out keresztvalidációt is kipróbáltam 3 legközelebbi szomszéd módszerével, ahol az adatbázisomhoz mérten maximális a tanító halmaz mérete. Az eredmények pontosan megegyeznek a legközelebbi szomszéd eredményeivel (2.4. táblázat).

Összességében nem igazán lehet messzemenő következtetést levonni az n legközelebbi szomszéd figyelésének hatásáról. Annyi érezhető csak, hogy kevés osztály esetén nagy eséllyel bizonytalanságot visz a rendszerbe, míg nagy tanító halmaz esetén nem változtat az eredményeken vagy robusztusabbá teszi az osztályozót.

2.5. Fourier képjellemező

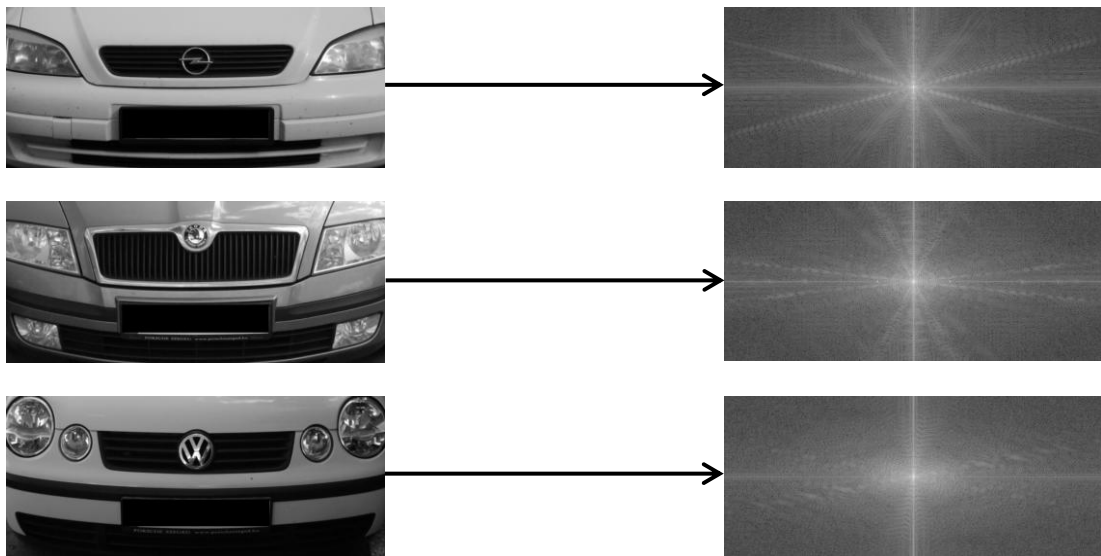
Az SMG jellemzőkön túl számos egyéb képi jellemző kivonása is elképzelhető. Az [1, 2] cikkekben több ilyen módszer is bemutatása kerül, de mindegyik rosszabbul teljesít, mint az SMG képjellemező. Az algoritmus implementálása közben felmerült bennem, hogy esetleg a ROI Fourier transzformáltját is fel lehetne használni, mint képi jellemzőt és így a Fourier képek terében keresnénk a legközelebbi szomszédját egy adott modellnek. A Fourier transzformált felhasználása mellett, mint képi jellemző szól az, hogy eltolás invariáns, így az esetleg eltolás erejéig felmerülő ROI kijelölési hibát a Fourier kép elnyeli, ezért nem befolyásolja az osztályozást. Egy $M \times N$ -es 2 dimenziós

szürkeárnyaltos kép diszkrét Fourier transzformáltja a következő képlettel számítható [3]:

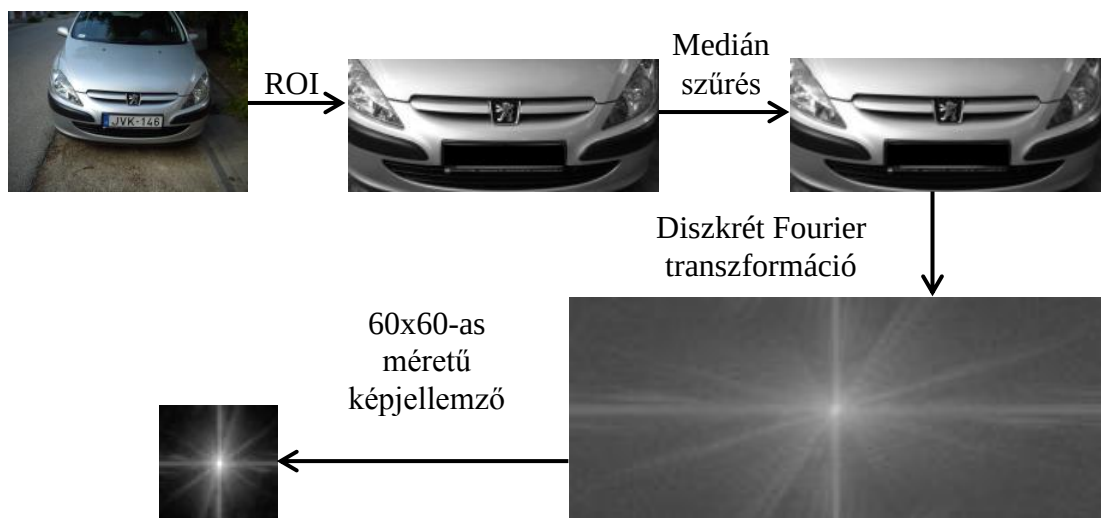
$$F(X, Y) = \frac{1}{MN} \sum_{x=1}^M \sum_{y=1}^N f(x, y) e^{-2\pi i (xX/M + yY/N)},$$

ahol f a bementi kép, $x = 1, 2, \dots, M$, $y = 1, 2, \dots, N$.

Mielőtt frekvencia térbe transzformálok a képet egy medián szűrést hajtok végre rajta, 7x7-es kernellel. Ezt követően a legalacsonyabb frekvenciát a kép közepére tolom, majd veszem a $\log(|F| + 1)$ valós képet és 60x60-as képpé skálázom át. Végül sorfolytonosan vektorrá alakítom.



2.5. ábra. ROI-k és a hozzájuk tartozó Fourier transzformáltak.
Látható, hogy általában modellenként jellegzetes a frekvencia kép.



2.6. ábra. A Fourier képi jellemző előállításának menete.

