



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

KÖNYVKATALÓGUS ÉPÍTÉS KÉPFELDOLGOZÁSI MÓDSZEREKKEL

Szerző: **Mihályi Martin**
mérnök informatikus BSc. szak, IV. évf.

Konzulensek: **Dr. Vámosy Zoltán**
egyetemi docens
Windisch Gergely
tanársegéd

Budapest, 2014.

Tartalomjegyzék

1	Absztrakt	4
2	Bevezetés	5
2.1	Probléma ismertetése.....	5
2.2	Megoldási javaslat.....	6
3	Irodalomkutatás.....	7
3.1	Hasonló rendszerek	7
3.1.1	Mobile Phone Book Finder	7
3.1.2	Building Book Inventories using Smartphones	9
3.1.3	Combining Image and Text Features: A Hybrid Approach to Mobile Book Spine Recognition	10
3.1.4	Hasonló rendszerek értékelése	11
3.2	Alkalmazható megoldások	12
3.2.1	Könyvdetektálás.....	12
3.2.2	Cím felismerés	15
4	A rendszer	20
4.1	Rendszervázlat.....	20
4.2	Könyvdetektálás	21
4.2.1	Átméretezés.....	23
4.2.2	Nyitás- Zárás	24
4.2.3	LSD – Line Segment Detection	25
4.3	Cím felismerés.....	27
4.3.1	Szöveg alapú cím felismerés.....	28
4.3.2	Szín alapú cím felismerés	34
4.3.3	Mintaillesztés alapú cím felismerés	37
4.4	Adatbázis keresés	37
5	Teszteredmények és értékelésük	39
5.1	Könyv elforgatás	39
5.1.1	Értékelés.....	40
5.2	Könyvdetektálás	40
5.2.1	Könyvdetektálás – Hough transzformáció	41
5.2.2	Könyvdetektálás – FSA	42
5.2.3	Összehasonlítás	42
5.3	Karakterfelismerés előfeldolgozás nélkül	43
5.3.1	Értékelés.....	44

5.4	Karakterfelismerés előfeldolgozással.....	44
5.4.1	Binarizáció	44
5.4.2	Utófeldolgozás	45
5.4.3	Karakterfelismerés modul	45
5.5	Színfelismerés	46
5.5.1	Értékelés.....	46
5.6	Teljes rendszer.....	46
5.6.1	Összehasonlítás	47
5.6.2	ABBYY.....	48
5.6.3	Értékelés.....	48
6	Összefoglalás	49
7	Ábrajegyzék	50
8	Irodalomjegyzék.....	52
9	Mellékletek.....	55
9.1	Publikációk.....	55
9.1.1	EME - Fiatal Műszakiak Tudományos Ülésszaka	56
9.1.4	CINTI 2014.....	60
9.2	Könyvgerinc detektálás	64
9.2.1	Bemeneti képek.....	65
9.2.2	Canny éldetektáló.....	67
9.2.3	LSD- Line Segment Detection.....	69
9.2.4	Canny éldetektáló és Hough Transzformáció	71
9.2.5	Canny éldetektáló és FSM	72
9.2.6	Watershed algoritmus	75
9.2.7	Canny éldetektáló, nyitás - zárás.....	77
9.2.8	Canny éldetektáló, nyitás - zárás, Hough Transzformáció	79
9.3	Részletes teszteredmények – Szegmentálás	81
9.4	Binarizáció.....	82
9.4.1	Mintaképek.....	82
9.4.2	Optimális eredmény	82
9.4.3	Otsu binarizáció	83
9.4.4	Niblack binarizáció	83
9.4.5	Bradley binarizáció	84
9.4.6	Sauvola binarizáció	84
9.5	Részletes teszteredmények – Binarizáció.....	86

1 Absztrakt

A könyvtárakban és könyvesboltokban gyakori probléma, hogy a rendelkezésre álló könyveket valamilyen módon kategorizálni kell. A kézi könyvkategorizálás egy nagyon időigényes és rengeteg hiba lehetőséget magánban hordozó feladat. A félautomata, vonalkóddal vagy rádió frekvenciás azonosítóval (RFID) ellátott rendszerek nagyon költségesek, és csak a nagyobb intézmények számára elérhetők. Ezen dolgozat célja egy képfeldolgozáson alapuló automatikus könyvkategorizáló rendszer, ami egy könyvespolcra készült fényképfelvételen ismeri fel a könyveket, és készíti el automatikusan a katalógust. E rendszer tervezésének és megvalósításának bemutatása, illetve eredményességének ismertetése a már működő hasonló rendszerek alapján.

Ezen dolgozat ismerteti a könyvgerincen történő szövegfelismerés egyes lépéseit, és az azokra alkalmazott technikákat. A dolgozat bemutatja milyen előfeldolgozási illetve utófeldolgozási módszerek szükségesek a legpontosabb karakterfelismerés érdekében. Illetve megvizsgálja milyen javulás érhető el e módszerek használatával.

A megvalósított eszköz mérésre és értékelésre kerül, mind különböző modulok hatásfokait illetőleg, mind a teljes rendszer hatásfokát illetőleg. Különös hangsúllyal az újonnan implementált vagy az előző cikk óta fejlődésen átesett modulok hatásfokát, mérési eredményeit mind saját mind más rendszereknél alkalmazott tesztadatbázisok alapján.

2 Bevezetés

Az informatika fejlődésével lehetőség nyílt arra, hogy olyan területeken használjuk munkák gyorsítására és megkönnyítésére, amelyeken eddig még csak manuális vagy félautomata eszközök álltak rendelkezésre, viszont ezen készülékeknek kellően optimálisak. Ilyen terület a könyvek kategorizálása is. E dolgozatban egy lehetséges megvalósítást ismertetek, illetve eredményeit értékelem.

2.1 Probléma ismertetése

Könyvtárakban vagy könyvesboltokban a könyvek fizikailag, polcokon valamiféle szisztéma szerinti rendezésre kerülnek. Könyvtárak esetén ez a rendezés történhet Cutter-szám vagy akár ETO-szám alapján. Könyvesboltok esetén marketing eszközök igénybevétele szerint történik a rendezés. Másrészről ezek az adatok tárolásra kerülnek úgynevezett katalógus cédulákon vagy adatbázisokban. E két viszonylag független tárolási rendszer összehasonlítása lehetővé teszi a könyvek leltározását. Mind könyvtárak esetén mind könyvesboltoknál egyértelműen belátható a leltározás szükségessége, és az, hogy ez a nagy könyvállomány miatt egy nagyon időigényes és optimalizációra szoruló feladat.

Manuális vagy kézi leltározási módszer alatt azt az eljárást értjük, amikor a könyveket egyenként le kell emelni a polcra, egyenként meg kell keresni benne a szükséges adatokat (cím, szerző, ISBN szám, kiadás éve, stb.), majd ezek begépelése után lehetséges a leltározás. Ez a megoldás nagyon hosszadalmas és monoton, valamint e két tényező miatt rendkívül magas a hiba lehetősége.

Félautomata rendszerről akkor beszélünk, ha valamilyen azonosítóval látjuk el a könyveket, majd adott eszköz segítségével leolvassuk, így időt takarítunk meg, mivel nem kell az adatokat begépelni, illetve ha a könyvek olyan részére helyezzük el az azonosítót, ahol anélkül hozzáférhető, hogy fizikai meg kellene mozdítani, ugyancsak sebesség növekedést érhetünk el.

Egyik ilyen félautomata leltározási módszer, ha a könyvek gerincét vonalkóddal látjuk el és egy vonalkód leolvasó segítségével leolvassuk azt. A módszer nagy hátránya, hogy végre kell hajtunk egy inicializáló lépést, amikor minden egyes könyvgerincére (azért a gerincére, mert az esetben leltározáskor nem kell a könyvet fizikailag elmozdítani a polcra) egy vonalkódot kell helyezni és a vonalkód adatait is hozzá kell adni a meglévő adatbázishoz. Ez az inicializáló lépés fizikailag károsítja a könyvet (ragasztás nyoma), így nem használható könyvesboltokban. Hátrány az is, hogy a megoldás minden esetben feltételezi egy vonalkód olvasó meglétét, ami egy kisebb költségvetésű könyvtár esetén problémát jelenthet.

Ugyancsak félautomata megoldás, ha rádió frekvenciás azonosítóval (RFID) látjuk el a könyveket, de sajnos ennek a megoldásnak is hasonló hátrányai vannak, mint a vonalkóddal való felcímkézésnek. Ez esetben a technológia még drágább, viszont így az azonosító helye nem megkötött.

Összességében elmondható, hogy az eddig ismertetett megoldások megfelelő eredményt szolgáltatnak, azonban félautomata, RFID vagy vonalkódos megoldás esetén szükség van egy elhúzódó inicializáló lépésre, valamint különleges eszköz iselengedhetetlen az eredményes működéshez. A hagyományos megoldás esetén viszont túlságosan nagy az időigény és a hiba lehetőség. Olyan problémamegoldásra lenne szükség, amely egyesíti a félautomata megoldás előnyeit, a teljesen manuális módszerével, azaz egy gyors, viszonylag hibamentes eredményt szolgáltató rendszer, bárminemű inicializáló lépés vagy különleges eszköz igénye nélkül.

2.2 Megoldási javaslat

A problémára megfelelő megoldást adhat egy képfeldolgozáson alapuló automatikus könyvkategorizáló rendszer. Ez a rendszer egy vagy több a könyvespolcra készült kép alapján végzi el az adatbázissal való összehasonlítást.

A könyvek a polcon kerülnek lefényképezésre, a szükséges információt a gerincükről nyerjük ki, így nem szükséges azok fizikai mozgatása, kinyitása, emiatt belátható, hogy a rendszer segítségével egy nagyságrendekkel gyorsabb leltározást valósíthatunk meg.

Napjainkban mindenki számára elérhető valamilyen képalkotó modalitás, legyen szó fényképezőgépről, webkamerával ellátott számítógépről vagy akár mobiltelefonról. Ezért nem jelent külön eszközigényt ez a megoldás, hisz minden felhasználó számára adott a szükséges konstrukció, amellyel fényképeket készíthet a könyvespolcon elhelyezett könyvekről.

Mivel az azonosítás a gerincen lévő nyomtatott szöveg alapján történik, nincs szükség semmilyen, a félautomata rendszerekre jellemző, azonosítóra. Ezáltal nem keletkezik semmi fizikai kár a könyvekben, így akár könyvesboltok esetében is alkalmazható ez a megoldás. Illetve nem növekednek a költségek a rendszer bevezetése által (nincs szükség azonosító berendezés beszerzésére), ezen okból kifolyólag kisebb költségvetésű könyvtárak számára is megfelelő megoldás lehet.

Az automatikus rendszer előnye az is, hogy akár saját otthoni könyveink kategorizálására is felhasználhatjuk, mivel a megoldáshoz szükséges eszköz biztosított és az adatbázist, amivel összehasonlítunk, maga az internet szolgáltatja.

3 Irodalomkutatás

Az irodalomkutatást két nagy részre bontom, először a már meglévő hasonló rendszereket mutatom be, majd kitérek a működésükre, felépítésükre, illetve értékelem a megoldásokat, ezt a [2, 3, 4] irodalmak alapján teszem. A második részben bemutatásra kerül, hogy egy adott modul szintjén milyen lehetséges megvalósítások léteznek. Ez az [5, 6, 9, 11, 13] irodalmak alapján valósul meg.

3.1 Hasonló rendszerek

3.1.1 Mobile Phone Book Finder¹

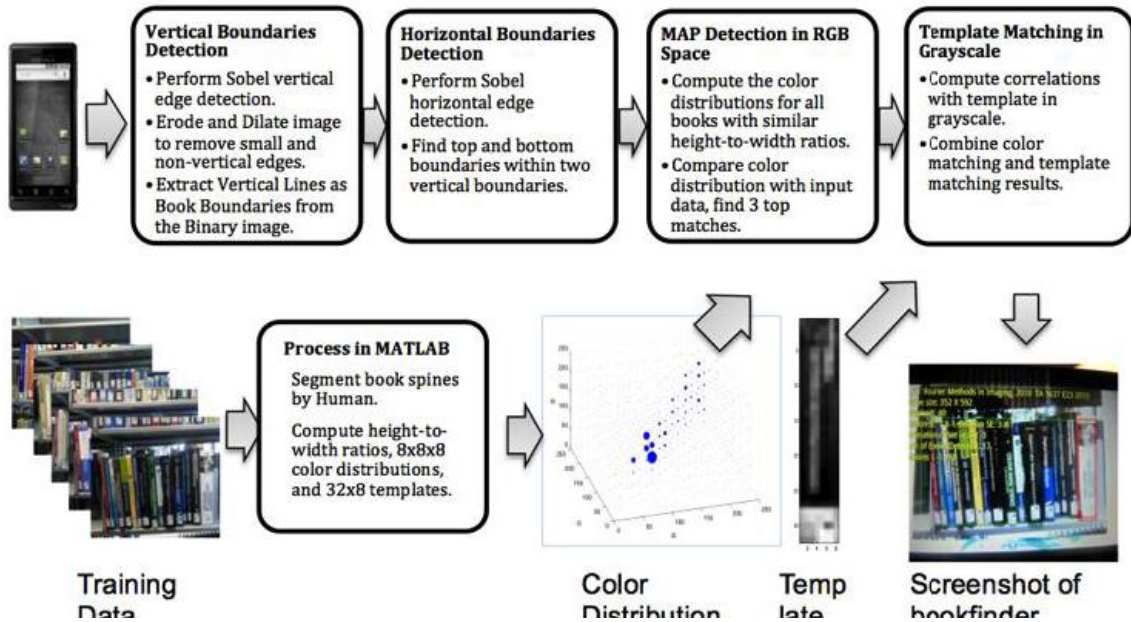
Az első bemutatni kívánt rendszer egy mobiltelefon alapú könyv felismerő alkalmazás, fejlesztésére 2012-ben került sor Y. Liao, J. Lu, M. Wu által a Stanford egyetemen[2]. Ezen alkalmazás célja, hogy a könyvtárban, vagy bármilyen olyan helységben, ahol valamilyen rendszer szerint rendezve állnak rendelkezésre a könyvek a polcokon, gyorsabban megtaláljunk egy adott könyvet.

A felhasználónak meg kell adnia a könyv címét, vagy könyvtári azonosítóját, majd a telefonjának kameráját a könyvespolcra irányítani. A rendszer felismeri a polcon lévő könyveket, és ha egyezést talál, azt a könyvet pirossal bekeretezve jelzi a képernyőn.

Ez a rendszer öt egységre bontható, függőleges élek detektálása, vízszintes élek detektálása, RGB színtéren maximum utólagos valószínűségű (MAP²) detektálás, majd sablonillesztés szürkeárnyaltos képen. A részletes rendszer vázlat az 1. ábrán látható.

¹ Mobiltelefon alapú könyv megtaláló alkalmazás

² MAP: maximuma posterioriprobability



1. ábra: Mobile Phone Book Finder rendszervázlata [2]

A rendszer vízszintes/ függőleges Sobel éldetektálást [6] használ a könyvek detektálására, ezután az RGB színtéren kiválasztja a három legvalószínűbb találatot, majd ezeken sablonillesztést végez, és ha megfelelő arányú az egyezés, kijelzi a találatot a képernyőre. A könyvgerincekről készült képekkel előzetesen rendelkezik egy adatbázis, ez az adatbázis az alkalmazással együtt a mobiltelefonon kerül eltárolásra.

A [2] irodalomból kiderül, hogy a rendszer hatásfoka 86%-os, azonban csupán 68 könyvből állt a tesztalmaz. 65%-ban erős detektálásról beszél, 21%-ban gyenge detektálásról, és 14%-ban nem sikerült detektálni a kívánt könyvet. Erős detektálás alatt azt érti a szerző, hogy a felhasználó által megadott könyvet sikerült pontosan detektálni és kiválasztani a képen, gyenge detektálás esetén a detektálás nem volt teljesen tiszta, néhány élt nem sikerült az algoritmusnak detektálni, de végeredményben a megfelelő könyvet ismerte fel a rendszer. Sikertelen detektálás esetén vagy nem sikerült felismernie a rendszernek a könyvet a képen, vagy hibásan ismert fel.

3.1.1.1Értékelés

Bár a 86%-os hatásfok jónak számít, jó pár hibával bír a rendszer. A Sobel éldetektáló [6] algoritmusnál már léteznek jobb hatásfokú megoldások is, elképzelhető, hogy egy Cannyéldetektáló [5] használatával sikerült volna csökkenteni a 14%-os meg nem találási arányt.

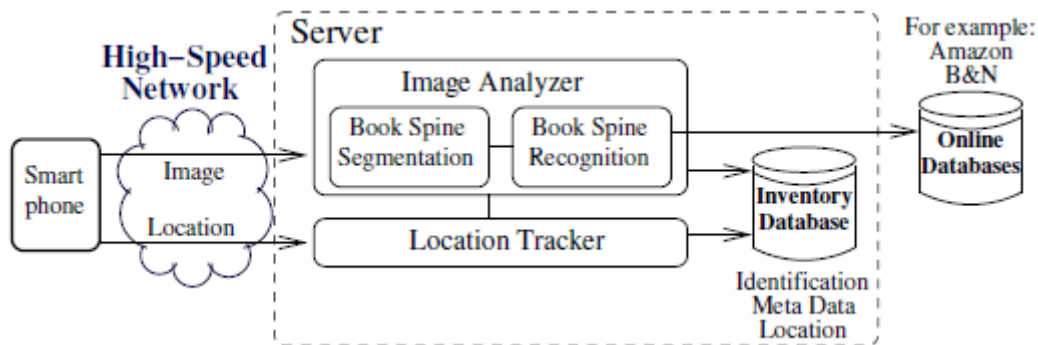
Illetve ez a megoldás az RFID és vonalkód alapú megoldáshoz hasonlóan inicializáló lépést igényel. A sablonillesztés miatt szükséges az a lépés, hogy minden egyes könyvgerincről készült képet egyenként felvigyünk az adatbázisban, mivel többek között célunk e lépés kiküszöbölése, ezért ez a megoldás sem elfogadható a számunkra.

Problémás lehet, hogy a könyvgerincekről készült képeket tartalmazó adatbázis az alkalmazással együtt a telefonon tárolódik, sok könyv esetén a mérete okozhat problémát, illetve az adatbázis frissen tartása is problémás, a felhasználónak időnként ellenőriznie kell, nem érhető-e el újabb adatbázis.

3.1.2 Building Book Inventories using Smartphones³

A következő bemutatni kívánt rendszer ugyancsak egy mobiltelefon alapú megvalósítás. D. Chen, S. Tsai, B. Girod, C-H. Hsu, K-H. Kim, J. P. Singh által fejlesztett rendszert 2010-ben mutatták be az ACM Conference of Multimedia (MM) konferencián [3]. Célja, hogy a könyvespolcra készített kép alapján olyan információkat is megjelenítsen a mobil kijelzőjén, ami magáról a könyvgerincéről nem meghatározható.

Ez a rendszer is több egységre bontható, rendelkezik egy könyvgerinc szegmentáló modullal, egy könyvgerinc felismerő modullal, és egyhelyzetkövetővel. A részletes rendszer a 2. ábrán látható.



2. ábra: Building Book Inventories using Smartphones rendszervázlata [3]

Működés közben nem csak a könyvespolcra készült képet veszi figyelembe, hanem a telefon helyzetét is. A képpel együtt a GPS koordinátákat is felhasználja a rendszer a detektálásra. A képen a könyvgerinc szegmentáló modul Canny éldetektáló[5] segítségével felismeri az éleket, majd Hough transzformációt[7] végez, a transzformáció eredményeként megkapjuk a szegmentált gerinceket. Ez a rendszer is feltételezi, hogy egy adatbázisban már rendelkezésre állnak a gerincekről a képek, amelyekkel össze tudjuk vetni a telefonról érkező képet. SURF [8] algoritmus alapján végez összehasonlítást, majd a telefon helyzetét figyelembe véve meghatározza a legjobban egyező találatot. Végül ez a rendszer a szerverről származó adatok alapján (könyvcím) keresést végez az interneten és az információkat megjeleníti.

A tesztelés során egy 2148 könyvgerincet tartalmazó adatbázist készítettek, majd 407 mintával tesztelték a hatásfokot. Sikeresnek számított a teszt, ha a megfelelő címet írta ki a könyvgerinc mellé a rendszer. A szerzők tesztelték a rendszert szegmentáció nélkül, azaz egyből SURF algoritmust hajtottak végre a képen, ahol az összes könyv egyben látszódott. Ilyen esetben a 407-ből 32 esetben sikerült a megfelelő címet párosítani. Ez 7,9%-os

³Könyv adatbázis építése okos telefon segítségével

pontosság. Ha a mintaillesztő algoritmust megelőzte egy szegmentáció, azaz könyvenként szétbontottuk a képet, 407 esetből 303 esetben volt sikeres a rendszer. Ez 74,5%-os hatásfokot jelent.

3.1.2.1Értékelés

Sajnos ez a rendszer is mintaillesztést használ, ezért ugyancsak könyvkép adatbázis nélkül használhatatlan. Jó elképzelésnek tűnik, hogy a GPS koordináták, egy másodlagos azonosítás céljából, eltárolásra kerülnek az adatbázisban, azonban ezek az adatok kezdetben nem állnak rendelkezésre. Illetve a jelenleg mobiltelefonon elérhető GPS beltérben 100 méteres pontosságú, ezért viszonylag nagy hibákat kell kezelnie a rendszernek, ami a pontosság csökkenését idézheti elő.

Ugyanakkor pozitívum, hogy nem a mobilon történik a feldolgozás, hanem egy szervert használ a megoldás. Egyértelműen belátható, hogy egy szerver teljesítménye nagyságrendekkel nagyobb egy telefonénál, ezért a feldolgozásra alkalmasabb. Továbbá az is pozitívum, hogy felhasználja az interneten elérhető forrásokat is a könyvről való adatgyűjtésre.

3.1.3 Combining Image and Text Features: A Hybrid Approach to Mobile Book Spine Recognition⁴

Ezen megvalósítás [4] fő célja egy olyan mobil alkalmazás fejlesztése volt, amely minél jobb hatásfokkal végzi el a könyvek azonosítását. 2011. novemberében publikálták az ACM Conference of Multimedia (MM) konferencián. Ez esetben is a rendszer a mobiltelefon segítségével készített képről ismeri fel a címeket, majd megjelenít hozzá további információkat az adatbázisból.

A rendszer két nagy egységből épül fel, a könyvgerinc kinyerő modul és a hibrid könyv felismerő modul.

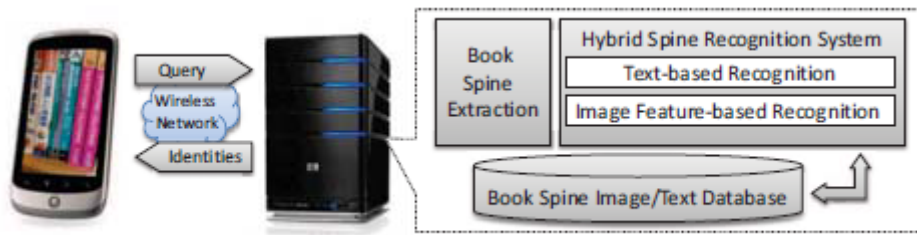
A könyvespolcra mobillal készült képet rögtön a szervernek küldi az alkalmazás és a szerveren történik a kiértékelés is. Ezen rendszer esetében is először a képen a könyvgerincek szegmentálása hajtódik végre. A gerincek alapján a felismerést egy hibrid rendszer végzi, a jobb hatásfok érdekében, mind mintaillesztés alapú megoldást, mind szöveg felismerést elvégez a rendszer. A részletes rendszerterv a 3. ábrán látható.

Sajnos nem derül ki a cikkből, hogy milyen módszert használtak a szegmentálásra. A mintaillesztés alapú felismerésre ez a rendszer is a SURF[8] algoritmust használja. A szöveg felismerésre MSER[10] és SWT alapú algoritmust használ, majd ezután OCR⁵ megoldással detektálja a betűket, a detektált betűket még összevetik egy értelmes szavakat tartalmazó

⁴Egy hibrid megoldás könyvgerincek felismerésére

⁵OCR: Optical Character Recognition

szótárral. Az összevetéshez a Nearest Neighbor(NN)[11] módszert használja. Ezen eredmények alapján történik az adatbázisban való keresés.



3. ábra: A hibrid megoldás rendszervázlata [4]

A hibrid rendszer 91%-os pontosságot ért el. A mérésekhez egy 2300 könyvgerinc képet tartalmazó adatbázist használtak, és egy 5398 szót tartalmazó szótárt. Külön mérték a szöveg alapú, képillesztés alapú és a hibrid megközelítést. A rendszer 91%-os pontosságot ért el az esetben, ha legalább a szöveg alapú vagy képillesztéses módszer sikeresen detektált. Legalább az egyik módszer által felismert gerincek 58%-át ismerte fel sikeresen mindkét módszer. Ebből következtethetünk arra, hogy viszonylag eltérő gerinceken hatásosak a módszerek. Csak képillesztéses megoldás esetén 72%-os pontosságot ért el a rendszer, csak szöveg alapú esetében rosszabb volt a határfok, mint 40%.

3.1.3.1Értékelés

Ezen megoldás is feltételezi egy könyvgerinceket tartalmazó adatbázis meglétét, e nélkül rendkívül gyenge határfokú. Ugyanakkor ez a hibrid megközelítés alkalmazható lehet, egyrészt magas határfoka miatt, másrészt a megoldás működőképes képerinc adatbázis nélkül is. Sajnos ebbe a megvalósításba nem került be semmiféle tanuló folyamat, azaz, ha egy könyvet csak szöveg alapú módszerrel sikerült azonosítani, akkor annak gerincéről készült képet a rendszer nem veszi fel utólagosan az adatbázisába, ez nagyon sokat ront a használhatóságán.

Problémás lehet az is, hogy a szerver a hozzá érkező képeken és az arra küldött válaszokon kívül nem kommunikál a külvilággal. Ha egy könyvet nem sikerült az adatbázisban levő egyik címmel sem párosítani, akkor a felismert szöveg alapján kereshetne a rendszer a világhálón könyv információk után (különböző könyvtárak, könyvesbolt láncok weboldalán).

3.1.4 Hasonló rendszerek értékelése

Összességében elmondható, hogy egyik bemutatott rendszer se nyújt a megfogalmazott céloknak megfelelő megoldást. Az összes bemutatott rendszer főként vagy teljesen arra támaszkodik, hogy rendelkezünk egy könyvgerinc képeket tartalmazó adatbázissal, amely hiánya nélkül nagyon alacsony határfokkal vagy abszolút nem működőképesek. Ez a megközelítés elfogadhatatlan, mivel így ugyanúgy szükség van egy inicializáló lépésre (adatbázis elkészítése), mint a félautomata (RFID, vonalkód) megoldások esetében.

Továbbá ezen ismertetett rendszerek nem engednek meg túl nagy szabadságot a könyvgerincek detektálását illetően. Az [2] rendszer teljesen függőleges éleket feltételez,

illetve a többi is maximum 15 fokos dőlést enged meg. Sok esetben felmerülhet az igény, hogy sokkal szabadabb elrendezés esetén (akár nem a gerincről, hanem a borítóról vagy hátlapról leolvasva is detektáljon) működőképes legyen a rendszer.

Bizonyos esetekben a rendszer „zárttsága” is problémát okozhat, azaz nem használ a felismert címek alapján történő kereséshez külső, internetes forrásokat. Így a bemutatott rendszerek kellően pontos működéséhez nagyméretű adatbázis építése szükséges, és még akkor is előfordulhatnak olyan esetek, amikor abból származóan eredménytelen a rendszer, hogy az adatbázis nem tartalmazza a keresett könyv információkat. Ezen hibák megoldása lenne, ha a rendszer nem csak a belső adatbázisában, hanem információk után az interneten is kutatna.

3.2 Alkalmazható megoldások

A hasonló projektek alapján a feladatot három részre érdemes bontani: könyvdetektálás, cím felismerése, adatbázis keresés (e felosztás indoklását lásd az 4.1 fejezetben). Ezen fejezetben egy-egy részegység lehetséges megvalósítási alternatíváit mutatom be. Mivel az adatbázis keresés modul egy implementációs kérdés, ezért annak irodalma nem bemutatható.

3.2.1 Könyvdetektálás

A rendszer első modulja bemenetként a könyvespolcra készült képet kapja, kimenetül pedig a könyvenként szegmentált képeket adja.

3.2.1.1 Éldetektálás alapú megoldások

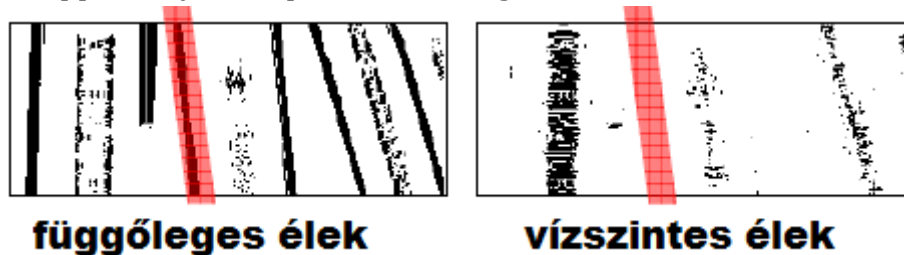
Ilyen megközelítés esetén először valamilyen éldetektáló algoritmus segítségével (Marr-Hildreth [21], Sobel [6], Prewitt [22], Roberts [22], Canny[5]) éldetektálást végzünk a képen, ennek eredményeképpen egy fekete-fehér képet kapunk, ahol a fehér színű pontok az élpontokat jelölik. Majd bináris képen egyenesek keresését végezzük (erre alkalmasabbak között a Hough transzformáció [7]). A felismert egyenesek közül kiválasztjuk a függőlegeseket, esetleg valamekkora hibatűrést megengedve (-15 fok $\rightarrow +15$ fok [2]), ezek az egyenesek jelentik a könyvek szélét, tehát utolsó lépésként ezen egyenesek mentén feldaraboljuk a képet.



4. ábra: A detektált élek [3]

3.2.1.2 FSA alapú megoldás

A könyvek szegmentálására megoldást jelenthet egy véges állapotú automatán (FSA⁶) alapuló megoldás is, erről az [9] irodalomban számolnak be. Véges állapotú automatának nevezzük azt az állapotgépet, amely véges állapotból áll. Ezen megoldás első lépése ugyancsak egy éldetektáló algoritmus elvégzése a képen, amely eredményeképp két kép keletkezik, az egyik csak a függőleges éleket, a másik csak a vízszintes éleket tartalmazza. Az éleket tartalmazó képeket függőleges szakaszokra (egy szakasz hossza 5-10 pixel) osztjuk, és szakaszonként végighaladva vizsgáljuk, hogy mennyi függőleges és mennyi vízszintes élt tartalmaz az adott szakasz. Ezt a lépést szemlélteti az 5. ábra. Az 1. táblázat alapján nagyjából (bizonyos esetekben nem egyértelműen pl. sok függőleges él – sok vízszintes él) meghatározható, hogy az adott szakasz éppen milyen állapotnak felel meg.



5. ábra: Szakaszonként végighaladás a képeken [9]

Adott szakaszban előforduló függőleges élek	sok		kevés	
Adott szakaszban előforduló vízszintes élek	sok	kevés	sok	kevés
könyv szél	lehetséges állapot	lehetséges állapot	nem lehetséges állapot	nem lehetséges állapot
cím	lehetséges állapot	nem lehetséges állapot	lehetséges állapot	nem lehetséges állapot
könyvgerinc	nem lehetséges állapot	nem lehetséges állapot	nem lehetséges állapot	lehetséges állapot
háttér	nem lehetséges állapot	nem lehetséges állapot	nem lehetséges állapot	lehetséges állapot

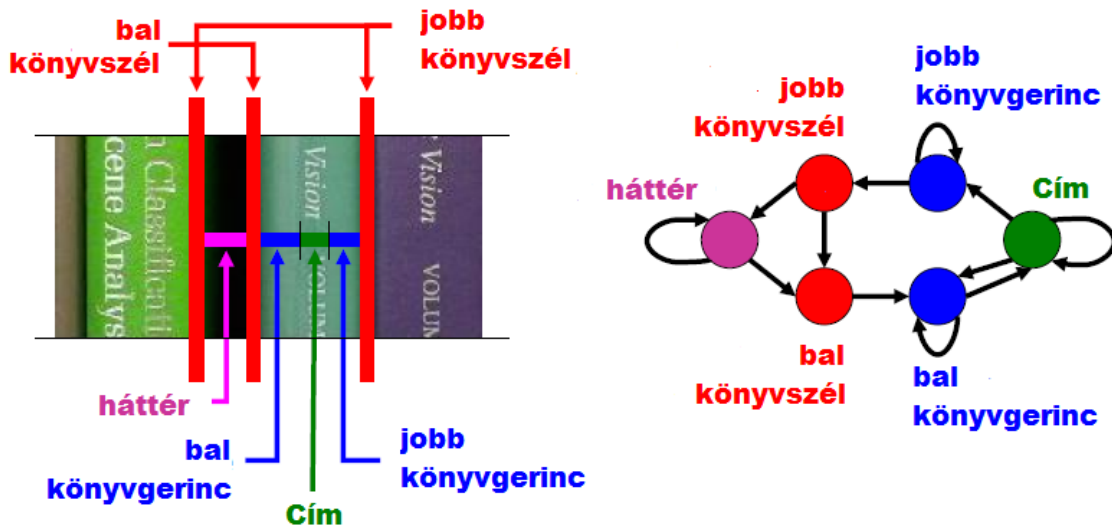
1. táblázat: döntési tábla az élek függvényében

Az 1. táblázatban szereplő viszonyszámok (sok – kevés) sajnos nem határozzák meg pontosan a megfelelő él számát, ezért tesztelésre került sor, aminek az eredménye megmutatta, hogy

⁶ FSA: Finite state automaton

sok élről akkor beszélünk, amikor az összes képen található él 1%-át meghaladó él mennyiség tartozik a tartományba, illetve kevés élről, ha 1% vagy kevesebb él esik a tartományba. Ezen arányok használatával érjük el a legjobb hatásfokot.

Látható, hogy bizonyos esetekben nem határozza meg a döntési tábla egyértelműen a helyzetet, ilyen esetekben fontos az előző stádium is, és aszerint, hogy az előző állapotból milyen állapotba lehetséges a haladás a felismert státuszok leszűkíthetők egyetlen állapotra. Az állapot gráfot lásd a 6. ábrán.



6. ábra: Az FSA állapot gráfja [9]

A6. ábrán jól látható, hogy a rendszer az esetben is jól működik, ha olyan könyvgerincet kell detektálni, amelyiken nem található semmilyen felirat (jobb könyvszél állapotból bal könyvszél állapotba átmenet).