A Fourier jellemzők osztályozásához számos távolságmetrikát kipróbáltam, többek között az euklideszi távolságot és a jellemző vektorok bezárt szögét, de egyik sem tudott a korrelációs metrika közelébe kerülni. Az 2.7. táblázat összefoglalja a Fourier jellemző, korrelációs távolságmetrikával elért eredményeit. Egy felismerés átlagosan 0.3 másodpercet vett igénybe.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	46	9	83,6%
307	47	44	3	93,6%
Astra	100	77	23	77%
Fabia	100	94	6	94%
Focus	43	38	5	88,4%
Golf	93	65	28	69,9%
Megane	45	35	10	77,8%
Octavia	47	38	9	80,9%
Polo	97	71	26	73,2%
Swift	100	76	24	76%
Összegzés	727	584	143	80,3%

2.7. táblázat. Eredmények: Fourier[60x60]_correlation.

3. A módszer robusztusságának vizsgálata

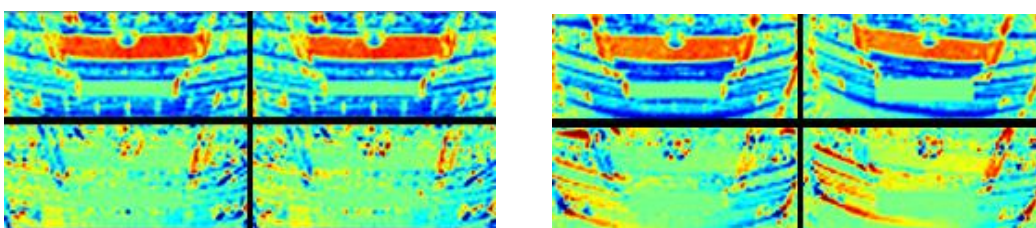
A robusztusságot egy perturbációs teszt segítségével mérem, melynek az a lényege, hogy az ideális, előre adott sarokpontokat valamilyen matematikai modell alapján elmozdítom és vizsgálom minden típus 3 modelljére az osztályozó által adott eredményt. Megvizsgálom továbbá azt is, hogy képes-e a rendszer megbízhatóan kívülálló autókat kiszűrni (outlier), tehát jelezni azt, ha a kereső bemenete egy olyan autó, ami nem szerepel az adatbázisban. Végül megvizsgálom, hogy sikerül-e elkülöníteni a 2 különböző évjáratú Swiftet.

3.1. Perturbációs tesztek

A gyakorlati alkalmazhatóság érdekében nagyon fontos megvizsgálni a módszer érzékenységet a sarokpontok kijelölésére, hiszen egy valós alkalmazásban szinte elképzelhetetlen, hogy tökéletes szegmentálását adjuk a rendszám táblának. Ebben a részben már csak a legjobban szerepelt módszert fogom vizsgálni, tehát 34x80-as SMG jellemzőket, korrelációs távolságmetrikával. Erre a célra mind a 10 különböző típus 3 véletlen választott modelljét fogom megvizsgálni. Az ideális rendszám tábla sarokpontokat egy normális eloszlású véletlen számmal helyettesítem, melynek várható értéke az aktuális koordináta, szórása pedig a modellezni kívánt szegmentálás minőségétől függ:

$$(x, y) \rightarrow (N(x, \sigma^2), N(y, \sigma^2)),$$

ahol (x, y) a rendszám tábla egy ideális sarokpontját jelöli. Ezt minden kiválasztott modellel, 1000 alkalommal végzem el és kiszámolom, hogy milyen arányban sikerült az 1000 vizsgálatból sikeres felismerést végrehajtani. Ez a módszer statisztikailag nagyon jól jellemzi a módszer robusztusságát. Az egész mérésorozatot 4 különböző várható értékre futtatom le: $\sigma = 4, 8, 12, 18$, ahol a szórás értékek pixelben értendők. A 4 várható értékű egy nagyon jól működő szegmentálást szimulál, míg a 8 és 12 egy átlagosan teljesítőt, végül pedig egy rosszul működőt próbálok modellezni.



3.1. ábra. A perturbáció hatása az SMG képekre. Bal oldalon $\sigma = 4$, jobb oldalon $\sigma = 18$

A 3.1. ábra szemlélteti a perturbáció képjellemzőre gyakorolt hatását. Egy képnégyes bal felső sarkában található az x-irányú SMG jellemző, alatta pedig az y-irányú. Mellettük, a jobb felső- és alsó sarokban pedig a perturbáció hatása található. Jól látható, hogy a kicsi szórás hatása szinte nem érzékelhető a képek ilyen mértékű visszaskálázása mellett, de a nagy szórás már szemmel láthatóan megváltoztatja a kocsik elhelyezkedését. A rosszul kijelölt sarokpontok természetesen forgatási hibát is előidéznek a képjellemzőn, hiszen a ROI kijelölés előtt egy automatikus forgáskompenzálás is végrehajtódik, ami rosszul kijelölt sarokpontok esetén hibás szöggel fog forgatni. A perturbációs tesztek eredményeit az alábbi táblázatok foglalják össze:

$\sigma = 4$	1	2	3		$\sigma = 8$	1	2	3
206	100%	100%	100%		206	100%	99,6%	99,3%
307	100%	99,9%	99,9%		307	99,2%	95,5%	95,5%
Astra	99,9%	100%	100%		Astra	96,4%	99,4%	99,9%
Fabia	100%	100%	99,8%		Fabia	98,9%	98,3%	95,3%
Focus	99,7%	100%	100%		Focus	90%	99,3%	99,4%
Golf	100%	100%	100%		Golf	100%	97,1%	99,5%
Megane	100%	100%	99,6%		Megane	99,3%	99,7%	90,3%
Octavia	100%	100%	100%		Octavia	100%	100%	100%
Polo	97,1%	95,3%	74,6%		Polo	62,4%	61,9%	39,1%
Swift	100%	99,9%	100%		Swift	99,9%	95,2%	99,7%

$\sigma = 12$	1	2	3		$\sigma = 18$	1	2	3
206	99,5%	97,4%	96,5%		206	91,9%	88,0%	86,0%
307	93,1%	86,4%	89,7%		307	78,4%	65,9%	77,9%
Astra	90,1%	97,3%	98,9%		Astra	77,3%	88,0%	92,1%
Fabia	91,9%	95,1%	86,9%		Fabia	84,4%	82,8%	74,2%
Focus	71,7%	93,4%	92,4%		Focus	44,9%	77,1%	70,7%
Golf	99,8%	87,5%	98,5%		Golf	97,7%	71,4%	88,2%
Megane	94,0%	95,9%	74,3%		Megane	77,2%	80,4%	59,7%
Octavia	100,0%	100,0%	100,0%		Octavia	98,3%	99,7%	98,4%
Polo	36,6%	41,7%	23,1%		Polo	22,7%	24,1%	14,2%
Swift	98,2%	88,7%	97,7%		Swift	93,6%	82,5%	91,4%

3.2. táblázat. A perturbációs teszt eredményei 1000 kísérletből. Az oszlopok minden típushoz a 3 véletlenszerűen kiválasztott modellhez tartozó sikeres felismerés arányát tartalmazzák.