3.2.1.3 Watershed algoritmus

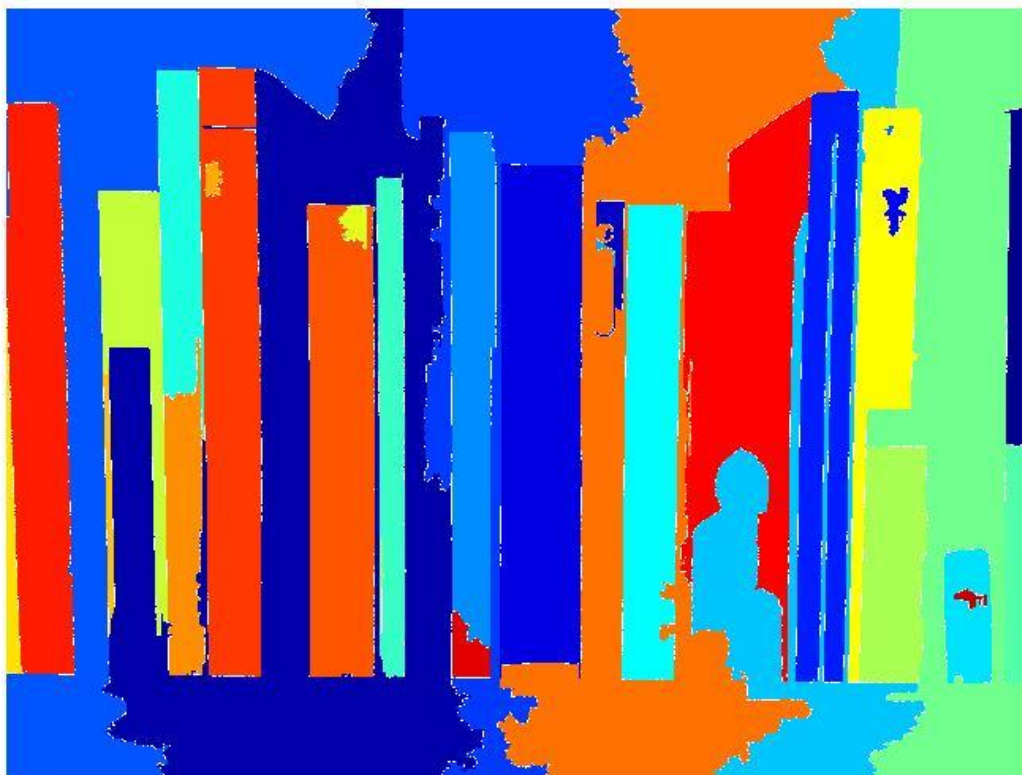
Bár a hasonló rendszerek nem tesznek említést arról, hogy kipróbálták volna a Watershed algoritmust [34], az eredményei alapján úgy tűnt, hogy megfelelő lehet a könyvek szegmentálására.

Az algoritmus menete, hogy először valamilyen éldetektáló algoritmus segítségével meghatározzuk a kép gradiens nagyságát. Jellemzően ehhez Sobel vagy Canny éldetektálót használnak, majd ezután a gradiensek segítségével létrehoznak egy vízválasztókat tartalmazó képet (watershed)

Ez után morfológiai műveletek sora következik az eredeti képen, először egy nyitást végzünk el a képen, amely egy erózióból, majd dilatációból áll. Ezt egy az eredeti képen elvégzett erózió követ, amely után az erodált kép és a nyitáson átesett képmorfológiai újjáépítésen (morphological reconstruction) esik át. Az így kapott képen végrehajtódik egy zárás és egy dilatáció külön-külön, majd ez a kép is morfológiai újjáépítésen esik át.

Ezen része az algoritmusnak a későbbiekben fontos részét képezi a megvalósított rendszernek.

Meghatározása kerülnek a lokális maximumpontok, majd ez alapján binarizálásra kerül a kép, amely alapján már egyértelműek az egyes vízválasztók (watershed).



7. ábra: A Watershed algoritmus eredménye könyvek esetében

A 7. ábrán látható a Watershed algoritmus eredménye egy könyvespolci sort tartalmazó képen. Jól látható, hogy ezek az eredmények jóval gyengébbek, mint az éldetektálást követő szegmentáció esetében (Canny éldetektálás, majd Hough trnaszformáció).

Feltehetően egyes esetekben a könyv gerinc színe és a háttérszín, vagy két egymást követő könyvgerinc színe nem mutatott kellő eltérést, ezért ez az algoritmus egy objektumként ismerte fel őket.

Az eredményei miatt ez az algoritmus a végső rendszerbe nem került implementálásra.

3.2.2 Cím felismerés

Ezen fejezet a lehetséges cím felismerési módszereket ismerteti. Ez a modul egy könyvet ábrázoló képet kap bemenetül és a modul kimenete az adott könyv címe.

3.2.2.1 Mintaillesztés alapú

Mintaillesztés alapú felismerés esetén szükséges egy adatbázis, melyben tároljuk a könyvekről már előre elkészített képeket, illetve szöveges formátumban a könyvek címét. Ez a megoldás kiválasztja a képek alapján a legjobban egyező könyvet az adatbázisból és a szövegesen a képpel tárolt címet továbbítja a következő modulnak.

A mintaillesztés referencia pontok alapján történik (referencia pont detektálása és társítása), majd az egyezés pontosságának meghatározása következik, illetve ezután a legnagyobb egyezésű kép kiválasztását végzi ez a modul.

A referencia pontokkal kapcsolatos algoritmusok két nagy részre oszthatók. Előbb jellemző pontokat kell meghatározni egy-egy képen, ezt nevezik kitüntetett pontkeresésnek, majd ezeket a pontokat egy másik képről való ponttal társítani ezt nevezik kitüntetett ponttársításnak. Bár a két téma szorosan kapcsolódik egymáshoz, a két probléma megoldása jelentősen eltér, függetlenek egymástól.

Kitüntetett pont meghatározásról akkor beszélünk, amikor egy képen olyan pontokat keresünk, amelyek összessége viszonylag egyértelműen meghatározza a képet (jellemzően ezek a sarok pontok). Kitüntetett pont meghatározását végezhetjük többek között: SUSAN sarok detektáló [16], Harris sarok detektáló [17], Laplacian of Gaussian⁷ (LoG), difference of Gaussians³ (DoG), determinant of the Hessian³ (DoH) [18] algoritmusokkal.

Példa a mintaillesztés alapú társításra a 8. ábrán látható.

⁷ A magyar nyelvű irodalom is az algoritmusok angol nevét használja



8. ábra: Referencia pont illesztés könyvek esetén [14]

Kitüntetett ponttársítás esetén két különböző képen detektált kitüntetett pontokat párosítjuk egymással össze. Ilyen társításra alkalmas algoritmusok a MOPS [19], GLOH [20], SIFT[18], SURF[8].

Ezek után a sikeres társítások és az összes kitüntetett pont arányának alapján rangsorolódnak az adatbázisban lévő képek, majd a legpontosabb egyezésű alapján meghatározásra kerül a cím.

Kitüntetett pont meghatározó és társító algoritmusok összehasonlítása a 4.3 fejezetben került részletezésre.

3.2.2.2Szín alapú

Az előző (3.2.2.1 fejezetben) bemutatott módszerhez nagyon hasonló a szín alapú felismerő, társító modul. Ebben az esetben is szükség van egy adatbázisra, amelyben a könyvgerinceket ábrázoló képek tárolódnak. Azonban fő különbség az előző módszerhez képest, hogy nem a képeken felismert referencia pontok alapján végzi a képek összehasonlítását, hanem a képek színösszetevőit alapján.

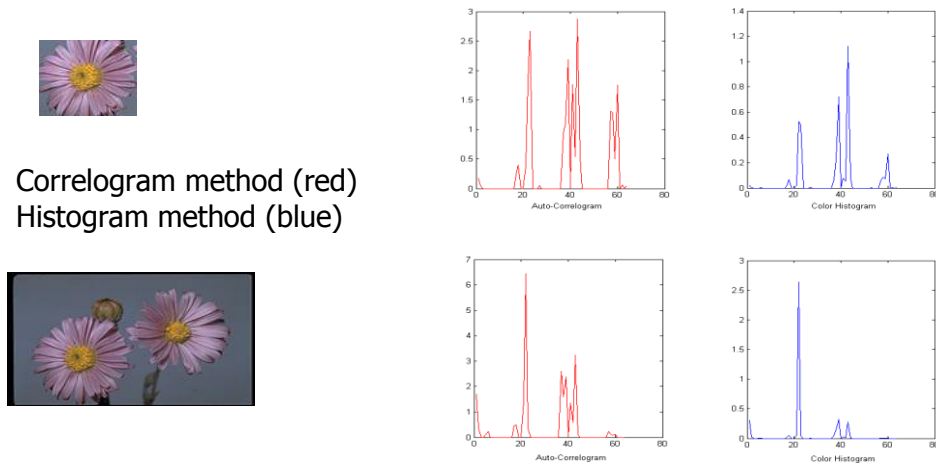
A szín alapú adatbázis keresések két nagy csoportra oszthatók. Az egyik esetben a képeken előre meghatározott pontok színeit hasonlítjuk össze és a színek bizonyos színtérben való euklideszi távolsága alapján határozzuk meg a távolságot [28].

Másik lehetőség, ha a képekről készült hisztogramokat hasonlítjuk össze. Ekkor azonban olyan információkat elvesztünk, mint az orientáció, méret, testek elhelyezkedése.

Hisztogram alapú megközelítések közül az első jelentős a Jain és Vailaya [29] által publikált. A módszer lényege, hogy a tesztek során a kép összehasonlításához három különböző hisztogram hasonlóságát vizsgálták. Azért hármat, mivel minden RGB komponenshez tartozott egy-egy hisztogram. Az egyezést a hisztogramok egyes rekeszeiben tárolt értékeinek euklideszi távolságából számolja ki. Ez alapján, az adatbázisban tárolt képek közül a legkevesebb mintakép és adatbázisban tárolt kép közötti távolsággal rendelkező képpel társítja a keresett képet és ez alapján történik a megfelelő adatbázisban tárolt kép és egyéb információk kiválasztása.

A Jeong [30] által publikált megoldás a speciális algoritmusok elemzésével és optimalizálásával egy jelentősebb analízist vázolt fel a hisztogram alapú társítások és megfeleltetések terén. A Jeong által javasolt módszer hisztogram készítési algoritmus jelentősen eltér a fenn bemutatott Jain és Vailaya [29] által javasoltétól. Ezen algoritmus, ahelyett, hogy egyenként mindenegyes piros, zöld és kék (RGB) csatorna hisztogramját külön – külön kiszámítaná, rekeszeket hoz létre, amely rekeszek mind a három csatornából tartalmazzák az információkat. Természetesen ezen rekeszek mérete jóval nagyobb mint az előző fejezetben bemutatott módszernél, azonban az algoritmus futási ideje gyorsabb, mivel csak egy hisztogrammal kell összehasonlítást végeznie. Ezen kívül ez a módszer már nem RGB színtérben végzi az összehasonlításokat, hanem HSV (hue, saturation és value) színtérben, amelyben már nagyságrendekkel jobban kezelhetők a fényviszonyokból adódó színelterések.

Ezen kívül lehetőség van egy hisztogramtól eltérő, korreogram alapú megközelítésre is [31]. Egy szín hisztogram egy kép globális szín eloszlását írja le. Egyszerűen meghatározható és érzéketlen a képeken lévő kisebb változásokra. Ugyanakkor nem tartalmaz semmilyen térbeli információt a képről, ezért könnyen eredményez fals pozitív eredményeket. Ez a tulajdonsága különösen a nagyobb adatbázisok esetén jelent problémát. Illetve a hisztogram nem robosztus módszer a nagy nézet belüli különbségekre (például egyik kép egy másik adatbázisban lévő képnek csak része). Egy szín korreogram kifejezi, hogy több színpár térbeli korrelációja hogyan változik távolság hatására (a korreogram kifejezést térbeli adatelemzésből kölcsönöztük). A korreogram és hisztogram közti különbségre a 9. ábrán látható példa.



9. ábra: A képek és azok hiszto- illetve korreogramjaik

A módszer nagy hátránya, hogy fekete-fehér képek esetén nagyon gyenge hatásfokot mutat, illetve a különböző fényvisszaverődésre (illumination) is érzékeny.

3.2.2.3 Karakterfelismerés alapú

Karakterfelismerés esetén a szegmentált, csak a könyvet ábrázoló képen megkeressük az összefüggő szegmenseket és megpróbáljuk valamilyen karakterként azonosítani.

Jelenleg már létezik jó pár karakter felismerő rendszer, ilyenek például a Tesseract [22] és aGOCR[23].

A karakterfelismerést végezhetjük rögtön a bemenetül kapott képen, vagy valamilyen előfeldolgozással javíthatunk még a képen, hogy pontosabb felismerést érjünk el. A 10. ábrán látható az előfeldolgozás nélküli hatékonyság, belátható hogy ez meglehetősen gyenge eredmény, tehát érdemes alaposan foglalkozni az előfeldolgozási lehetőségekkel.



10. ábra: OCR hatékonysága előfeldolgozás nélkül [3]

Előfeldolgozás során a jobb karakterfelismerés érdekében valamilyen adaptív küszöböléssel binarizálhatjuk a képet (erre egy megoldás az Otsu algoritmus [25], vagy a Sauvola algoritmus[26] esetleg Niblack binarizáció [27]). Megfelelő megoldás lehet az MSER [10] is.

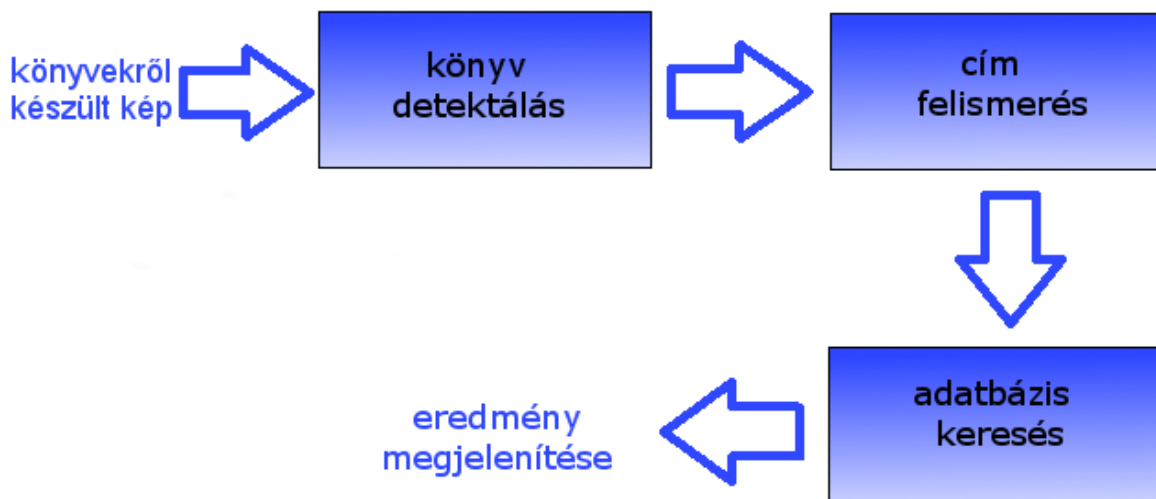
Ezen megoldások összehasonlítására és értékelésére az 5.3.1 fejezetben térek ki.

4 A rendszer

E nagy fejezetben a saját fejlesztésű rendszer kerül bemutatása. Részletesen ismertetem a rendszer felépítését, indoklom az egyes modulok szükségességét, ezt követően összevetem az irodalomkutatásban ismertetett módszerekkel (3.2 fejezet) és bemutatom, hogy milyen megoldás került megvalósításra.

4.1 Rendszervázlat

A hasonló rendszerek alapján (lásd 3.1 fejezet) három modulra érdemes bontani a rendszert: egy könyvdetektáló modulra, egy cím felismerő modulra és egy adatbázis keresés modulra. A rendszer vázlata a 11. ábrán látható.



11. ábra: A rendszer vázlata

A könyvdetektáló modul feladata, hogy a bemeneti képen felismerje a különböző könyveket, hogy ezeket szegmentálva inputként szolgáltassa a következő modulnak. Az olvasóban felmerülhet a modul szükségességének megkérdőjelezése. Megoldást jelenthet az is, hogy a rendszerbe érkező képen azonnal cím felismerést végzünk, és a detektált szövegek távolságából következtetünk arra, hogy azonos könyvön találhatók-e. Így nem lenne szükség a könyvek szegmentálására, és javíthatnánk a rendszer gyorsaságán, azonban az irodalomban [3] és a rendszer fejlesztése során is (lásd: 5.3 fejezet) tesztelve lett ez a megoldás, és kiábrándító 10% alatti pontosságot hozott.

Cím felismerő modul bemenetül egy könyvet ábrázoló képet kap és kimenetül a képen található szöveget tartalmazza. Feladata valamilyen előfeldolgozás után minél pontosabb szöveget szolgáltatni.

Az adatbázis kereső modul egy címet kap bemenetül és a cím alapján további információkat juttat a felhasználóhoz. Ez a modul keresést végez nem csak a saját rendszerhez tartozó adatbázisban, hanem az interneten is. Feladata sikeres internetes keresés esetén (internetről olyan információhoz jutott, ami a saját adatbázisban még nem szerepelt) nem csak a

felhasználónak szolgáltatni újonnan szerzett információt, hanem a saját rendszerhez tartozó adatbázisban is eltárolni.

4.2 Könyvdetektálás

Az irodalomkutatás fejezetben (lásd 3.2.1.) már ismertettem a lehetséges könyvdetektáló algoritmusokat, e fejezetben összehasonlítom őket és kiválasztom a legjobb hatásfokút.

Két lehetséges módszer került bemutatásra. Az éldetektálás alapú megvalósítás, ez esetben élt detektálunk majd Hough transzformációval meghatározunk vonalakat és ezek a vonalak adják a könyvek szélét. A másik lehetséges megoldás egy FSA alapú megközelítés, amikor éldetektálás után egy véges állapotú automata segítségével különböző állapotokat rendelünk a kép részeihez (háttér, könyv szegély, gerinc, cím).

Minden képfeldolgozással foglalkozó rendszernél szükséges pár szót ejteni a színterekről. Mind a két bemutatott megoldás (3.2.1.1, 3.2.1.2. fejezetek) éldetektálást használ, amelynek hatásfoka függ a színterektől [12]. Az irodalomban [12] összehasonlítást végeztek és az RGB, HSV, YIQ [15]színterek közül a HSV színtér a legmegfelelőbb az éldetektálás elvégzésére, a legtöbb él így detektálható [12].

Mivel mindkét ismertetett módszer éldetektálás alapján dolgozik, ezért első lépésben a legjobb hatásfokú éldetektáló eljárás megtalálása volt a célom. Ehhez a [13] irodalmat használtam, amely összehasonlította a lehetséges éldetektáló módszereket. [Az értékelésnél figyelembe vette a fals pozitív találatok számát, fals negatív találatok számát, él dőlésszögének pontosságát, és az átlagos eltérést a valós él és a detektált él között. Ezen összehasonlítás [13] eredménye alapján a Canny éldetektáló [5] a legpontosabb.