A táblázatok első oszlopa jelzi a modellt, míg az első sor a véletlenszerűen kiválasztott 3 modell sorszámát jelöli. A beírt végeredmények az 1000 kísérletből elért sikeres felismerések arányát jelentik. A bal felső táblázatból kiolvasható, hogy egy kicsi

szegmentálási hiba tulajdonképpen nincs hatással az osztályozás eredményességére, de még egy közepesen jó ($\sigma = 8$) szegmentálóval is stabil osztályozást lehet biztosítani. Továbbá az is megfigyelhető, hogy a robusztusság modellfüggő is, hiszen a 3 kiválasztott Polo modell esetében már közepes szegmentálási hiba is jócskán lerontja az eredményeket. A nagy meglepetést a rossz szegmentálási modell adta, hiszen a 30 tesztsorozatból 24 alkalommal 70% felett teljesített az osztályozó, ami a módszer robusztusságát bizonyítja.

Összességében az elmondható a módszerről, hogy rendkívül jól tolerálja a rendszámtábla sarokpontjainak kijelölési pontatlanságát, ami nélkülözhetetlen ahhoz, hogy gyakorlati alkalmazásokban is használható legyen a módszer.

3.2. Outlier kocsik detektálása

Megvizsgáltam, hogy a módszer mennyire képes az adatbázisban nem szereplő autókról megállapítani, hogy „kívülállók”, továbbá, hogy az adatbázisban szereplő autókról mekkora eséllyel válaszolná azt, hogy nem részei az adatbázisnak.

A kívülállók teszteléséhez egy 100 képet tartalmazó adatbázist használtam, melyek nem szerepelnek a felismerni kívánt kocsik között. Az osztályozó minden esetben egy valós számot ad vissza, amely azt méri, hogy mennyire „korrelál” a legközelebbi modell az input képpel. Ezt a számot megfelelően küszöbölve egészen jó arányban tud olyan diplomatikus választ adni az osztályozó, hogy „Az adott autó nagy valószínűséggel nem szerepel az adatbázisban. A hozzá legközelebb álló modell: xy”. Ebből jól látszik, hogy ez a küszöb nem egyértelműen határozza meg a kívülállókat, de hasznos jelzés lehet a felhasználó felé, valamint a robusztusságot és azt is jellemzi, hogy hogyan viselkedne az algoritmus egy sokkal telítettebb jellemző térben, sok osztállyal. A küszöbszámot 0.55-ra állítottam úgy, hogy a korreláció metrika eredménye a $[0,1]$ intervallumból kerül ki, ahol 0 tökéletes egyezés.

Első esetben 3-4 modellt tartalmazó tanító halmazt használó módszert vizsgáltam meg. A 100 outlier kocsik közül 96 esetben sikerült megállapítani, hogy nem részei az adatbázisnak, ami elfogadható eredmény. Ezt követően azt vizsgáltam, hogy mennyi, ténylegesen az adatbázisban szereplő kocsi kapná meg a kívülállók címkét ezzel a küszöbszámmal. A teszt során 115 kocsi lett tévesen kívülállók osztályba sorolt, ami 15,8%-os hibaarány.

Második körben az osztályonként 20-20 modellt tartalmazó tanító halmazzal elérhető eredményt vizsgáltam. Azt vártam, hogy ez javítani fog az eredményeken a 2-3 modellel szemben, hiszen 100%-os találati arányt értem el vele az osztályozás során. Az eredmények a vártak megfelelően alakultak, hiszen a 100-ból 83 autóról állította meg helyesen, hogy kívülálló. Ez persze romlás az előző eredményekhez képest, de az adatbázisba tartozók közül pedig mindössze 42 lett tévesen kívülálló az 527 esetből, ami 8%-os hibaarányt jelent. Említésre méltó még, hogy ezekből az esetekből a Swift 22-t tudhat magáénak, amiről az látszik, hogy a Swifték között nagyobb korrelációs távolságok is előfordulhatnak.

Végül a 2 kép híján maximális elemszámú tanító halmazt teszteltem, tehát a leave-one-out keresztvalidációt futtatom le. A 100 kívülállóból 80 kapott helyes címkét, ami elfogadható teljesítménynek minősül. A 727 kép közül azonban mindössze 35 kapott tévesen outlier címkét, ami mindössze 4,8%-os hibaarány, amiből ebben az esetben is a Swifték dominálnak 19 téves esettel.

Az előző alfejezetben említett perturbációs tesztet végrehajtottam az outlier osztályból származó 3 véletlen kocsihoz. Ezzel azt szerettem volna vizsgálni, hogy egy olyan autón rosszul kijelölt sarokpontok, amely nem része az adatbázisnak, mekkora eséllyel következik be az, hogy a hibás ROI kijelölés miatti jellemző kép jobban korrelál egy tanult modellel. A teszt ebben az esetben is 4 féle szórás értékkel futott, kocsinként 1000-1000 iterációt. Az eredményeket az alábbi táblázat összegzi:

outlier	1	2	3
$\sigma = 4$	100%	99,5%	100%
$\sigma = 8$	100%	97,7%	100%
$\sigma = 12$	100%	97,2%	100%
$\sigma = 18$	100%	98,1%	100%

3.3. táblázat. A perturbációs teszt eredményei 1000 kísérletből. Az oszlopok minden szóráshoz kiválasztott 3-3 modell eredményeit tartalmazzák.

A perturbációs eredmények a várt módon alakultak. Látszik, hogy bármilyen rosszul vannak kijelölve az outlier kocsi rendszámtábla pontjai, akkor is ugyanakkora eséllyel kerül rájuk az outlier címke, hiszen nagyon kicsi a valószínűsége, hogy pont egy tanító halmazbeli modellhez nagyon hasonló képi jellemző keletkezzen.

Összességében elmondható, hogy a módszer elég nagy valószínűséggel meg tudja mondani egy autóról, hogy része-e a felismerni kívánt halmaznak, úgy, hogy közben

kevés valóban felismerni kívánt kocsira kerül rá ugyanaz a címke. Látható, hogy az eredmények egyértelműen függnak a tanító halmaz elemszámától. Minél nagyobb elemű a tanító halmaz, összességében annál több sikeres címkézés érhető el.

3.3. A Suzuki Swifték szétválasztása, az évjárat megállapítása

A Swifték gyűjtésekor 2 különböző évjáratú modelltől gyűjtöttem képeket, amelyek az eddigi tesztek során nem lettek megkülönböztetve. Annyira nincs nagy hasonlóság a 2 modell között, de érdemes megvizsgálni, hogy az osztályozó különbséget tud-e tenni közöttük, így az osztályozásnak egy mélyebb szintjére érve, ahol nem csak a gyártót és a típust ismeri fel az algoritmus, de az évjáratot is. Így ennek a tesztnek az erejéig 11 osztályosnak is felfogható a probléma, mert a Swift osztály 2 részre esik szét.



3.4. ábra. A 2 Suzuki modell, amelyek évjáratban térnek el

A 100 Swiftből 46 a baloldali ábrán szereplő évjárat (Swift1) és 54 a baloldali ábrán szereplő évjárat (Swift2). A 2 évjáratot 100%-osan sikerült megkülönböztetni egymástól, ami azt jelenti, hogy +1 osztály felvétele semmilyen problémát sem okozott, és a hasonló modelleket is jól kezeli a módszer, így képes lehet több osztály esetén évjárat szerinti felismerésre is.

Ebből a fejezetből kiderült, hogy a módszer robusztus. A perturbációs tesztek megmutatták, hogy a gyakorlatban is alkalmazható ez a fajta osztályozó módszer, mert még a viszonylag rossz szegmentálást modellező tesztek során is elfogadható eredmények születtek. Az outlier kocsik detektálása is elképzelhető, amely a tanító halmaz méretének növelésével robusztusabbá tehető. Továbbá elmondható a módszerről, hogy alkalmas lehet az osztályozást egy mélyebb szintre vinni, ahol az aktuális kocsik évjáratát is megmondjuk. Ezek alapján érdemes a módszer automatizálásán gondolkodni.

4. Félautomata gépkocsi felismerés

A manuális módszer vizsgálata után, ebben a fejezetben bemutatok egy gyakorlatban is alkalmazható módosítást a módszernek, ahol a sarokpontokat nem egy szöveges fájlból olvassuk ki, hanem a rendszámtábla félautomata szegmentálásával kapjuk meg. A félautomata arra utal, hogy szükség van felhasználói beavatkozásra. Feltételezzük, hogy ismert egy pontja (seed point) a rendszámtáblának, amit például a felhasználó ad meg okostelefonjának érintőképernyője segítségével, vagy asztali gép esetén az egérrel. A tesztek során minden kocsira előre lementettem egy véletlenszerű seed pontot, amely nem a rendszámtábla betűire/számaira esik. Az automata módszer esetében látni fogjuk, hogy ez a feltevés életszerű, hiszen elérhető, hogy a seed ne betűre/számra essen.

4.1. A rendszámtábla seed pont alapú szegmentálása

A szegmentálás egy régiónövelő eljárás alapul [6], amely eredetileg 2 és 3 dimenziós régiónövelő eljárás volt, de átalakítottam 2D-re a futási idő javítása érdekében. A módszer lényege, hogy adott pontból indulva próbálunk létrehozni egy régiót, egészen addig, amíg van az aktuális régió pontjainak olyan szomszédja, amelyre a következő feltételek teljesülnek:

- Az intenzitása egy adott intervallumba esik.
- A seed ponttól mért távolsága nem nagyobb egy küszöbnél.

Az intenzitás intervallumot természetesen célszerű az aktuális seed pont intenzitása alapján meghatározni, vagy annak egy adott környezetéből. A seed intenzitásának függvényében finom hangolható a szegmentálás nagyon sötét megvilágítás kezelésére is úgy, hogy ezekben az esetekben csak szűk intenzitástartományt engedünk meg. A seedtől mért távolságküszöböt a kép szélessége függvényében adtam meg úgy, hogy a kép szélességének harmadánál messzebb lévő pont nem kerülhet be a régióba.

A régió határán belül keletkező lyukakat az algoritmus automatikusan feltölti az *imfill* függvény segítségével (4.1. ábra). Az ábrán látható eset ideális, hiszen a rendszámtábla körvonalak jól adóttak. Gyakran előfordul, hogy a régió kifolyik a rendszámtáblán lévő egyéb feliratok, a gyenge megvilágítás vagy a koszos rendszámtábla miatt vagy darabokra esik szét. Ezeket az eseteket megfelelően alkalmazott morfológiai műveletek segítségével korrigálhatjuk (4.2. ábra).

JVK- 46



4.1. ábra. A régió belül keletkező lyukak kitöltése (*imfill*). Anonimitás miatt van kitakarva egy szám.

TF-7 6

7



4.2. ábra. Az első képen a régiónövelés eredménye látható. A másodikon a lyukfeltöltés eredménye, végül az utófeldolgozás eredménye és annak visszavetítése az eredeti képre. A kitakart karakterek anonimitási célt szolgálnak.

A régiónöveléshez nem a szürkeárnyaltos képet használom, hanem az RGB kép B csatornáját, azaz a kék komponenst. Ennek az az oka, hogy az EU-s rendszámtáblák bal oldala kék, ami viszonylag magas intenzitással jelenik meg a kék csatornán, így az esetek döntő többségében nem okoz szegmentálási hibát (4.3. ábra). Arra az esetre, ha nem sikerülne a rendszámtábla szegmenshez hozzávenni a kék EU-s részt, minden modellhez felvettem a tanító halmazba egy speciális tanító elemet, amelynél a kék szegmens nélküli rendszámtábla számít a tökéletes szegmentálásnak, így kezelve az osztályozásnál ezt a nagy valószínűséggel bekövetkező szegmentálási hibát (4.4. ábra).



4.3. ábra. A kék komponens előnye az EU-s rendszámtábláknál.

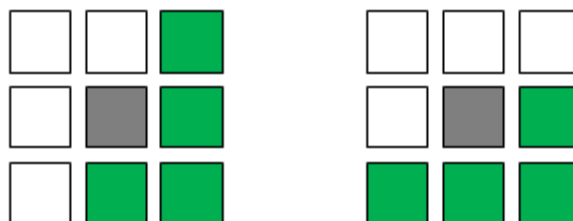


4.4. ábra. Rosszul szegmentált, de jól osztályozott EU-s rendszámtáblás kocsi. A kitakart karakterek anonimitási célt szolgálnak.

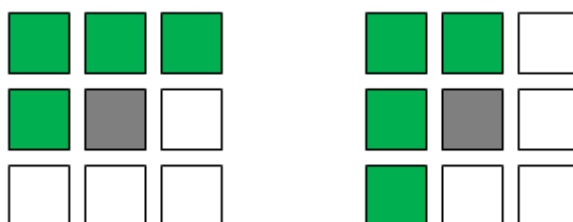
4.2. A sarokpontok kinyerése

Ebben az alfejezetben azt mutatom meg, hogy egy szegmentált rendszámtábla sarokpontjait milyen módszerrel célszerű kinyerni. Csak a kontúrponatok érdekesek, így minden más pontot eldobunk. A sarokpont probléma még tökéletes szegmentálás esetén sem oldható meg egy egyszerű sarokdetektálással, hiszen kicsit elfordult rendszámtábla esetén is számos pont sarokpont címkét kapna, így ez nem járható út. A legrobosztusabb megoldásnak az ígérkezik, ha vesszük a szegmens kontúrját és a szomszédságokat vizsgálva 4 halmazba szeparáljuk őket, amelyek a rendszámtábla megfelelő oldalpontjai lesznek és ezekre a halmazokra illesztünk 1-1 egyenest úgy, hogy minimalizáljuk a pontok egyenestől mért távolságának négyzetösszegét, tehát a legkisebb négyzetek módszere szerint optimalizálunk. Végül az egyenesek metszéspontjai lesznek a rendszámtábla sarokpontjai.