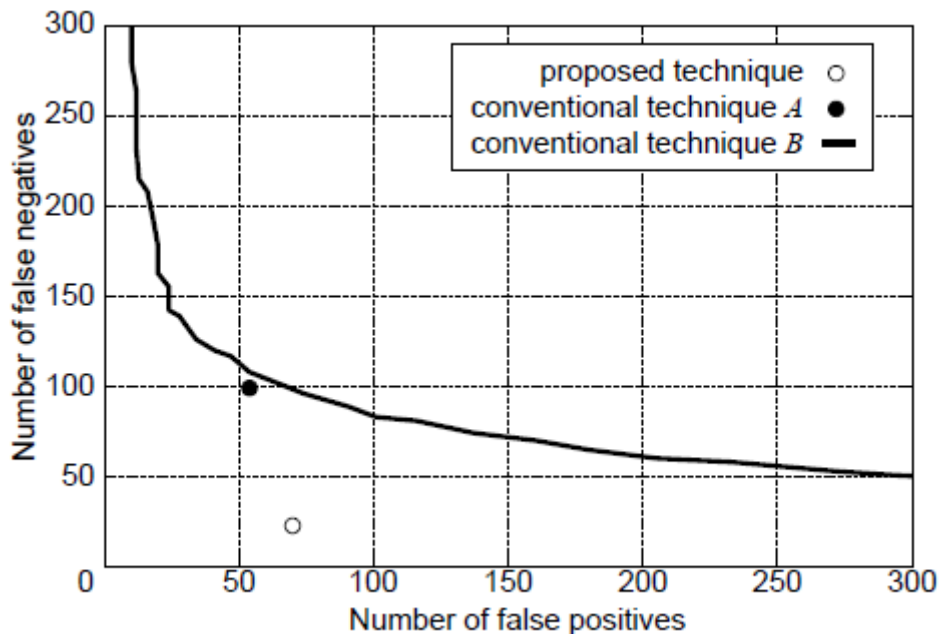
Mivel a fejlesztett rendszer egyik célja, hogy minél szabadabb pozícióba is lehetővé tegye a könyvek detektálását, az éldetektálást követően meghatározott függőleges és vízszintes élek aránya alapján elforgatásra kerülhet a kép 90 fokkal (részletek a 5.1 fejezetben). Így ha a könyvek nem egymás mellett állnak a képen, hanem egymáson fektetve helyezkednek el, a forgatás segítségével akkor is lehetővé válik a szegmentálás. Ilyen elforgató megoldással nem rendelkeznek a bemutatott hasonló rendszerek [2, 3, 4].

Majd az FSA és az éldetektálás alapú megoldás került összehasonlításra. Az irodalomban szereplő pontossági értékek alapján végeztünk összehasonlítást. Három lehetséges megvalósítás került szóba: Hough – transzformáció konstanst küszöb értékkel (conventional technique A), Hough – transzformáció adaptív küszöbölés használatával (conventional technique B) és a véges állapotú automata (proposed technique). Az eredmények a 13. ábráról leolvashatók.

Mivel az A esetben adaptív küszöbölést használtak, ezért ott a különböző küszöbök miatt vonalként jelennek meg az eredmények, a másik két esetben egy küszöb érték került használatra, ezért azon esetekben csupán pontként jelenik meg az eredmény.

E mérés esetében a mért eredmény minél jobban megközelíti mindkét tengely mentén a nulla értéket, annál jobb eredményről beszélhetünk (a kisebb érték a jobb).

Az eredmények a 12. ábráról olvashatók le.



12. ábra: A különböző könyvszegmentáló technikák hatásfoka [9]

Az FSA módszert 610 könyvből álló tesztalmazon mérték (összesen 60 darab kép) [9]. 218 könyv esetében teljesen függőlegesen álltak a könyvek, nem volt semmilyen dőlés, ilyen esetben 90%-os pontosságot ért el a rendszer. Egyező irányú dőlés esetén, 197 könyvön tesztelve 88,8% volt a pontosság, míg össze-vissza dőlő könyvek esetében 91,2%. A fals negatív találatok, tehát amikor könyvet kellett volna detektálni ám ezt a rendszer nem tette, 3,2%-os arányban fordultak elő a teljes tesztalmazra nézve.

Hough-transzformáció esetén A (conventional technique A) esetben 13,6% volt a fals negatív találatok száma. B (conventional technique B) esetben ez a szám közel kétszerese volt az A esetben tapasztaltaknak.

Fals negatív találatok esetén információ-vesztés keletkezik, míg fals pozitív esetben többlet információ, ami későbbi megfelelőfeldolgozás esetén még kiszűrhető, azonban az elvesztett információ már nem pótolható, ezért a lehető legkevesebb fals negatív teszt eredménnyel rendelkező megoldás, azaz a FSA kerül megvalósításra a tervezett rendszerben.

Későbbiekben további javításokat végeztem ezen a modulon. Figyelembe vettem milyen előfeldolgozási módszerekkel javítható tovább a könyv szegmentáció, illetve milyen alternatív, jobb megoldások kínálóznak a Hough transzformáción kívül a probléma megoldására. Az előfeldolgozási lehetőségeket a következő fejezetben részletesen mutatom be.

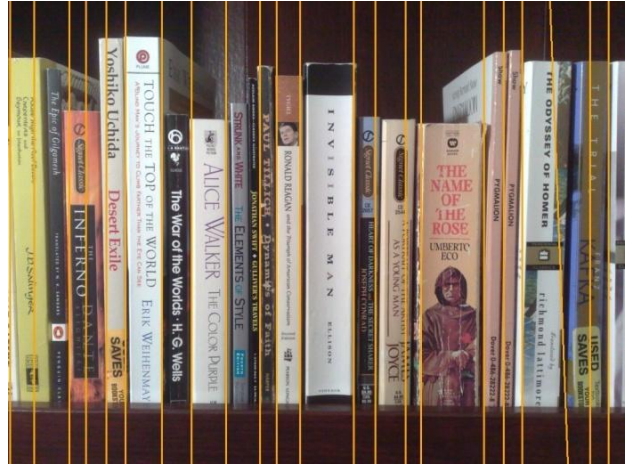
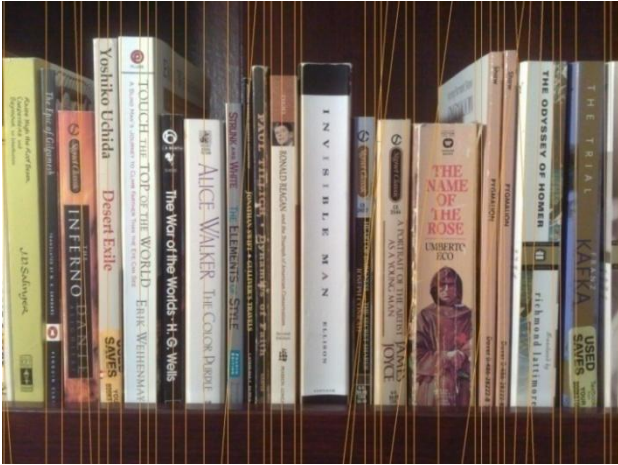
4.2.1 Átméretezés

A már rendelkezésre álló tesztadatbázis [3] és saját tesztképekkel folytatott tesztek során éles különbséget tapasztaltabb a nagyobb és kisebb felbontású képek eredményei között. Célunk a könyv gerincek széleinek a detektálása, ez általában a képeken detektált hosszú (a kép 60-80% hosszú) élként jelenik meg. Azonban nagy felbontású képek esetén a jobb felbontás miatt sokkal több él is detektálódik a képen, például a könyveken lévő betűk körvonala vagy a gerinc grafikáján történt színváltozás. Ezek a detektált élek rendszerünk nem fontos információk, zajnak tekinthetők, amelyek csak nehezítik a későbbi felismerés pontosságát. Tehát elhanyagolhatók. Kisebb felbontású képek esetén sokkal jobb hatásfokot ért el a szegmentálás, mivel ezek a felesleges élek nem kerültek detektálásra, így később egy másik modulban nem eredményeztek fals pozitív találatot. Viszont a felismerés se romlott a kisebb képek esetében, mivel felismerni kívánt élek általában olyan markánsan vannak jelen a képeken, hogy azok kisebb felbontású, rosszabb minőségű képen is felismerhetők.

Ennek a tapasztalatnak köszönhetően ez a modul kiegészült egy átméretező lépéssel, amely jelentősen segítette a fals pozitív találatok csökkentését. Felmerülhet az olvasóban, hogy ha az átméretezéssel a célunk csupán az, hogy a kevésbé fontos éleket a képről eltüntessük vagy csökkentjük azt, amellet, hogy a fontosabb markáns (jellemzően a könyvek szélén) megjelenő éleket meghagyjuk, akkor miért nem valamilyen szűrő algoritmust alkalmaztunk. Például erre a célra megfelelő lenne egy Gauss elmosás is.

Néhány teszt azt mutatta, hogy nagyon hasonló eredményt hoz az átméretezés és a Gauss elmosás alkalmazása, azonban egy területen rendkívül nagy eltérés volt a kettő között, mégpedig a futási időben. A kép átméretezése esetén nem csak a fellelhető élek száma, de az egész kép mérete csökken, aminek köszönhetően ugyancsak csökken a Canny éldetektáló és a Hough transzformáció futási ideje. Ami hatalmas milisecundumokban mérhető (lásd a melléklet ide kapcsolódó táblázatát) időnyereség. Ezért a szinte megegyező működésmelletti jobb futási idő miatt került az átméretezés megvalósítására a választás.

A 13. ábra: Eredeti méretű kép a detektált könyvcszélekkel 14. ábra: Átméretezett kép a detektált könyvcszélekkel ábrákon a sárga vonalak jelzik a detektált könyvcszéleket. Jól látható a fals pozitív találatokat illetően a különbség a két kép között.



13. ábra: Eredeti méretű kép a detektált könyvszélekkel 14. ábra: Átméretezett kép a detektált könyvszélekkel

Továbbá fontos megjegyezni, hogy ez a méret csökkentés csak a könyvszegmentáló modul hatásfokán javít, a későbbi mintaillesztés alapú, vagy szövegfelismerés alapú modul nagyobb bementi kép esetén jobb pontosságot ér el. Tehát az éldetektálás és könyvszegmentálás után a feldolgozást az eredeti képen folytatja tovább a rendszer.

Kapcsolódó részletes tesztképek a mellékletek között tekinthetők meg.

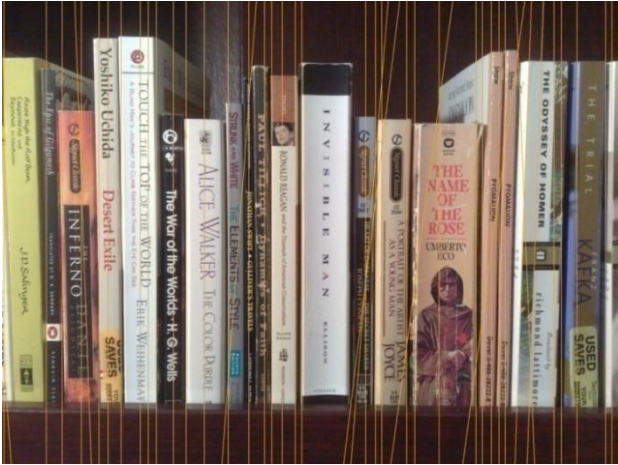
4.2.2 Nyitás- Zárás

Az irodalomkutatásban részletesen bemutatott Watersphed [34] algoritmus könyveken való tesztelése során tűnt fel, hogy morfológiai operátorok használata a képen jelentősen javít a végső eredményen.

Az egymást követő nyitás, zárás és morfológiai újjáépítés hatására a könyveken található alakzatok (jellemzően betűk) eltűnnek, aminek hatására az előző fejezethez hasonlóan felesleges élekkel csökken a könyvgerinceket tartalmazó képünk, és javul a fals pozitív találatok száma (csökken).

Az eredeti képen (esetünkben az átméretezett képen), először egy nyitást végzünk el a képen, amely egy erózióból, majd dilatációból áll. Ezt egy az eredeti képen elvégzett erózió követ, amely után az erodált képet és a nyitáson átesett képet morfológiai újjáépítésen (morphological reconstruction) esik át. Az így kapott képen végrehajtódik egy zárás és egy dilatáció külön-külön, majd ez a kép ismorfológiai újjáépítésen esik át

A műveleteken átesett képet a 15. ábrán lehet megtekinteni, ahol a sárga függőleges vonalak jelzik a detektált könyv széleket. A 16. ábrán az eredeti átméretezésen és morfológiai műveleteken nem átesett kép látható, ugyancsak sárga függőleges vonalakkal a detektált könyvszélekkel. Szabad szemmel is jól látható az előfeldolgozási (átméretezés, morfológia) műveleteken átesett és nem átesett képek közti különbség.



15. ábra: Eredeti kép a detektált könyvszélekkel



16. ábra: Módosított kép a detektált könyvszélekkel

Kapcsolódó részletes tesztképek a mellékletek között tekinthetők meg.

4.2.3 LSD – Line Segment Detection

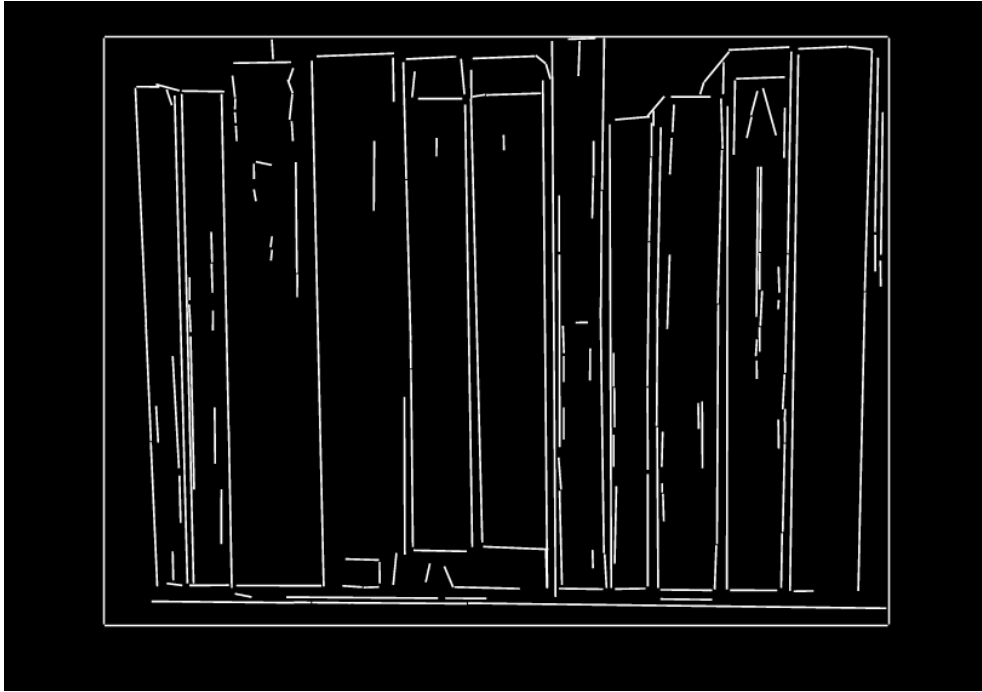
Az LSD – Line Segment Detection algoritmus [33], a Hough transzformációhoz hasonlóan egy vonaldetektáló algoritmus. Ez az algoritmus egy szürkeárnyaltos képet vár bemenetül és a detektált vonal szegmensek listáját adja vissza. Ez egy automatikus képanalizáló algoritmus, tehát más bemeneti paraméterre nincs szüksége a vonalak detektálására.

Ez a módszer az irodalomban leírtak szerint [33] először a képet az eredeti méret 80%-ára kicsinyíti. Ugyancsak zajszűrés és a feldolgozás gyorsítása a kicsinyítés célja. Mivel már a saját rendszer rendelkezik egy átméretezés lépéssel, ezért ezt a 80%-ra való átméretezést újra nem végzi el.

Az LSD képen lokális egyenes kontúrokat, úgynevezett Line segment-eket keres. Ezek a kontúrok olyan zónái a képnek, amelyekben kellően gyors színátmenet valósult meg fehérből feketébe vagy fordítva. Ezekhez a zónákhoz egy-egy dőlésszöget is meghatároz, azaz, hogy a Line segment milyen irányú dőléssel van jelen a zónában.

Ezen kívül az algoritmus egy kétállapotú állapotgépet is felhasznál. A két állapot a Used (használt) és Not Used (nem használt). Az állapotok számára a kép méretével megegyező tömböt létrehoz, amelyben minden egyes zónára eltárolja, hogy a feldolgozás közben már előzőleg használatban voltak-e egy másik vonal meghatározása során.

Majd régiónövelő módszerrel az azonos irányú, not used állapotú Line segmenseket összefűzi. Ezek a régiók adják az egy-egy detektált vonalat.



17. ábra: Detektált élek LSD módszerrel

A 17. ábrán látható egy könyvgerinceket ábrázoló képen elvégzett LSD transzformáció, a képen szabad szemmel is jól kivehető, hogy mennyire pontos az éldetektálás. A tesztkép a Line Segment Detection előtt már előfeldolgozva lett, azaz át lett méretezve, és átesett a megfelelő morfológiai műveleteken, hogy eltűnjenek a nem kívánt a könyv cím miatti fals pozitív vonal detektálások.

A képről jól látszik, hogy egyrészt részben az előfeldolgozásnak hála, másrészt az LSD-nek köszönhetően csupán kevés olyan vonal detektálódott, amely nem kívánt könyv szél. De ezeket a nem szükséges éleket is sikerült kiszűrni, egy egyszerű hosszúság vizsgálattal. Ha egy él hossza nem érte el a kép méret 40%-át akkor az eldobásra került. Illetve a Vízszintes élek is eldobásra kerültek.

A mellékletek között található teszt eredmények kimutatták, hogy mint sebességileg, mint pontosságilag az LSD felülmúlta a Hough transzformációt.

A 18. ábrán egy már megfelelő előfeldolgozáson (Átméretezés, cím eltüntetés) LSD módszerrel detektált könyv szegmensek láthatók. A sárga függőleges vonalak a vágások helyét jelöli.