A kontúr első pontját a JPEG kódolásnál megismert Zig-Zag séma alapján keresem, vagyis nem egyszerűen sorfolytonosan vagy oszlopfolytonosan. Első körben a bal felső sarokból indítok el egy keresést, amivel megkapok a rendszámtábla bal felső sarkának közeléből egy pontot. Ebből a pontból indulva 2 szomszédsági maszkkal indítok el keresést (4.5. ábra), amelyek felső és bal oldali egyenes ponthalmazait gyűjtik be. Ezt követően a jobb alsó sarokból indítok egy Zig-Zag pásztázást, majd az így megtalált indulópontból a 4.6. ábrán szemléltetett szomszédsági maszkokkal begyűjtöm a rendszámtábla jobb oldalegyenesének és alsó oldalegyenesének ponthalmazait. Természetesen más szomszédsági konfigurációk is elképzelhetők. Adódhatnak olyan esetek is, hogy ezek a maszkok nem jó oldalhoz tartó pontokat is begyűjtenek, de ezeket a hibákat a legkisebb négyzetek módszere szerinti illesztés enyhítheti. A maszkok megalkotásánál a fő cél a szegmentálási hibák kiszűrése volt.



4.5. ábra. A felső és bal oldali egyenes pontjait megállapító szomszédsági maszkok. Ha az aktuális pont (szürke mező) bármely zölddel jelölt szomszédja nem nulla, akkor bekerül az aktuális halmazba.

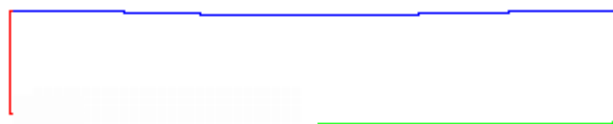


4.6. ábra. A jobb oldali és alsó egyenes pontjait megállapító szomszédsági maszkok. Ha az aktuális pont (szürke mező) bármely zölddel jelölt szomszédja nem nulla, akkor bekerül az aktuális halmazba.

Azért Zig-Zag séma szerint járom be a képet, mert a sarkok többnyire jól szegmentálhatóak, de az alsó és/vagy felső oldalak mentén túlszegmentált lehet a rendszámtábla, amit a kereten lévő feliratok okoznak (4.7. ábra). Továbbá feltételezhető, hogy az autó nincs $\mp 45^\circ$ -nál nincs jobban elfordulva a képen. Természetesen az ábrán szereplő végeredményre illesztett egyeneset negítívan befolyásolja a túlszegmentálás, ha a pontjai belekerülnének az alsó egyenes ponthalmazába. A Zig-Zag bejárással a 4.7. ábrán szereplő probléma remekül kezelhető és ugyanez igaz a felső oldali kifolyások esetén (4.8. ábra).



4.7. ábra. Az alsó kifolyás szemléltetése. A kitakart karakterek anonimitási célt szolgálnak.



4.8. ábra. A Zig-Zag bejárás segítségével megkapott oldalegyenes pontok halmazok

A 4.8. ábrán nagyon szépen látszik, hogy a problémás alsó oldal kontúrpointjainak csak közel fele került bele az illesztő halmazba, de amik bekerültek azok kevés kivétellel valóban az oldalegyenes pontjai, így nagyon kis hibával illeszthető rá az

egyenes. Valójában nincs is szükség az összes kiszegmentált oldalegyenes pontra, a töredéke is elég egy megfelelő pontosságú egyenes illesztésére.

Az egyeneseket SVD felbontás segítségével illeszttem a ponthalmazokra [8]. Jelölje X és Y az adott ponthalmaz x és y koordinátáit. Mindkét vektor hossza megegyezik a pontok számával. Első lépésként kiszámítom a ponthalmaz súlypontját, ami legyen az (x_0, y_0) pont. Létrehozok egy olyan $n \times 2$ méretű A mátrixot, melynek első oszlopában $X - x_0$ vektor szerepel és második oszlopában az $Y - y_0$. Ennek a mátrixnak veszem az $A = UDV^T$ felbontását. Matlabban ez a következő módon tehető meg:

$$[U \ D \ V] = svd(A, 0),$$

ahol a második paraméter arra utal, hogy csak az első 2 oszlopát szeretnénk kiszámítani az U mátrixnak. Ha adott a felbontás, akkor már ismerjük is a legkisebb négyzetes értelemben vett, a ponthalmazra illeszkedő egyenest, hiszen át fog menni a súlyponton, ami az (x_0, y_0) pont és az irányvektora a V mátrix első oszlopa lesz. Ez a megoldása a problémának azért különösen kedvező, mert kezelhetővé válnak a gyakran végtelen meredekségű jobb- és bal oldalegyenesek. Az egyenesek ismeretében a metszéspontjuk egy egyszerű lineáris egyenletrendszer megoldásával megkapható [5].



4.9. ábra. A ponthalmazokra illesztett egyenesek és azok metszéspontjai. Kék színnel a szegmentált kontúr látható. A kitakart karakterek anonimitási célt szolgálnak.

Most, hogy adottak a rendszámtábla sarokpontok, így már tesztelhető a félautomata módszer, hiszen a felismerő eljárás csak a sarokpontok ismeretére támaszkodik. Ebben az esetben 3 tesztet hajtottam végre:

- a 3-4 osztályonkénti tanítóképet használt (42 elemű),
- az osztályonként 20-20 elemű tanító halmazút (200 elemű), valamint

- a leave-one-out keresztvalidációt.

Az első esetben 93,6%-os találati arányt sikerült elérni a 42 elemű tanító halmazzal, ami alig 6%-kal gyengébb, mint az előre adott ideális sarokpontok esete (4.10. táblázat). A 20-20 elemű tanító halmaz nem hozott számottevő változást, mindössze 3 kocsival hibázott el többet az osztályozó, így 93,1%-ot ért el (4.11. táblázat). Végül a keresztvalidációs teszt további 2 autót osztályozott sikertelenül, ezzel 93%-os eredményt elérve (4.12. táblázat).

Az eredményekből leolvasható, hogy a hibát itt nem az autófelismerő módszer követi el, hanem a szegmentálási eljárás ad vissza olyan sarokpontokat, amelyek által kijelölt ROI messze eltér az ideális ROI-tól, így nem lehetséges az osztályozás. Az eredmények kiválóan tükrözik a perturbációs tesztekben levont következtetések helyességét, hiszen remekül tolerálja a módszer a kijelölési pontatlanságokat. Több osztály bevezetése sem képes jobb eredményt elérni, sőt néha véletlenszerűen más osztályba kerülnek át a rosszul szegmentált rendszám táblához tartozó modellek. A félautomata módszer eredményei nagyon biztatóak, hiszen ez már egy olyan módszer, amely gyakorlatban is alkalmazható. A következőben bemutatok egy lehetséges módszert a felismerés teljes automatizálására.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	44	3	93,6%
Astra	100	94	6	94%
Fabia	100	88	12	88%
Focus	43	37	6	86%
Golf	93	90	3	96,8%
Megane	45	44	1	97,8%
Octavia	47	44	3	93,6%
Polo	97	85	12	87,6%
Swift	100	99	1	99%
Összegzés	727	680	47	93,5%

4.10. táblázat. Eredmények: SMG[34x80]_correlation. 42 elemű tanító halmaz félautomata tesztje.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	45	2	95,7%
Astra	100	94	6	94%
Fabia	100	89	11	89%
Focus	43	36	7	83,7%
Golf	93	90	3	96,8%
Megane	45	44	1	97,8%
Octavia	47	44	3	93,6%
Polo	97	85	12	87,6%
Swift	100	95	1	95%
Összegzés	727	677	50	93,1%

4.11. táblázat. Eredmények: SMG[34x80]_correlation. 200 elemű tanító halmaz félautomata tesztje.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	55	0	100%
307	47	45	2	95,7%
Astra	100	95	5	95%
Fabia	100	90	10	90%
Focus	43	36	7	83,7%
Golf	93	90	3	96,8%
Megane	45	44	1	97,8%
Octavia	47	44	3	93,6%
Polo	97	83	14	85,6%
Swift	100	93	7	93%
Összegzés	727	675	52	93%

4.12. táblázat. Eredmények: SMG[34x80]_correlation. Félautomata leave-one-out cross-validation teszt.

5. Teljesen automatikus gépkocsi felismerés

Az előző fejezetekben kidolgozásra került egy olyan módszer, amely a rendszám tábla sarokpontok ismeretében képes egy kocsiról megmondani, hogy ki gyártotta, melyik modell, sőt akár az évjáratot is. Ezt követően bemutattam egy eljárást, amely segítségével kinyerhetőek a rendszám tábla sarokpontjai, ha ismerek egy pontot róla. Ebben a fejezetben azt fogom tárgyalni, hogy miként lehetséges egy rendszám tábla pontot keresni a képen, teljesen automatikusan.

5.1. A ROI kijelölése

Mivel a képek egy jól meghatározott protokoll szerint készülnek, így felhasználhatjuk ezt az információt, hogy a lényegtelen területek egy részét eldobjuk. Ha $h \times w$ a kép felbontása, akkor a lényeges részeket a következő képpontok alkotják:

$$\left\{ (x, y) \mid x \in \left[\frac{h}{3}, h \right] \text{ és } y \in \left[\frac{w}{4}, \frac{3w}{4} \right] \right\}$$

Az 5.1. ábra szemlélteti a ROI kijelölését. Az algoritmus az eredeti színes kép szürkeárnyaltos változatán dolgozik, semmilyen színinformációra nincs szükség. Végül elsimítom a ROI-t egy 9-es kernelméretű medián szűréssel.



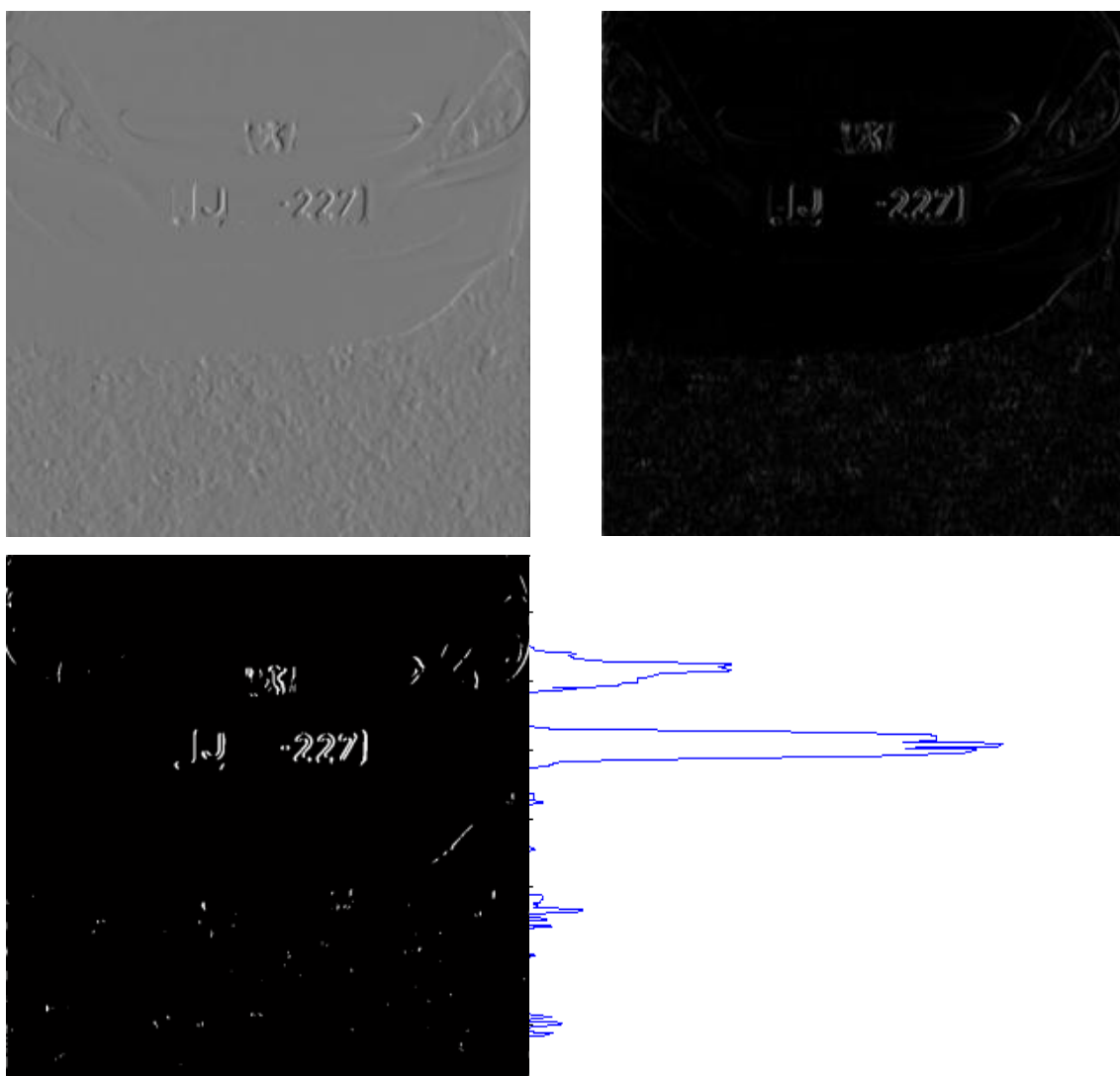
5.1. ábra. A rendszám tábla régió kijelölése. A kitakart karakterek anonimitási célt szolgálnak.