18. ábra: Az átméretezésen és morfológiai műveleteken átesett képen LSD módszerrel detektált élek

Részletesebb teszt eredmények a mellékletek között találhatók.

4.3 Cím felismerés

E fejezetben a megvalósított cím felismerő modult mutatom be. Az irodalomkutatás fejezetbe már bemutatásra kerültek a hasonló rendszerek megoldásai. Három lehetséges módszer van, a szöveg alapú, amikor a bementi képen megpróbálunk betűket felismerni, és ezen betűk kerülnek a rendszer kimenetelére, illetve a mintaillesztés alapú, amikor a bementi képet egy képadatbázissal vetjük össze, a képeken található referencia pontok alapján és a legpontosabban egyező képhez adatbázisban tartozó címet adjuk a kimenetre. Illetve a szín alapú társítás, amikor az előzőhöz hasonlóan egy képi adatbázissal társítjuk a mintát és a legjobban egyező adatbázis elem alapján határozzuk meg a címet. Ezen megoldásnál azonban nem a referencia pontok alapján végezzük az összehasonlítást, hanem a képeken található színek alapján.

A hasonló projektek[2, 3, 4] mind a mintaillesztés alapú megközelítésre támaszkodtak, ez a projekt célja miatt viszont nem alkalmazható, ugyanis kezdetben nem áll rendelkezésünkre képadatbázis, amellyel az összehasonlítást végezhetnénk.

A saját megvalósítás is a 3.1.3 fejezetben ismertetett rendszerhez hasonlóan egy hibrid megoldás lesz. Az irodalomban jegyzett értékek [4] és a saját mérések (lásd 5.3 fejezet) is azt bizonyítják, hogy csak szöveg alapú módszer pontossága nem kellően jó, ahhoz hogy csak az alapján működőképes rendszert tudjunk építeni. A saját rendszer abban eltér a 4.1.3 fejezetben ismertetett megoldástól, hogy hasonló rendszereknél alkalmazott mintaillesztés alapú megoldás helyett a saját rendszerünk szín alapú adatbázis keresést végez. Ez a modul csupán a szöveg alapú kiegészítésre szolgál. A rendszerünk esetében először karakterfelismerést végzünk, és ha így sikeres találatra jutunk, akkor a képet felvesszük az

adatbázisba, így folyamatosan bővül az adatbázisunk, rendelkezünk egy tanulási folyamattal, amellyel a többi rendszer nem, illetve ilyen megközelítés esetén, hogy a színfelismerés csak kiegészítő lépés, belátható, hogy kezdetben nem szükséges az adatbázis megléte, az épül „magától” a detektálások során. Természetesen, ha a szöveg alapú felismerés nem hozott eredményt, akkor a rendszer a már meglévő képadatbázisban keres.

4.3.1 Szöveg alapú cím felismerés

A szöveg alapú megközelítés négy részre osztható: binarizációra, orientáció meghatározásra, karakterfelismerésre és utófeldolgozásra. E modul megvalósítása állhat simán egy karakterfelismerésből, vagy megvalósítható úgy is, hogy előfeldolgozás előzi meg a karakterfelismerést, illetve azt utófeldolgozás is követi. Jelen megvalósítás az előfeldolgozást két részre bontotta: binarizációra és orientáció meghatározásra.

Mérések bizonyították (5.3 fejezet), hogy előfeldolgozás nélkül használhatatlanul gyenge hatásfokot ér el a karakterfelismerés. Tehát a saját rendszerünk esetén mindenképpen előfeldolgozó modul/modulok is implementálásra kerültek.

A binarizáció célja, hogy egy olyan fekete-fehér képet hozzon létre, amely az eredeti képhez képest kiemeli a képen található betűket. Ezzel segítve az OCR engine felismerésének javulását. Szükségességét olyan tesztek igazolták számunkra, mikor mesterségesen képszerkesztő program segítségével elkészítettük a tökéletesen binarizált változatát az eredeti képnek és inputként ezen verziót, illetve az eredeti és az eredeti kép szürke árnyaltos verzióját teszteltük az OCR engineken. Nagyságrendekkel jobb eredményt ért el a karakterfelismerés binarizált képen, mint a másik kettő változaton.

Az orientáció meghatározó modul szükségességét az indokolta, hogy a tesztek során kiderült, hogy a karakterfelismerő engine érzékeny a felismerendő betűk irányultságára, tehát messze nem elhanyagolható, hogy fejtetőn, vagy valamekkora fokkal elforgatva szerepel a képen az adott betű. Ezen modul célja, hogy megfelelő orientációban, irányultságba helyezze a képen szereplő betűket.

E két modul (binarizáció és orientáció meghatározás) valósítja meg a szükséges előfeldolgozásokat a karakterfelismerés előtt. A karakterfelismerő modul célja, hogy a képről meghatározza azokat a részeket, amelyek betűk és ezek alapján egy szöveget adjon kimenetül.

A tesztek sajnos azt bizonyították, hogy mivel a binarizáció eredménye nem teljesen pontos, néhol zajok is maradnak a képen, amelyeket a karakterfelismerő modul különböző betűknek ismer fel. Illetve előfordulhat az, hogy egy-egy betűt hibásan ismer fel a karakter felismerő, ezért szükség van egy felismerő utáni utófeldolgozó modulra, amely a már szöveges információban található esetleges hibákon javít.

Az irodalomkutatásban ismertetett módszerek (3.2.2.2 fejezet) összehasonlítására, mérésére nem állt rendelkezésre megfelelő irodalom. Ezért a megfelelő előfeldolgozási és

utófeldolgozási algoritmusok, illetve a megfelelő karakterfelismerő a saját mérési eredményeink alapján kerültek meghatározásra.

Ezen fejezet (4.3 fejezet), amely részletesen a könyvgerincről való szövegfelismeréssel foglalkozik, mind a rendszer által használt módszerek mind a rendszer eredményességéről készült cikk elfogadásra és bemutatásra került a 2014. Márciusában megrendezésre került XIX. Fiatal Műszakiak Tudományos Ülésszakán [32], valamint a 15th IEEE International Symposium on Computational Intelligence and Informatics (CINTI 2014) nemzetközi konferencián.

A szövegalapú felismerés lépéseit a következő ábra szemlélteti:



19. ábra: A szövegalapú felismerés modul almoduljai

Annak érdekében, hogy el tudjunk dönteni, hogy melyik binarizációt használjuk, először a képnek elkészíti a program a hisztogramját, ha ezennagy számban található sötét vagy világos érték, arra következtethetünk, hogy a könyv kevés képi információt tartalmaz a szín intenzitásnak megfelelően a Sauvola algoritmussal történik a binarizáció, ha ilyenre nem következtethetünk a binarizáció alapján Otsu binarizációt használunk.

4.3.1.2 Orientáció meghatározás

A szöveg orientációjának meghatározása is egy fontos kérdés, mivel a karakterfelismerő (OCR) csak a megfelelően elhelyezkedő betűket képes detektálni, azaz amelyek nem fejtetön, vagy adott szöggel elforgatva helyezkednek el.

Sajnos a könyv orientációjából nem következik a gerincen lévő betűk orientációja, mivel egyrészt a nemzetközi szabvány szerint a szöveget fentről lefele, a magyar szabvány szerint lentől felfele kell a könyvgerincekre írni, másrészt előfordulhat esztétikai okból ettől a szabványtól való eltérés is, például a szerző neve állítva, a cím elfordítva szerepel a könyvgerincen.

Erre a problémára megoldást jelentett, hogy dilatációt végeztünk a binarizált képen, annak érdekében, hogy a detektált betűk összeérjenek. Ezek után egyszerűen meghatározhatóvá váltak az egy-egy összefüggő szövegrészek, amelyek szerint szegmentálható a kép, majd az egyes részeket különböző irányba elforgatva többször is karakterfelismerést végeztünk. Amelyik irányban a legnagyobb egyezést mutatja az adatbázisból egy rekorddal az adott rész, az a megfelelő irány.

Ez a megvalósítás azonban a könyvek változatos színvilága miatt (sötét háttéren világos szöveg, világos háttéren sötét szöveg) nem javított jelentősen az ennél egyszerűbb és emiatt gyorsabb megoldáson, azaz amikor az egész binarizált képen négyszer végzünk karakterfelismerést (eredeti állapot, 90 fokkal való elforgatás, 180 fokkal való elforgatás, 270 fokkal való elforgatás). Mivel ez a modul bemenetül a megfelelően szegmentált képeket kapja, amelyek egy-egy könyvgerincet tartalmaznak a megfelelő (állított) helyzetben ezért elég ennek a négy iránynak a vizsgálata, eldől, ferde könyv nem fordulhat elő.

Ez esetben időigényesebb a karakterfelismerés (nagyobb területen alkalmazzuk), viszont a dilatáció és szegmentálás elhagyása miatt összességében gyorsabb megoldás. Illetve az elvégzett tesztek kimutatták, hogy a felismerés pontosságára nem volt negatív hatással az utóbbi algoritmus, ezért rendszerünkben ez a megoldás került alkalmazásra.

4.3.1.3 Szöveg felismerés

Ezen modul célja, hogy a képen betűket detektáljon és ezek alapján egy szöveget szolgáltatson kimenetül. Az alkalmazáshoz egy már létező online karakterfelismerő (OCR) engine-t használtunk. Saját tesztjeink során a Tesseract, CuneiForm, GOCR, Ocrad, OCRFeeder, OCRopus rendszerek kerültek vizsgálatra. Az eredmények alapján a legmegfelelőbb ilyen rendszernek a Tesseract OCR engine [23] bizonyult, ezért ezt használtuk. Az engine-t csak alfanumerikus karakterek felismerésére korlátoztuk (könyvek esetén kis eséllyel fordul elő

más karakter). Mivel e modul nem saját fejlesztés, hanem már egy meglévő kész rendszert alkalmaz, ezért a modul ennél részletesebben nem kerül bemutatásra.

4.3.1.4 Utófeldolgozás

A mérések során kiderült, hogy sok esetben a karakterfelismerő a valós képen lévő betűkön kívül a megjelenő zajokat is „felismerte”. A modul célja ezen zajokból eredő hibás betűk felismerése, korrigálása.

Ezen fals pozitív betűk kiszűrésére először egy szótár került kiépítésre, amely értelmes szavakat, könyvcímeket tartalmaz. A feldolgozás során a karakterfelismerés eredménye mindig ezzel a szótárral kerül összevetésre. Az összehasonlítást longest common subsequence (LCS) módszerrel közelítettük meg és amelyik adatbázisban tárolt címmel a leghosszabb részletet ismerte fel az algoritmus azt a rekordot fogadtuk el találatnak. Néhány példa a modul bemenetire és kimeneteire az 1. táblázatban látható.

Bemeneti adat	Kimeneti adat
MlltrtgHBlii.Uie5DCI.il?? Aguadg 8u.maeoT é\DH?1ofrJf?iliiklr Learning OpenCV ttedmt	BradskiLearningOpenCV
NIRDIONULWyZmI 3U CFFamo mFE 3 3D 5munDJceLKeelFilSH!CioZ..o MII svxasmaa d1 t iillliW2lllltill 2llMllllllllgliW gllllglm g5IANQiq.ZLL LBEBFFEc.Z1“5!2EEmw“ncram2 0al.2ElQireECP.2NOIUlllig5 Dynamic Programming and Optimal Control Ire 1x w I BERTSEKAS C	Bertsekas Dynamic Programming and Optimal Control
fhntili?bf.isEl.b.fl T. T. iiiiiiit Illlllil5 FtjgawitiU2FfiE G O H a Deslgn Pattens 101K322 v t	Gamma Design Patterns
EnidHM!Nor?1 gecomdEdioum UQ3AlP1DEFEl Q\$39PPmh TLeCaOLacJeleftl *TWptLe.LeC6LKlaGS. ITeLeQJjkTiQ4marow. dmmm Mill.lllltllllil Pattern Classification E Wb?	Duda Pattern Classification

1. Táblázat: A szótár alapú utófeldolgozás eredményei

Az 1. Táblázaton jól látható, hogy minden esetben a cím közel teljesen pontosan felismerésre kerül (a bal oldali oszlopban félkövérrel kiemelt szavak), azonban nagyon sok zaj is betűként felismerésre került. Jól látható az is, hogy ez a modul a szótár alapján sikeresen kiszűrte a zajokat, tehát nagyjából megfelelő eredményt hoz.

Azonban nagyon zajos esetekben az longest common subsequence (LCS) már nem hozott megfelelő eredményeket. Nagyon zajos szövegeknél, azaz olyan esetekben, ahol nem szinte tökéletesen felismert szöveg illetve zajok jelennek meg, hanem maga a felismert szöveg is pontatlan (például: 111111 INFRODUCTION ro PROBABILITY ti iitiili My), már közel az összes címet hibásan ismerte fel. Példánk esetében az LCS algoritmus nem volt képes a „INFRODUCTION” szóból az „INTRODUCTION” szóval való hasonlóságt felismerni.

Ezért áttértem a Levenshtein távolság [] alkalmazására az LCS helyett. Azaz minden egyes szótárban szereplő és detektált szó távolsága kiszámolására kerül és a legkisebb távolsággal történik a párosítás/ megegyeztetés.

E algoritmus már megfelelő eredményt adott a kissé hibásan felismert címek párosítása esetén is.

Azonban szerettük volna ennek a modulnak a szabadságfokát megnövelni, úgy gondoltuk, hogy bizonyos esetekben problémás lehet a szótárral való hasonlítás a szótár véges mivolta miatt elképzelhetőnek tartottuk, hogy esetleg olyan eset miatt nem sikerül a felismerés, mivel a felismerni kívánt cím nem szerepel a szótárunkban. Ezen problémákra megoldást jelentene egy meglehetősen nagy szótár is, azonban egyrészt a méretből fakadóan jelentősen lassú a megtalálás (Ne felejtjük minden egyes könyv esetén a szótárban található összes címmel párosítunk), másrészt nagyobb szótár esetén is elképzelhető, hogy egy-egy cím/szó hiányzik a szótárból.

Ezért ezt a modult egy teljesen más módszer szerint is megközelítettük. A Google keresések esetén, ha elgépelést feltételez, javaslatot tesz egy másik szó szerinti keresésre. Ez az algoritmus nagy vonalakban, úgy működik, hogy a Google tárolja az összes kereső mezőbe irt szöveget, és ha érzékeli, hogy egy közel a begépelte szóhoz hasonló szóra jelentősebben többen kerestek, mint a begépelte szóra, akkor javaslatot tesz. Sajnos ilyen adatbázis kialakítására és folyamatosan karbantartására nincs lehetőségünk, mivel a felhasználók, semmilyen szöveges inputot nem adnak meg. Azonban szerencsére http-n keresztül lehetőség van kéréseket továbbítani a Google felé, ami kérdésben egy-egy szó van és a Google a saját adatbázisa alapján a fenn leírt módszer szerint JSON válaszban megadja az ahhoz hasonló szavakat.

Ez a módszer is kipróbálásra került. Jól látható, hogy ez esetben semmilyen szótárat nem vesz igénybe a megoldás, így ez esetben a szótározás esetén felmerülő problémákkal nem kell számolni, azonban ez a megoldás egyes esetekben rosszabb eredményt ért el, mint a szótár alapú. Az egyes eredmények a 2. táblázatban láthatók.

Bemenet	Javasolt szavak
EEDUCCDFiaEDICD11MrMtDMrMINi	-
INFRODUCTION	"introduction", "introduction in english", "introduction to r for quantitative finance", "introduction to international legal english", "introduction to algorithms", "introduction to autonomous mobile robots", "introduction to the language of law", "introduction yourself", "introduction to algorithms third edition pdf"
PROBABILITY	"probability", "probability density function", "probability theory", "probability distribution", "probability calculator", "probability jelentése", "probability mass function", "probability measure", "probability magyarul", "probability generating function"

2. táblázat: A Google rendszerének eredményei

A táblázatból látható, hogy sok esetben megfelelő javaslatot szolgáltat a rendszer, azonban olyan esetekben, amikor olyan hibával történik a karakterfelismerés, amely elgépelés nem jellemző az emberekre, ilyen például az „rn”(RN) és „m”(M) betűk összekeverése, nem ad megfelelő találatot a módszer, az általa használt adatbázis sajátosságai miatt. (Ha az emberi felhasználók nem vétének olyan elgépelési hibákat, amelyek karakterfelismerésnél előfordulnak, nem épül ki arra a hibás bevitelre megfelelő adatbázis).

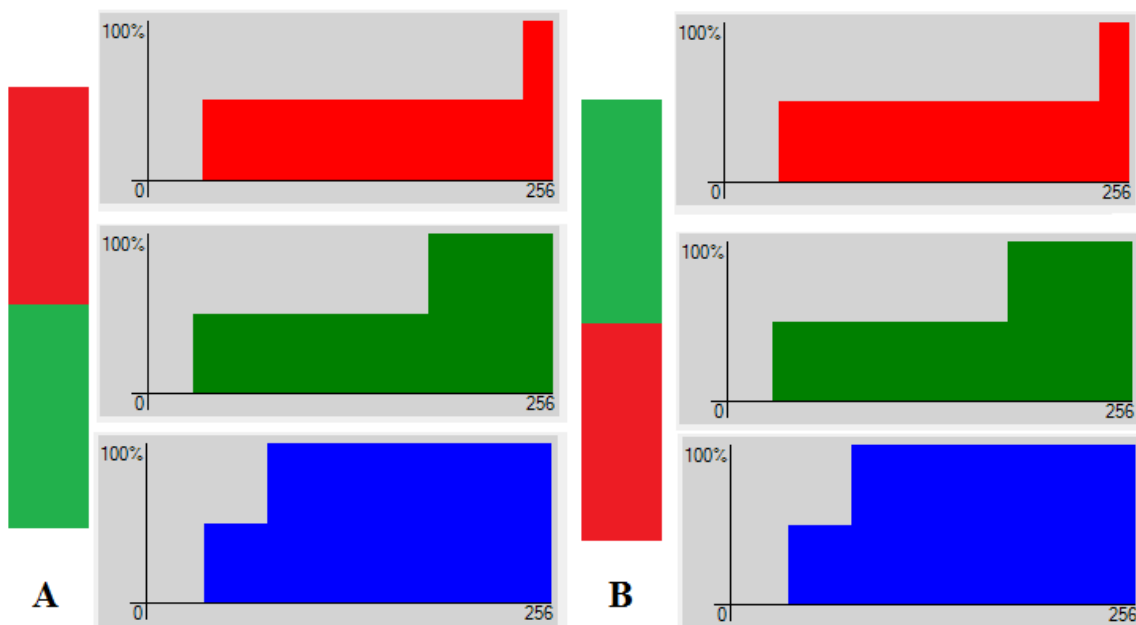
A szótárat használó megoldással jobb eredményeket értünk el, azonban úgy gondoljuk, hogy a modul további fejlődésének az utóbb bemutatott módszer a megfelelőbb. Függetlenebb és általánosabban használható a megoldás, a jövőbeli továbbfejlesztéseknél is ezt a megvalósítást képzeltük el továbbgondolni. Ugyanakkor a jelenlegi rosszabb hatásfoka miatt mindkét megvalósítás szerepel a rendszerben, a felhasználó képes kiválasztani melyiket szeretné használni.