5.2. A rendszám tábla detektálása

Az alapötlet Ondrej Martinsky BSc dolgozatából származik, aminek lényege, hogy olyan terület lesz nagy valószínűséggel rendszám tábla, amelyen sűrűn helyezkednek el az élek [7]. Ebből kiindulva konstruáltam egy saját x-irányú gradiens maszkot, amellyel detektálom az x-irányú intenzitásváltozásokat:

$$M_x = \begin{pmatrix} 1 & 2 & 0 & -2 & -1 \\ 2 & 3 & 0 & -3 & -2 \\ 3 & 4 & 0 & -4 & -3 \\ 2 & 3 & 0 & -3 & -2 \\ 1 & 2 & 0 & -2 & -1 \end{pmatrix}$$

A konvolúció után kapott képnek vesszük az abszolút értékét, amelyet egy könnyen számítható x-irányú gradiens magnitúdó képként kezellek. Azt követően küszöbölem a magnitúdó képet egy 900-as küszöbvel és elkészítem a horizontális projekcióját (5.2. ábra).



5.2. ábra. A jobb felső képem az x-irányú gradiens látható, mellette pedig az abszolút értéke, azaz a magnitúdó. Alul a küszöbölt magnitúdó kép és annak horizontális vetülete látható. A kitakart karakterek anonimitási célt szolgálnak.

Miután ismert a horizontális projekció, azt találtam ki, hogy kiszámítom a maximum helyét és azt a sorát a képnek egy, a rendszám táblán átmenő pixelsornak tekintem. Ezzel a módszerrel a nagyméretű 2 dimenziós keresési teret egy kicsi 1 dimenzióssá

redukálom. Sőt, még ezt a szakaszt is tovább redukálom úgy, hogy csak a ROI második harmadában vizsgálom az egyenest, hiszen ott már szinte biztos, hogy metszenie kell a rendszámtáblát. A kérdés már csak az, hogy erről a rendszámtáblán átmenő egyenes szakaszcól, milyen stratégia alapján válasszak seed pontot.



5.3. ábra. A rendszámtáblán átmenő szakasz, amin a seed pontot keressük. A kitakart karakterek anonimitási célt szolgálnak.

5.3. A megfelelő seed pont kiválasztása

Az előzőekben láttuk, hogyan kell egy olyan egyenes szakaszt meghatározni a képen, amely nagy valószínűséggel áthalad a rendszámtábla nagy területén. Óvatosnak kell lenni, amikor seed pontot szeretnénk választani erről az egyenesről, hiszen nem akarunk betűről, számról vagy esetleg zárt és üreges karakter belső területéről (pl.: 6, 9, D) választani.

A seed pont választáshoz több pontot is megvizsgállok az egyenesről. A kezdőpont az egyenes középpontja, ahonnan végrehajtom a régiónövelést és elmentem a hozzá tartozó régió területét. Ha ez a terület kisebb, mint egy küszöbszám, akkor karaktert vagy karakter lyukat találtunk el és nem veszem figyelembe. A következő pontokat adott távolsággal történő jobb és bal irányú ugrásokkal veszem, tehát felváltva ugrálok jobbra és balra (5.4. ábra). Minden ponthoz kiszámítom a belőle növelhető régió területét úgy, hogy a kicsi területeket eldobom. Ha 4 régiónövelés közel azonos méretű területet eredményez, akkor az utoljára használt seed pontot elfogadom és végrehajtom vele a rendszámtábla szegmentálást, majd a felismerést. Ha végigértünk az egyenesen,

de nem volt 4 hasonló területű régió, akkor véletlenszerűen választhatunk vagy valamilyen heurisztika alapján, mint például a kép mérete alapján várható rendszámtábla terület.



5.4. ábra. Az egyenes bejárási sémája. A kitakart karakterek anonimitási célt szolgálnak.

Az automata módszeren is lefutattam a 4. fejezetben ismertetett teszteket, melyeket rendre az 5.5.-5.7. táblázatok foglalnak össze. Az eredmények az ideális tesztkörülményekhez mérten nagyon jónak mondhatóak. Ebben az esetben is ugyanaz mondható el, mint a félautomata esetben, tehát az eredményeket nem változtatja számottevően a különböző méretű tanító halmazok használata. Az eredményekből tehát látszik, hogy nem a felismerő eljárás okozza a hibákat, hanem a szegmentálás. A rendszámtábla detektálás és a szegmentálás javításával ezek az eredmények is messze 90% fölé vihetőek. A teljesen automata eset is igazolja a perturbációs teszteket, hiszen rendkívül robusztus a felismerő eljárás.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	54	1	98,2%
307	47	41	6	87,2%
Astra	100	88	12	88%
Fabia	100	82	18	82%
Focus	43	37	6	86%
Golf	93	85	8	91,4%
Megane	45	43	2	95,6%
Octavia	47	41	6	87,2%
Polo	97	80	17	82,5%
Swift	100	92	8	92%
Összegzés	727	643	84	88,4%

5.5. táblázat. Eredmények: SMG[34x80]_correlation. 42 elemű tanító halmaz automata tesztje.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	54	1	98,2%
307	47	42	5	89,4%
Astra	100	89	11	89%
Fabia	100	84	16	84%
Focus	43	37	6	86%
Golf	93	85	8	91,4%
Megane	45	43	2	95,6%
Octavia	47	41	6	87,2%
Polo	97	74	23	76,3%
Swift	100	93	7	93%
Összegzés	727	642	85	88,3%

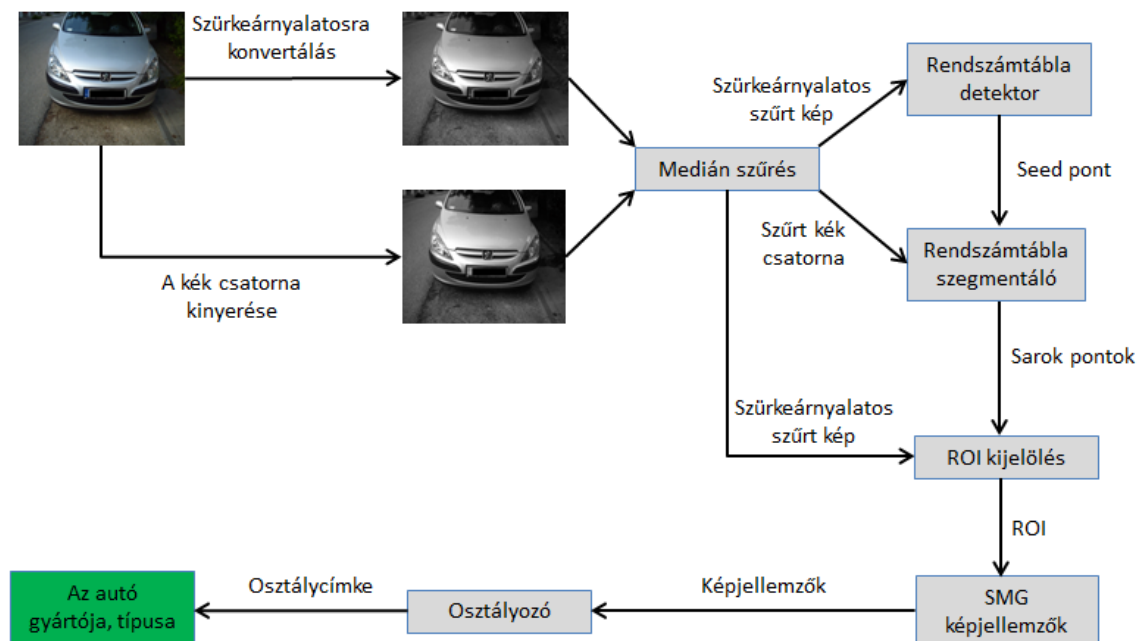
5.6. táblázat. Eredmények: SMG[34x80]_correlation. 200 elemű tanító halmaz automata tesztje.

Modell	Minták száma	Sikeres felismerés	Sikertelen felismerés	Találati arány
206	55	54	1	98,2%
307	47	41	6	87,2%
Astra	100	90	10	90%
Fabia	100	85	15	85%
Focus	43	37	6	86%
Golf	93	85	8	91,4%
Megane	45	43	2	95,6%
Octavia	47	41	6	87,2%
Polo	97	75	22	77,3%
Swift	100	93	7	93%
Összegzés	727	643	84	88,6%

5.7. táblázat. Eredmények: SMG[34x80]_correlation. Automata leave-one-out cross-validation teszt.

6. Az eredmények összefoglalása

A szakirodalomban leírt módszernek sikerült olyan módosításait létrehozni, amelyek egy valós gyakorlati alkalmazásnál előforduló fényképeken is képesek gépkocsi gyártó és típus azonosításra. Az eredeti módszerrel az volt a probléma, hogy a ROI kijelöléséhez szükséges sarokpontok pontos ismeretét igényelték és még ebben az ideális esetben is 50% alatt teljesített egy gyakorlatias adatbázison. Láttuk, hogy megfelelő módosításokkal a találati arány akár 100%-ra is emelhető 10 osztály esetén. A robusztusságot mérő tesztekben kiindulva elkészült a félautomata (~93%) és automata (~88%) változat, amelyek a vártak megfelelően, az ideális esetnél rosszabbul, de összességében alacsony hibaszázalékkal teljesítettek. A szegmentálási eljárás javításával az eredmények további javulása várható. Az algoritmus tehát skálázható olyan értelemben, hogy az aktuális igényeknek megfelelően eldönthető, hogy a futási idő vagy a nagyobb találati arány a fontos.



6.1. ábra. Az automata algoritmus fő lépései.

6.1. Gyakorlati alkalmazások

Egy lehetséges gyakorlati megvalósítás lehet például egy mobilalkalmazás, ahol a felhasználó mobiltelefonja segítségével lefényképez egy autót és elküldi azt a képet egy nagyobb számítási kapacitású kiszolgálónak, ami pár másodpercen belül visszaküldi a képen látható kocsi pontos típusát, esetleg az évjáratát is. Az autók gyártójának és típusának felismerése számos okból hasznos lehet:

- Gépkocsi demográfiai vizsgálatok a célzott hirdetések érdekében.
- Statisztikai elemzések.
- Rendőrségi munka könnyítése. Lopott autók azonosítása rendszámtábla adatokkal együtt.
- Parkolási csalások detektálása.
- Parkolóházak parkolóhelyeinek optimalizálása statisztikai adatok alapján.
- Piackutatás.

Látható volt, hogy az osztályozás során sehol sem használtuk ki, hogy gépkocsikat szeretnénk felismerni. Egyedül csak a ROI kijelölés volt feladat függő, hiszen a rendszámtábla alapján hoztunk létre a vizsgálandó területet. Ebből az is érezhető, hogy a tárgyalt módszer számos egyéb felismerési feladatra is alkalmas lehet, ahol valamilyen módon lehetőség van egy, a felismerni kívánt objektumokra jellemző ROI kijelölésére. Az algoritmus viszonylag könnyen implementálható lépésekből áll és megfelelő beállítások mellett, akár néhány tized másodperc alatt képes lehet eredményt szolgáltatni.

Irodalomjegyzék

- [1] V.S.Petrovic and T.F.Cootes: Analysis of Features for Rigid Structure Vehicle Type Recognition, University of Manchester, M13 9PT, UK, British Machine Vision Conference, 2004
- [2] Greg Pearce and Nick Pears: Automatic Make and Model Recognition from Frontal Images of Cars. 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance, 2011.
- [3] R.C. Gonzales, R.E. Woods: Digital Image Processing, 3rd edition, Prentice-Hall, Inc., 2008
- [4] J. R. Parker - Algorithms for Image Processing and Computer Vision, Wiley, 2011
- [5] Matlab Central Thread Subject: Linear Fit
http://www.mathworks.de/matlabcentral/newsreader/view_thread/170303
- [6] Daniel Kellner, Region growing implementation,
<http://www.mathworks.com/matlabcentral/fileexchange/32532-region-growing-2d3d-grayscale/content/regionGrowing.m>
- [7] Ondrej Martinsky - Algorithmic and Mathematical Principles of Automatic Nuber Plate Recognition Systems
<http://javaanpr.sourceforge.net/anpr.pdf>
- [8] Alfio Quarteroni, Riccardo Sacco, Fausto Saleri - Numerical Mathematics, Springer, 2007