4.3.2 Szín alapú cím felismerés

Szín alapú cím felismerés esetén adott egy adatbázisunk, amelyben tároljuk a könyvgerincekről készült képeket, illetve a hozzájuk tartozó címeket. Ez a modul ezzel az adatbázissal végez összehasonlítást, majd annak eredményeképpen meghatározza az adatbázis alapján a megfelelő könyvcímet. Az adatbázis elemekkel való összehasonlítást a képeken lévő színek szerinti hasonlóság alapján végzi.

A 3.2.2.2 fejezetben bemutatunk egy lehetséges szín alapú keresési módszert. Sajnos a hasonló projektek nem használnak ilyen összehasonlítást, azonban egyik projekt se zárta ki, vagy indokolta meg, hogy miért ne lenne megfelelő megoldás egy ilyen szín alapú összehasonlítás. Rendszerünk azon esetekben, ha kevés képi információ van a képen, azaz kevés alakzat, vagy kevés szín, (ilyen esetek általában ha egyszínű háttéren vagy a könyv címe és szöveges információk vannak) általában a szövegfelismerő modul segítségével megfelelően detektálja a címet, tehát ilyen esetekben ez a modul nem is kerül használatra, mivel ennek a modulnak a célja csupán egy másodlagos keresés, ha a szövegfelismerő modul nem jár sikerrel. Tehát ez a modul általában olyan képeket kap bemenetül, amelyek gazdagabb kép világgal rendelkező könyv borítókat ábrázol, tehát olyan képeket, amelyeken több szín jelenik meg, innen fogva adott, hogy érdekesebb a szín alapú egyezés keresés az adatbázissal.

Az irodalomkutatásban (3.2.2.2 fejezet) bemutatott módszerek általánosságban valamilyen hisztogramot készítettek a képről és e hisztogramokat hasonlították össze az adatbázisban szereplő képek hisztogramjaival.

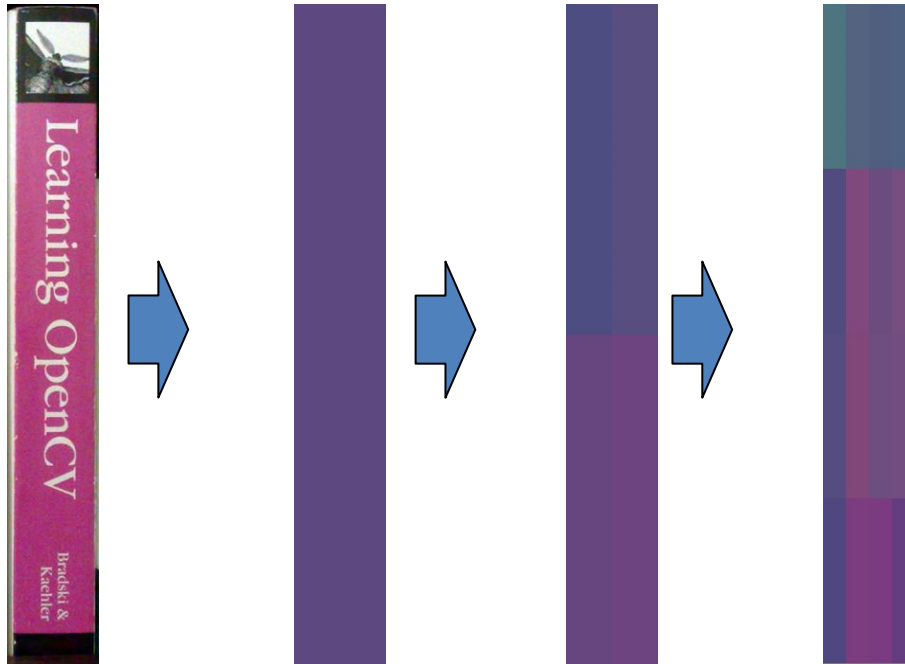


21. ábra: A és B kép, illetve e képek hisztogramjai

Ez a megoldás gyors mivel viszonylag kevés információt kell összehasonlítani és pontos is, ha nagyon különböző hisztogramú képek szerepelnek az adatbázisunkban, de ezen projektnél semmi sem garantálja, hogy nagyon különböző hisztogramú képek szerepeljenek csak az adatbázisunkban, illetve a hisztogram alapú színtársításnak egy másik nagy hátránya, hogy a hisztogram nem ad információt arról, hogy az adott színek, a kép melyik részén szerepelnek. Például a 21. ábrán látható, hogy két teljesen különböző (egyik pixelük sem egyező) kép esetén is teljes mértékben megegyezhet a képek hisztogramja. Ezen okok miatt ez a módszer nem került implementálásra.

Egy olyan módszert fejlesztettünk ki, amely megfelelően gyors és figyelembe veszi, a hisztogrammal ellentétbe azt is, hogy az adott szín a könyvborító, melyik részén helyezkedik

el. A könyv adatbázisunkban nem a könyvről készült hisztogramot tároljuk el, hanem egy képpiramist. Azaz a legfelső szinten a teljes kép színeinek átlagát tároljuk, az alatti szinten már 4 részre osztva a képet azokban az egyes szegmensekben található színek átlagát és így tovább a felosztást. Erre egy példa a 22. ábrán látható.



22. ábra: Az eredeti könyvgerincről készült kép és az abból generált Gauss piramis

Mivel a képek színek esetében különösen érzékenyek az esetleges szín visszaverődésekre RGB szintér helyett HSV szintérbe konvertáltuk a színeket és ott a V csatorna elhagyásával kiküszöböljük az esetleges ilyen visszaverődéseket.

Ez a modul először a bemeneti kép saját képpiramisát határozza meg, majd a legmagasabb szintű piramis elemet összehasonlítja az adatbázis összes legmagasabb szintű piramis elemével és minden elemmel eltérést számít. Az eltérést úgy határozza meg:

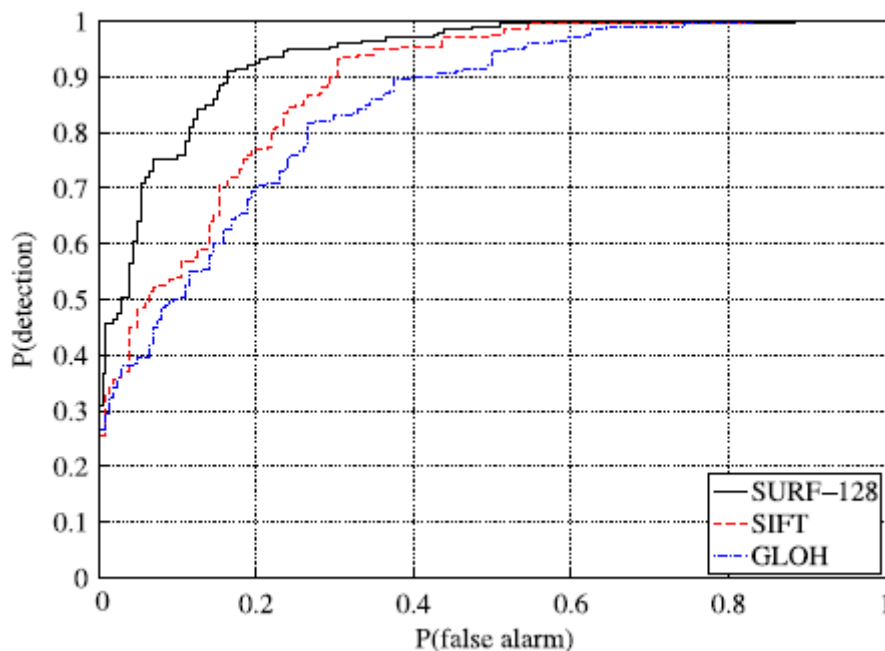
$$\text{eltérés} = |H^{\text{adatbázis elem}} - H^{\text{Bementi kép}}| + |S^{\text{adatbázis elem}} - S^{\text{Bementi kép}}|$$

Ahol H és S az adott Hue és Saturation csatornák értékei. Ezt követően eltérési érték szerint csökkenő sorrendbe listázza az adatbázis elemeket, majd az 50 legkisebb eltéréssel rendelkező elemmel újra eltérést számol a következő képpiramis szint szerint, majd ezt megismétli az utolsó szinten is, de már csak a legjobb 10 képpel, és amelyik kép ezek közül a legjobb értékű, az kerül kiválasztásra. A szintenkénti összehasonlítás szám miatti számolási igény megnövekedése következtében kerül folyamatosan csökkentésre az egyes szinteken az adatbázisból hasonlított elemek száma.

4.3.3 Mintaillesztés alapú cím felismerés

Mintaillesztés alapú megoldás esetén minden képen meghatározunk kitüntetett pontokat, majd társítjuk őket. Ez két különböző lépés, azonban minden kitüntetett ponttársító algoritmusnak megvan a legmegfelelőbb kitüntetett pontmeghatározó algoritmus „párja”, ezért a két lépés egyben kerül meghatározásra.

Az irodalomban [8] összehasonlításra kerültek a már bemutatott algoritmusok, az eredmények a 11. ábráról olvashatók le. Az eredmények annál jobbak minél jobban közelíti a függőleges 1 és vízszintes 0 értéket. Az értékek azért vonalként jelennek meg, mert az adott algoritmusok különböző paraméterekkel kerültek mérésre.



23. ábra: A különböző referenciapont-társító algoritmusok pontossága [8]

A 23. ábráról egyértelműen megállapítható, hogy minden paraméter esetén a SURF algoritmus érte el a legjobb pontosságot, azonban a megvalósított rendszerünk egy szín alapú keresést fog használni.

4.3.3.1 Színterek

A különböző színterek befolyásolják a mintaillesztés pontosságát is, ugyanarról a könyvről más megvilágításban készült képek összehasonlítása nehézkes lehet, mivel a fényintenzitás befolyásolja a színek élénkségét. Megoldás lehet a HSV színtérbe való konverzió, majd ott a V (fény intenzitás) csatorna elhagyása [14].

4.4 Adatbázis keresés

Kétmegoldás lehetséges a problémára. Online és offline megközelítés. Offline megközelítés esetén minden adat a programmal együtt a felhasználó eszközén kerül eltárolásra, online tárolási elv esetén egy központi adatbázisba kerülnek az adatok eltárolásra.

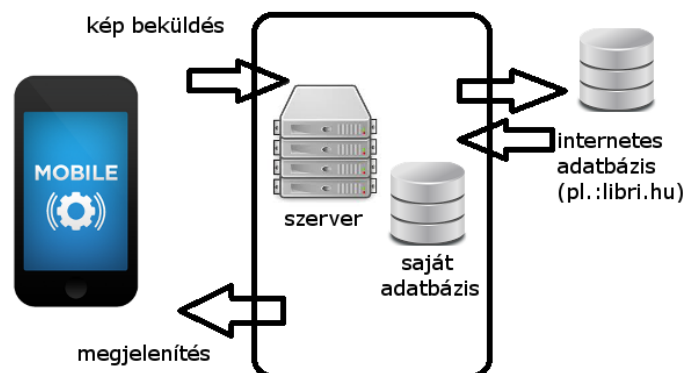
Az offline tárolás esetén gyorsabb az adatokhoz való hozzáférés (nem függ az internet elérésétől), de sokkal nagyobb a szoftver helyigénye, mivel az adatbázis is a felhasználó eszközén kerül eltárolásra, ezen kívül mivel az egész feldolgozási folyamat az eszközön hajtódik végre ez mobiltelefon vagy gyengébb teljesítményű eszköz alapú megvalósítások esetén nem túl energia-gazdaságos megoldás.

Ennél jobb megoldást jelent, ha az adatbázis egy szerveren kerül eltárolásra, a kliens oldal a könyvekről készült képet beküldi a szervernek, majd onnantól minden feldolgozási lépést a szerver végez (könyvdetektálás, cím felismerés, adatbázis keresés) és az eredményeket visszaküldi a kliensnek, ami azt megjeleníti a felhasználó számára.

Ezen megoldás bár internet kapcsolatot igényel, nem veszi olyan mértékben igénybe a kliens oldali eszköz erőforrásait, mint az offline. Az adatbázis folyamatos karbantartása sem okoz így problémát, nem alakulhat ki inkonzisztencia frissítések esetén. Továbbá egyértelműen belátható, hogy egy szerver teljesítménye nagyságrendekkel meghaladja egy mobiltelefonét, így bár az internet szükségessége lassítja a folyamatot, a nagyobb teljesítményből adódó gyorsabb feldolgozás „visszahozhatja” ezt az időt.

Továbbá a hasonló projektek [2, 3, 4] mind mobil eszközre korlátozzák a rendszert, holott ez a korlátozás felesleges. Mivel ezen megoldás esetén, szerveren történik a teljes feldolgozási folyamat, csupán egy kép érkezik a kliensről és a szerver szövegesen a megfelelő információk küldi el a klienshez, ezért a kliens bármilyen internettel rendelkező képalkotásra alkalmas eszköz lehet (mobiltelefon, PDA, webkamerával rendelkező laptop, vagy egy sima képfeltöltési lehetőséggel ellátott honlap).

A fent említett okok miatt a megvalósított rendszer egy ilyen szerver alapú megközelítést használ. Kiegészítve egy külső, internetről származó adatforrással, a rendszer elsődlegesen a saját adatbázisában keresne, majd ha itt nem talál semmi egyezést online könyvtárakban, könyvtárak internetes adatbázisában keres, ezzel is javítva a teljesen sikertelen találatok arányán. A rendszer kapcsolatai a külvilággal az 24. ábrán látható.



24. ábra: A rendszer kapcsolata a külvilággal

A jelenleg PC-n megvalósított rendszer egy bármilyen inputról érkező képet képes megnyitni, legyen az a helyi számítógép vagy hálózati meghajtú, vagy a számítógéppel összekötött mobil

eszköz. A feldolgozás megkezdése előtt kiválasztható a kép, és ha az valamilyen formában rendelkezésre áll a gép számára, akkor a program képes a feldolgozásra. Mobilos alrendszerek még nem állnak rendelkezésre, mivel a dolgozat elsődleges célja a feldolgozás hatásfokának javítása. Ugyancsak igaz ez a külső internetes adatbázisokra, ha megfelelő formátumba rendelkezésre állnak a program számára, képes használni azokat, azonban jelenleg nem megfelelő formátumu adatok konvertálását nem képes elvégezni.

5 Teszteredmények és értékelésük

Ezen fejezetben a már elkészült rendszer tesztjét mutatom be. Ismertetem a tesztek körülményeit, eredményeit majd értékelem azokat. A tesztek során a következő elkészült egységek működését vizsgálom: könyv elforgatás, könyv éldetektálás FSA [9], illetve Hough transzformáció [7] módszerrel, karakterfelismerés előfeldolgozás nélkül, karakterfelismerés elő- és utófeldolgozással, ennek a modulnak egyes részmoduljai is tesztelésre kerültek külön (binarizáció, utófeldolgozás), szín alapú felismerés. Illetve vizsgálom és értékelem az egész rendszer működését és eredményeit a kezdő képtől a végső könyv adatokig. Ezen kívül a rendszer hatásfoka összevetésre került a piacon felelhető OCR programokkal is.

A legpontosabb összehasonlítás érdekében a teszteket ugyanazon a képeken lenne a legjobb elvégezni, mint amelyeken a hasonló projektek is elvégezték a saját implementációjuknak a tesztjét. A [3] irodalomban jegyzett megvalósítás tesztjéhez használt könyv adatbázist a szerzők megkeresésünk után rendelkezésünkre bocsájtották és a teszteket ezen adatbázis alapján végezzük. Ez az adatbázis 33 olyan képet tartalmaz, amelyeken egészben látszódik a könyvespolc egy sora, illetve 146 előre feldarabolt (azaz csak egy könyvgerincet ábrázoló) és felcímkézett (szöveges formában is ismert a képen található könyv címe és szerzője) képet tartalmaz, ezen kívül 2000 darab előre feldarabolt, de nem felcímkézett képet is tartalmaz.

Sajnos néhány teljesen saját fejlesztésű modulnál (könyv elforgatás, szín alapú keresés) nem megfelelőek a tesztekre ezek a tesztadatok, emiatt saját tesztadatbázist kellett összeállítani, amelyhez a képek főként az egyetem könyvtárában, illetve az otthoni könyvespolcra készültek. A tesztadatbázis 22 képet tartalmaz, amelyen összesen 108 könyv található meg. Mivel a modulok, amelyek teszteléséhez felhasználásra kerültek ezek a képek a hasonló rendszereknél nem szerepelnek, ezért az eltérő tesztadatokból való esetleges eltérés a teszt eredményekben nem lehetséges (mivel egyáltalán ez a modul nem is létezik más hasonló rendszereknél).

5.1 Könyv elforgatás

Első esetben a könyv elforgatás tesztelésére került sor, azaz milyen pontossággal képes felismerni a rendszer, hogy a képen a könyvek tájolása függőleges vagy vízszintes, azaz egymás mellette állatva vagy egymáson fektetve helyezkednek el a könyvek.

Egy tesztnek három lehetséges kimenetele lehet, beszélhetünk sikeres elforgatásról, ha a kép vízszintes tájolású és elforgatásra került, vagy ha a kép függőleges tájolású és nem került

elforgatásra, fals pozitívról, ha annak ellenére, hogy függőleges tájolású a kép elforgatásra került, illetve fals negatívról, ha az elforgatás szükséges lett volna, de ezt nem tette meg a rendszer. Az eredmények a 3. táblázatról olvashatók le.

teszt adatok száma	sikeres elforgatás	fals negatív	fals pozitív
22 (ebből 8 fekvő) (100%)	19 (ebből 6 fekvő) (86,36%)	2 (9%)	1 (4,5%)

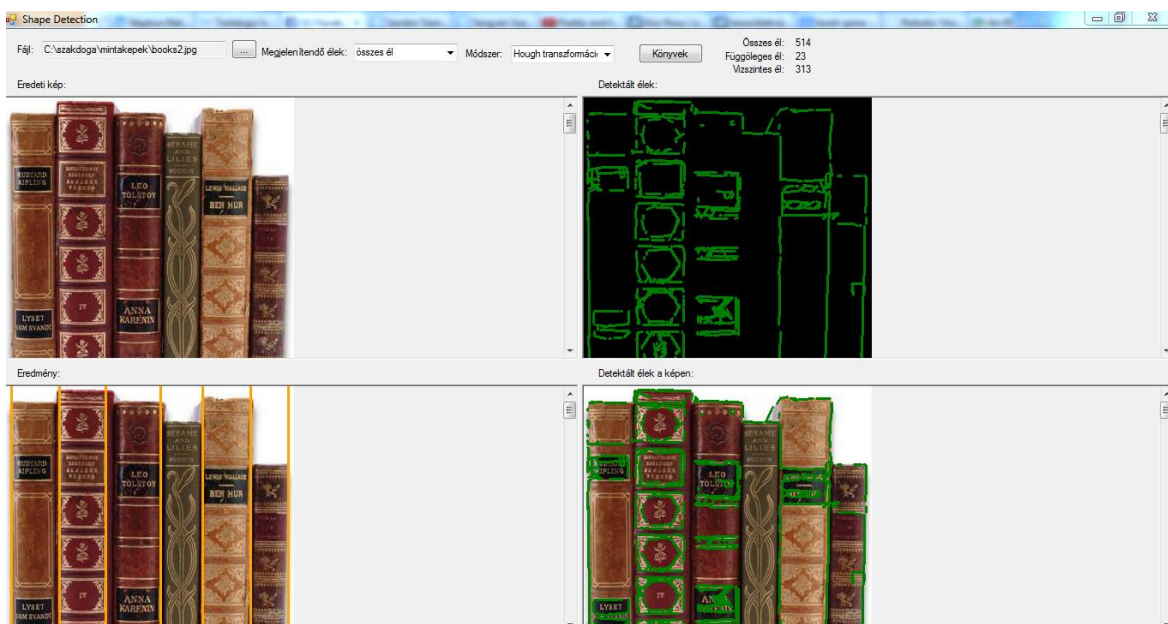
3. táblázat: könyv elforgatás modul pontossága

5.1.1 Értékelés

Sajnos e modul összehasonlítása a hasonló projektekkel nem lehetséges, mivel azok nem rendelkeztek ilyen lépéssel. Azonban, ha csupán az alapján végezzük az értékelést, hogy mi a modul célja (a könyv éldetektáló modul csak függőleges helyzetben detektál, tehát a vízszintes képek felismerése és elforgatása), elmondható hogy a modul 86,36%-ban, csak a fekvő képek esetén 75%-ban javít a teljes rendszer működésén. Az esetek 9%-ában nem forgatja el a képet, holott azt el kellene, ez esetben se nem ront, se nem javít a teljes rendszer működésén, mivel e modul nélkül sem történne meg az elforgatás. Az esetek 4,5%-ában viszont ront a teljes rendszerre nézve a pontosságon. Ha ezt a számot összevetjük a 86,36%-al egyértelműen belátható, hogy ez a modul összességében javít a rendszeren.

5.2 Könyvdetektálás

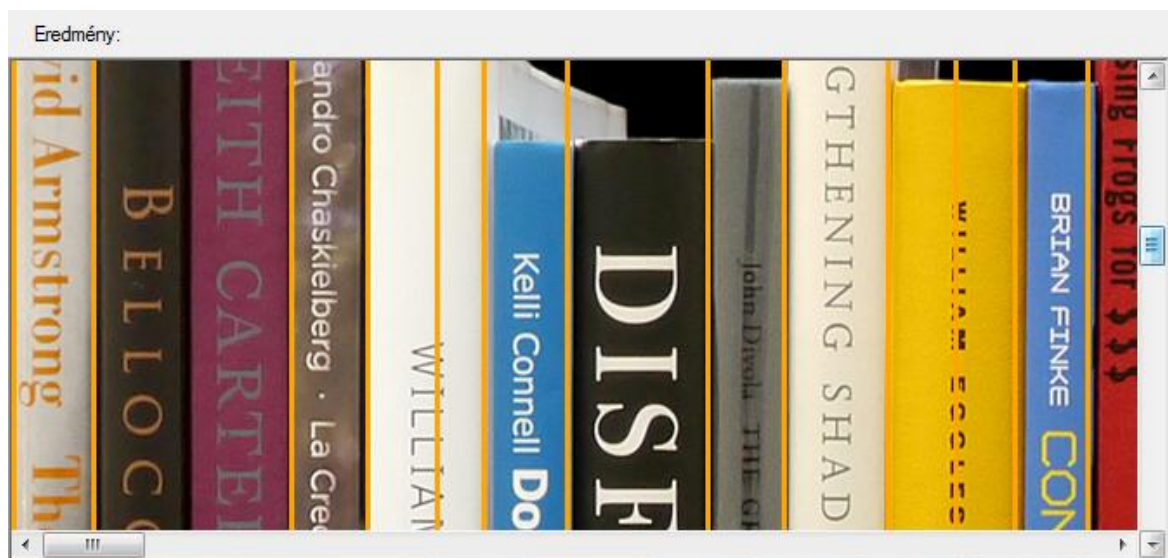
Ezen fejezetben a két könyvdetektáló módszer saját implementációjának eredménye kerül tesztelésre, összehasonlításra és értékelésre. Mindkét módszer célja, hogy a függőlegesen egymás mellett álló könyveket elszeparálja a köztük lévő élek alapján. A teszt során ezen élek megtalálásának pontossága került mérésre. A tesztkörnyezet a 25. ábrán látható.



25. ábra: A teszteléshez használt környezet

Ezen teszt esetén is három eredményt különböztethetünk meg: pontos találat, amikor a képen egy könyv széle található és ezt jelzi is a rendszer, fals negatív, ha egy könyvélt nem jelez a

rendszer, és fals pozitív, ha könyvélt jelez a rendszer, holott ott nem található. A 26. ábrán látható egy tesztminta, amin a detektált éleket sárga csík jelöli, az ábrán jól megfigyelhető mind a három tesztet (sikeres, fals pozitív, fals negatív).



26. ábra: Minta a tesztek lehetséges kimeneteleire

5.2.1 Könyvdetektálás – Hough transzformáció

A tesztet a saját könyvadatbázison végeztük, az előző fejezetben leírtak (5.2 fejezet) szerint került sor a mérésre. Az eredmények a 4. és 5. táblázatban tekinthetők meg.

tesztadat neve	összes élék száma	sikeresen detektált élék száma	fals pozitív találatok	fals negatív találatok
könyvespolc #1	7	7	0	0
könyvespolc #2	25	22	9	3
könyvespolc #3	62	51	6	11
könyvespolc #4	23	21	12	2

4. táblázat: az egyes tesztképeken történő detektálás pontossága

tesztadatok száma	sikeres éldetektálás	fals pozitív	fals negatív
117 él	101 (74 %)	27 (20 %)	16 (11 %)

5. táblázat: Hough transzformáció alapú könyvdetektálás mérési eredményei.

Az 5. táblázatban szereplő százalékos adatok az adott eredmény és az összes detektált eredmény hányadosaként kerültek meghatározásra (hiába összesen 117 élt tartalmaztak a képek, a fals pozitív találatok miatt 135 éllel kerültek kiszámításra a százalékos adatok).

5.2.1.1Értékelés

Az irodalomban jegyzett [3] 72%-os hatásfokot ez az implementáció véleményem szerint jól közelítette. Bár 2% eltérés van a mért és az irodalomban jegyzett érték között, fontos megjegyezni, hogy a valós és mért hatásfok közötti 2 % eltérés hibahatáron belüli.

5.2.2 Könyvdetektálás – FSA

A megvalósított FSA alapú rendszer tesztelésére is az előzőleg (4.2 fejezet) leírtak szerint került sor. Fontos megjegyezni, hogy a tesztadatok pontosan megegyeztek a Hough transzformáció alapú megoldás mérésekor használt tesztadatokkal. Ez azért fontos, mert így realisabb a két módszer hatásfokának összehasonlítása. A mérési eredmények a 6. és 7. táblázatban tekinthetők meg.

tesztadat neve	összes élek száma	sikeresen detektált élek száma	fals pozitív találatok	fals negatív találatok
könyvespolc #1	7	7	0	0
könyvespolc #2	25	24	2	1
könyvespolc #3	62	52	2	10
könyvespolc #4	23	19	13	4

6. táblázat: az egyes tesztképeken történő detektálás pontossága

tesztadatok száma	sikeres éldetektálás	fals pozitív	fals negatív
117 él	102 (76 %)	17 (12 %)	15 (12 %)

7. táblázat: FSA alapú könyvdetektálás mérési eredményei.

A 6. táblázatban szereplő százalékos adatok az adott eredmény és az összes detektált eredmény hányadosaként kerültek meghatározásra (hiába összesen 117 élt tartalmaztak a képek, a fals pozitív találatok miatt 134 éllel kerültek kiszámításra a százalékos adatok).

5.2.2.1Értékelés

Az irodalomban jegyzett [9] közel 90%-os pontossághoz és a 3,2%-os fals pozitív arányhoz képest sajnos a megvalósított rendszer csupán 76 %-os pontosságot ért el.

5.2.3 Összehasonlítás

Mivel az FSA alapú és a Hough transzformációt alkalmazó megvalósítás is ugyanazt a célt szolgálja (szegmentációs helyek meghatározása a képen) és ugyanazon képekkel végeztük el a teszteket, lehetséges a két modul hatásfokának összehasonlítása.

A mérések alapján az irodalomban jegyzet értékekhez hasonló (részletekért lásd 5.2 fejezet) eredményekre jutottunk, az FSA alapú módszer ért el jobb eredményt a sikeres detektálásokat

illetően. A fals pozitívokra nézve az irodalomban jegyzett tesztekkel hasonló eredmény született, azaz az FSA alapú megoldás ért el jobb eredményt. Tehát összességében elmondható, hogy a FSA alapú módszer alkalmazása helyes döntés volt. Jobb eredményeket ért el a Hough transzformációnál.

5.3 Karakterfelismerés előfeldolgozás nélkül

Ezen fejezetben a megvalósított előfeldolgozás nélküli karakterfelismerő modul eredményeit ismertetem. E modul egy szegmentált képet kap bemenetül (egy könyvgerincet ábrázoló képet) majd azon végez karakterfelismerést.

A rendszer pontosságát az alapján mértük, hogy a könyvgerincen található karakterek közül hány karaktert ismer fel pontosan a modul. A mérés során 3 eredmény került megkülönböztetésre. Sikeres karakterfelismerés, ha a modul sikeresen felismerte a képen található karaktert, fals pozitív, ha karaktert detektált, holott nem kellett volna és fals negatív állapotot, ha karaktert kellett volna felismernie, de ezt nem tette. Az eredmények a 7. és 8. táblázatról olvashatók le.

tesztadat neve	összes karakter száma	sikeresen felismert karakterek száma	fals pozitív találatok	fals negatív találatok
könyvgerinc #1	39	15	1	24
könyvgerinc #2	18	10	5	8
könyvgerinc #3	19	3	0	16
könyvgerinc #4	20	1	1	19
könyvgerinc #5	29	4	2	25

8. táblázat: az egyes tesztképeken történő felismerés pontossága

tesztadatok száma	sikeres karakterfelismerés	fals pozitív	fals negatív
125 karakter	33 (25,1%)	9 (6,8%)	92 (70,2%)

9. táblázat: előfeldolgozás nélküli karakterfelismerés mérési eredményei.

Megjegyezendő, hogy a könyvgerinc #1 tesztadat egy hibásan szegmentált könyvgerinc volt, azaz valójában két gerinc található azon a képen, ez indokolja a magas karakterszámot is (a teljes körű teszt érdekében szándékosan ilyen tesztadat is alkalmazásra került, annak tesztelése érdekében, hogy egy modul esetleges hibáit az azt követő modul mennyire tudja kezelni).

A 8. táblázatban szereplő százalékos adatok az adott eredmény és az összes detektált eredmény hányadosaként kerültek meghatározása (hiába összesen 125 karaktert tartalmaztak

a képek, a fals pozitív találatok miatt 134 karakterrel kerültek kiszámításra a százalékos adatok).

5.3.1 Értékelés

Az irodalomkutatás alapján (részletek a 3.2.2.2 fejezetben) várható gyenge teljesítményt hozott az előfeldolgozás nélküli karakterfelismerés. Bár a mért 25,1%-os pontosság sokkal jobb az irodalomban jegyzett [4] közel 10%-osnál, azonban viszonylag kevés adaton került végrehajtásra a teszt, elég magas szórást feltételezhetünk a mérést illetően. Azonban mind az irodalom, mind saját tesztünk alapján arra a megállapításra juthatunk, hogy magában, előfeldolgozás nélkül túlságosan pontatlan a karakterfelismerő modul és jövőben szükséges e rendszer valamilyen előfeldolgozással való támogatása.

5.4 Karakterfelismerés előfeldolgozással

Mivel az előző fejezetben bemutatott eredmények alapján jól látható, hogy előfeldolgozás nélkül gyakorlatilag használhatatlan a karakterfelismerés, ezért különböző elő- és utófeldolgozó modulok kerültek fejlesztésre, amelyek javítják a karakterfelismerés hatásfokát, ezen fejezet ezeknek a moduloknak külön-külön illetve a karakterfelismerő egységnek a teszteredményeit mutatja be és értékeli.

5.4.1 Binarizáció

A binarizáció célja, hogy megfelelően előkészítse a képet a karakterfelismerő (OCR) engine számára. Kiemeli a kép fontos részeit, azokat, amelyek vélhetően betűk. Ezt a részmodult sajnos magában nem lehetséges tesztelni, mivel egy képről lehetetlen megmondani az ember számára, hogy az mennyire optimális az OCR számára. Ezért ez a modul a karakterfelismerő modullal együtt került tesztelésre.

Kétfajta eredménye lehetett a teszteknek, sikeres illetve sikertelen. Sikeres esetben a binarizált kép alapján előállt szöveg pontosan tartalmazza a könyvcímet. Azaz akkor is sikeresnek vettünk egy binarizációt, ha a könyvcímen kívül más zajokat is felismert, mivel ezen a későbbi modulok még javítani tudnak. Sikertelen eredmény akkor született, ha a felismert szöveg nem tartalmazta vagy csak részben a könyven található címet.

Tesztesetek száma	Sikertelen binarizáció	Sikeres binarizáció
20	3 (15 %)	17 (85 %)

10. táblázat: A binarizáció eredménye

5.4.1.1 Értékelés

Sajnos ilyen részletesen az irodalomban nem tértek ki az általuk megvalósított előfeldolgozások hatásfokára, ezért azokkal nem lehet összevetni az eredményt. Azonban összevetési alapnak tekinthető az előfeldolgozás nélküli karakterfelismerés, amely a 25%-os pontosságával, nagyságrendekkel gyengébb eredményeket mutat az előfeldolgozással rendelkező karakterfelismerőhöz képest. Bár ez a modul még csak az előfeldolgozást végzi, de

már ennek a használatával is nagyságrendekkel való javulást értünk el a karakterfelismerés terén.

5.4.2 Utófeldolgozás

Az utófeldolgozás célja, hogy a megkapott szöveges adatból egy szótár segítségével meghatározza a pontos könyv címet. Ez a modul a tesztek során mindig olyan adatot kapott bemenetül, amely a binarizáció után tartalmazott könyvcímet, tehát az input adatok alapján minden esetben sikeresen felismerhető volt a szövegből a könyvcím. Az eredmények a 11. táblázatban láthatók.

Tesztesetek száma	Sikertelen utófeldolgozás	Sikeres utófeldolgozás
16	1 (7 %)	15 (93 %)

11. táblázat: A binarizáció eredménye

Sikeres akkor volt az utófeldolgozás, hogy ha a bemeneti szöveg alapján a megfelelő könyvcímet adta meg, immár zaj mentesen. Sikertelen, ha viszont valamely másik adattal párosította a szótár állományából.

5.4.2.1Értékelés

Jól látható az eredmények alapján, hogyha a binarizáció megfelelő volt és sikerült a könyvcímet, ha még zajosan is, de felismernie az OCR-nek, akkor ez a modul már szinte hibátlanul „zaj mentesítette” a szöveget. Erről tanúskodik a 93%-os pontossági adat is.

5.4.3 Karakterfelismerés modul

A karakterfelismerés modul. tehát egyben a binarizáció, orientáció meghatározás, karakterfelismerés és utófeldolgozás egy szegmentált képet kap bemenetül, amelyen egy könyv gerince látszódik és egy könyvcímet ad kimenetül.

Ebben az esetben is két különböző kimenetele lehetett a tesztnek, sikeres karakterfelismerés vagy sikertelen karakterfelismerés. Sikeres karakterfelismerés esetén, a képen lévő könyv címét adta vissza a rendszer, sikertelen esetben nem. Az eredmények a 12. táblázatban láthatók.

Tesztesetek száma	Sikertelen karakterfelismerés	Sikeres karakterfelismerés
20	5 (25 %)	15 (75%)

12. táblázat: A karakterfelismerés eredménye

5.4.3.1Értékelés

Modul szinten már a hasonló rendszerek is határoztak meg hatásfokokat, az erre a célra használt mintaillesztéses felismerés a különböző hasonló rendszerek esetében 70-85%-ot ért el, míg a karakterfelismerés 65-70%-os hatásfokot ért el. Tehát az irodalomban említett hatásfokot eltérte és túlszárnyalta a saját rendszerünk, bár szótár alapú a felismerés és ezért léphette azt túl. De ez az eredmény alapján kijelenthető, hogy esetünkben egy szöveg alapú felismerés is elérheti egy mintaillesztés alapú módszer hatásfokát.

5.5 Színfelismerés

Ez a modul a képen lévő színek alapján keres egyezést a könyvgerinc adatbázisban. Arra szolgál, hogyha a szöveg felismerő modul nem hoz eredményt, akkor a meglévő képadatbázis alapján, szín alapján keresést végez, és a legkevésbé eltérő adatbázis elemhez tartozó címet és szerzőt visszaadja.

A teszteknek itt is 2 lehetséges eredménye lehetett, Sikeres esetben a megfelelő rekordot sikerült kiválasztani az adatbázisból, sikertelen esetében egy nem megfelelő rekord került kiválasztásra. A tesztadatok minden esetben olyan képek voltak, amelyeket előzőleg a karakterfelismerés modulnak nem sikerült helyesen felismernie. Az eredmények a 13. táblázatból olvashatók le.

Tesztesetek száma	Sikertelen színfelismerés	Sikeres színfelismerés
15	4 (27%)	11 (73%)

13. táblázat: A karakterfelismerés eredménye

5.5.1 Értékelés

Ebben az esetben sem lehetséges a modul és az irodalomkutatás fejezetben bemutatott rendszerekben lévő modulok összevetése, mivel a hasonló rendszerek nem rendelkeznek szín alapú adatbázis-kereső modullal. Mivel ez a rendszer a tesztek során csak olyan tesztadattal lett tesztelve, amelyeket a karakterfelismerő modul nem ismert fel, a 73%-os sikeres színfelismerési arány kimagaslóan jó.

5.6 Teljes rendszer

A teljes rendszer tesztjére úgy került sor, hogy egy, a könyvespolc egysorán elhelyezkedő könyvekről készült képet kapott bemenetül és a kimenetnek a képen lévő könyvek címeinek szöveges formátumában kellett lennie. Sikeresnek számított a felismerés, ha a felismert cím valóban szerepelt a képen. Sikertelennek, ha a képen szereplő könyv nem került felismerése vagy hibás cím jelent meg helyette. Az eredmények a 14. táblázatban láthatók.

Tesztesetek száma	Sikertelen	Sikeres
30	12 (40%)	18 (60%)

14. táblázat: A teljes rendszer eredménye

5.6.1 Összehasonlítás

Ezen fejezetben két már piacon kapható OCR program hatásfokát mutatom be ugyanazon tesztadatok esetén, amelyeket a saját rendszerünk esetén is alkalmaztunk. Ezek a programok általános célú OCR programok, azaz nem kifejezetten könyvgerincek feldolgozására készültek. Sajnos jelenleg a piacon nincs jelen semmi olyan alkalmazás, amely kifejezetten az általam bemutatott problémára készültek volna, az irodalomkutatásban bemutatott rendszerek kísérletű jellegűek, forgalomban nem elérhetők, így azokkal nem lehetséges összehasonlító teszt végzése. Ezért döntöttem az általános szöveg-digitalizáló programok mellett, mivel jelenleg a piacon ezek állnak legközelebb az általunk fejlesztett alkalmazáshoz.

Az egyes felismeréseket négy kategóriába osztottam: sikertelen, orientáció felismerés, sikeres, tökéletes. Tökéletes esetben a könyvön lévő cím összes karakterét sikerült felismerni. Sikeres esetén sikerült felismerni a karaktereket, de kisebb hibával, amin utófeldolgozással még javítani lehet. Orientáció felismerés esetén, bár nem sikerült pontosan felismerni a címet, de a képen lévő szöveges elemeket sikerült meghatározni. Sikertelen esetben semmilyen szöveges információt nem sikerült kinyerni a képről.

5.6.1.1 OmniPage

Először a Nuance OmniPage nevű szoftverében végeztem teszteket. Ez a rendszer meglepően gyengén szerepelt sok esetben a szöveg irányultságát sem sikerült felismernie.

Tesztesetek száma	Sikertelen felismerés	Orientáció felismerés	Sikeres felismerés	Tökéletes felismerés
15	9 (60 %)	1 (6,66 %)	2 (13,33 %)	3 (20%)

15. táblázat: Az OmniPage hatásfoka

A 15. táblázatból jól látható, hogy csupán az esetek 33 %-ában sikerült valamilyen információt kinyerni a könyvgerincekről, ami rendkívül gyenge eredmény és messze alul múlja a saját fejlesztésű rendszerünket.

5.6.2 ABBYY

A másik jelentős könyv-digitalizáló eszköz, amely tesztelésre került az ABBYY Fine Reader 11 volt. Ez a program már messze jobb eredményeket ért el erre a feladatkörre.

Tesztesetek száma	Sikertelen felismerés	Orientáció felismerés	Sikeres felismerés	Tökéletes felismerés
15	1 (6,66 %)	1 (6,66 %)	3 (20%)	10 (66%)

16. táblázat: Az ABBYY Fine Reader 11 hatásfoka

A 16. táblázatból jól látható, hogy összesen 80 %-ban sikerült megfelelően felismerni.

5.6.3 Értékelés

A teljes rendszert tekintve 60%-os pontosságot értünk el. Azonban egyes megvalósított moduljaink kiemelkedő 80-90%-os pontosságot értek el. Sajnos sok esetben a könyvszegmentáló modul hibájából csökkent az alkalmazás hatásfoka. A hasonló irodalomkutatás részben kifejtett rendszerek 70% körüli eredményt értek el, azonban azok nem szöveg alapú felismerést használtak, hanem mintaillesztés alapján adatbázishoz párosítottak. Így nagy eredménynek értékelem, hogy saját a szöveg alapú módszerünk megközelítette azt.

A hasonló már forgalomba lévő szoftverek tesztelésére már előre szegmentált képeket használtunk, mivel azok nem rendelkeznek könyvszegmentáló modullal, jóval gyengébb hatásfokot értek volna el. Ha a saját szegmentálás utáni eredményünket nézzük, akkor 70%-os pontosságról beszélhetünk, amivel az ilyen típusú input adatoknál megelőztük az OmniPage nevű programot. Azonban sajnos az ABBYY Fine Reader 11-hez képes még van fejlődni valónk.

6 Összefoglalás

Dolgozatomban ismertettem a könyvkategorizálásra jelenleg alkalmazott megoldásokat, majd bemutattam a hasonló rendszereket, ezek értékelése után három nagy hiányosságot határoztam meg a már meglévő rendszerekkel szemben. A könyvfelismerő modul szabadságfokának növelése, ne csak a gerincről függőleges helyzetben működjön a detektálás, hanem akár a könyv elejéről vagy hátlapjáról tetszőlegesen elforgatva. A cím felismerő működéséhez ne legyen elengedhetetlenül szükséges egy képadatbázis. A könyvekről való információ kinyerés ne csak saját belső adatbázisból történjen, hanem az interneten elérhető források is felhasználásra kerüljenek.

Az előző félévben a saját rendszer tervezésénél a könyvdetektáló modul esetében saját mérések és az irodalom alapján meghatároztam a legmegfelelőbbnek tűnő megoldást, illetve javítottam az irodalomban szereplő rendszerekhez képest a szabadságfokon (már vízszintes tájolású gerincek felismerését is elvégzi a rendszer).

A cím felismerés modul esetében bemutattam a lehetséges megvalósításokat és meghatároztam egy lehetséges implementációt. Ezen félévben ennél a modulnál történt a legtöbb előrelépés, meghatározásra kerültek a szükséges részmodulok, azok feladatai és konkrét implementáció, egyes esetekben a kezdeti algoritmusok már fejlesztése is kerültek a jobb hatásfok érdekében.

Ezen kívül meghatározásra és implementálásra került egy szín alapú adatbázis keresés modul, ami megfelelő „segítséget” nyújt a cím felismerő modul számára. Ez a modul egy kész adatbázissal hasonlítja a könyvgerinceket és a színek alapján határozza meg az egyezést.

Az adatbázis-keresés modulnál meghatároztam és indokoltam az implementációs tervet.

A legfontosabb jövőbeli továbbfejlesztési iránynak a cím felismerő modul fejlesztését tartom, jelenleg elfogadható hatásfokkal működik, azonban meglehetősen időigényes. Olyan módszerek megtalálása a továbbfejlesztés célja, amelyek nem csökkentik a modul hatásfokát, ugyanakkor nagyságrendekkel csökkentik a futási idejét.

Összességében egy megfelelően működő alkalmazást hoztam létre, amely elfogadható hatásfokkal képes egy bemeneti képről meghatározni a képen látható könyvek címét. Innen már csak egy Amazon vagy valamilyen nagyobb könyvtár API-jának meghívás a cím alapján (amely nagyobb tudományos jelentőségű újítást nem hordoz magában), és az alapján további adatok szolgáltatása a felhasználó felé hiányzik, és egy irodalomban szereplő rendszerekkel is összevetve megfelelő rendszer valósul meg.

A dolgozatban bemutatott módszerek eredményei publikálásra kerültek a XIX: Fiatal Műszakiak Tavaszi Ülésszakán 2014. márciusában [32] és az IEEE CINTI 2014 konferencián.

7 Ábrajegyzék

#	Megnevezés	Forrás
1. ábra	Mobile Phone Book Finder rendszervázlata	[2]
2. ábra	Building Book Inventories using Smartphones rendszervázlata	[3]
3. ábra	A hibrid megoldás rendszervázlata	[4]
4. ábra	A detektált élek	[3]
5. ábra	Szakaszonként végighaladás a képeken	[9]
6. ábra	Az FSA állapot gráfja	[9]
7. ábra	A Watershed algoritmus eredménye könyvek esetében	-
8. ábra	Referenciapont illesztés könyvek esetén	[14]
9. ábra	A képek és azok hisztó- illetve korreogramjaik	[31]
10. ábra	OCR hatékonysága előfeldolgozás nélkül	[3]
11. ábra	A rendszer vázlata	-
12. ábra	A különböző könyvszegmentáló technikák hatásfoka	[9]
13. ábra	Eredeti méretű kép a detektált könyvszélekkel	-
14. ábra	Átméretezett kép a detektált könyvszélekkel	-
15. ábra	Eredeti kép a detektált könyvszélekkel	-
16. ábra	Módosított kép a detektált könyvszélekkel	-
17. ábra	Detektált élek LSD módszerrel	-
18. ábra	Az átméretezésen és morfológiai műveleteken átesett képen LSD módszerrel detektált élek	-
19. ábra	A szövegalapú felismerés modul almoduljai	-
20. ábra	A különböző binarizációs algoritmusok és hatásfokuk	-
21. ábra	A és B kép, illetve e képek hisztogramjaik	-
22. ábra	Az eredeti könyvgerincről készült kép és az abból generált Gauss piramis	-
23. ábra	A különböző referenciapont-társító algoritmusok pontossága	[8]
24. ábra	A rendszer kapcsolata a külvilággal	-

#	Megnevezés	Forrás
25. ábra	A teszteléshez használt környezet	-
26. ábra	Minta a tesztek lehetséges kimeneteleire	-

8 Irodalomjegyzék

- [1] Sz. **Zilahi**: „*RFID: A jövő vonalkódja. Automatikus azonosítás a könyvtárban*”, könyv, könyvtár, könyvtáros, 2011/1
- [2] Y. **Liao**, J. **Lu**, M. **Wu**, „*Mobil Phone Book Finder*”, 2012
- [3] D. **Chen**, S. **Tsai**, B. **Girod**, C-H. **Hsu**, K-H. **Kim**, J. P. **Singh**, „*Bulding Book Inventories using Smartphones*”, ACM Conference of Multimedia (MM), 2010. október.
- [4] S. S. **Tsai**, D. **Chen**, H. **Chen**, C-H. **Hsu**, K-H. **Kim**, J. P. **Singh**, B. **Girod**, „*Combining Image and Text Features: A Hybrid Approach to Mobile Book Spine Recognition*”, ACM Conference of Multimedia (MM), 2011. november.
- [5] J. **Canny**, „*A computational approach to edge detection*”, IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.
- [6] I. **Sobel**, G. **Feldman**, „*A 3x3 isotropic gradient operator for image processing*”, a talk at the Stanford Artificial Project in, 1968, 271-272.
- [7] S. N. **Srihari**, V. **Govindaraju**, „*Analysis of textual images using the Hough*”, transform. Machine Vision and Applications, 1989, 2.3: 141-153.
- [8] H. **Bay**, A. **Ess**, T. **Tuytelaars**, L. V. **Gool**, „*Speeded-up robust features (SURF)*”, Computer Vision and Image Understanding, 110(3):346–359, 2008.
- [9] E. **Taira**, S. **Uchida**, H. **Sakoe**, „*A Model-based book boundary detection technique for bookshelf image analysis*”, 2004
- [10] H. **Chen**, S. S. **Tsai**, G. **Schroth**, D. M. **Chen**, V. **Chandrasekhar**, G. **Takacs**, R. **Vedantham**, R. **Grzeszczuk**, B. **Girod**, „*Robust text detection in natural images with edge-enhanced maximally stable extremal regions*”. In International Conference on Image Processing, 2011.
- [11] D. **Nister**, H. **Stewenius**, „*Scalable recognition with a vocabulary tree*”, In Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR’06), New York, NY, 2006, június.
- [12] G. **Jeon**, „*Edge Detection in Different Color Spaces*”, 2013.
- [13] E. **Nadernejad**, S. **Sharifzadeh**, H. **Hassanpour**, „*Edge detection techniques: evaluations and comparisons*”, Applied Mathematical Sciences, 2008, 2.31: 1507-1520.
- [14] D-J. **Lee**, „*An effective color addition to feature detection and description for book spine image matching*”. ISRN Machine Vision, 2011, 2012.

- [15] G. H. **Joblove**, D. **Greenberg**, „*Color spaces for computer graphics*”. In: ACM SIGGRAPH Computer Graphics. ACM, 1978. p. 20-25.
- [16] S. M. **Smith**, J. M. **Brady**, „*SUSAN—A new approach to low level image processing*”. International journal of computer vision, 1997, 23.1: 45-78.
- [17] C. **Harris**, M. **Stephens**, „*A combined corner and edge detector.*” In: Alvey vision conference. 1988. p. 50.
- [18] D. G. **Lowe**, „*Distinctive Image Features from Scale-Invariant Keypoints*”. International Journal of Computer Vision, 2004, 60.2: 91-110.
- [19] M. **Brown**, R. **Szeliski**, S. **Winder**, „*Multi-image matching using multi-scale oriented patches*”. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. IEEE, 2005. p. 510-517.
- [20] K. **Mikolajczyk**, C. **Schmid**, „*A performance evaluation of local descriptors*”. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2005, 27.10: 1615-1630.
- [21] T. G. **Smith**, W. B. **Marks**, G. D. **Lange**, Jr, W. H. **Sheriff**, E. A. **Neale**, „*Edge detection in images using Marr-Hildreth filtering techniques*”, Journal of neuroscience methods, 26(1), 75-81. (1988)
- [22] R. **Jain**, R. **Kasturi**, B. G. **Schunck**, „*Machine vision*”. New York: McGraw-Hill, 1995.
- [23] R. **Smith**, „*An overview of the Tesseract OCR engine*”. In: Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on. IEEE, 2007. p. 629-633.
- [24] J. **Schulenburg**, Gocr. Elérhető: <http://www-e.unimagdeburg.de/jschulen/ocr>, 2010.
- [25] N. **Otsu**, „*A threshold selection method from gray-level histograms*”. Automatica, 1975, 11.285-296: 23-27.
- [26] J. **Sauvola**, M. **Pietikäinen**, „*Adaptive document image binarization*”. Pattern Recognition, 2000, 33.2: 225-236.
- [27] W. **Niblack**, *An introduction to digital image processing*. Strandberg Publishing Company, 1985.
- [28] **Wang**, Z. **James**, ALIPR/SIMPLIcity/ALIP: Object Concept Recognition/Content Based Image Retrieval/Annotation. 1995.
- [29] **Jain**, K. **Anil**, and A. **Vailaya**, Image Retrieval Using Color And Shape. Great Britain: Elsevier Science Ltd, 1995.

- [30] **Jeong, Sangoh.** Histogram-Based Color Image Retrieval. Psych221/EE362 Project Report, Stanford University, 2001.
- [31] **J. Huang.** *Color-Spatial Image Indexing and Applications*. Ph.D thesis, Cornell Univ., 1998
- [32] **M. Mihályi, G. Windisch, Z. Vámosy:** *Szövegfelismerésen alapuló automatikus könyvkategorizáló alkalmazás*, Műszaki Tudományos Füzetek, Fiatal Műszakiak Tudományos Ülésszaka XIX., Erdélyi Múzeum-Egyesület kiadványa, Kolozsvár, 2014., p. 297-300. (ISBN 2067 – 6 808)
- [33] **R. G Von Gioi,** et al. LSD: *A fast line segment detector with a false detection control*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32.4: 722-732.
- [34] **L J Belaid; W. Mourou.** *Image segmentation: a watershed transformation algorithm*. Image Analysis & Stereology, 2011, 28.2: 93-102.
- [35] The MathWork Inc, <http://www.mathworks.com/help/images/examples/marker-controlled-watershed-segmentation.html> Utolsó látogatás:2014.11.11

9 Mellékletek

E fejezetben a témához kapcsolódóan publikált cikkek, teszt képek, amelyek mint a tesztelés során felhasznált képeket és a funkció végrehajtása után eredményül kapott képeket tartalmazzák, illetve táblázatos formában szerepel a tesztelés során elvégzett mérések részletes, egy-egy mérésre lebontott eredménye.

9.1 Publikációk

A TDK témájában kettő cikk született egy-egy magyar illetve angol nyelven. A magyar nyelvű 2014 márciusában megrendezésre került EME - Fiatal Műszakiak Tudományos Ülésszakán, Kolozsváron került bemutatásra. Az angol nyelvű 2014 novemberében a CINTI 2014 konferencián, Budapesten kerül bemutatásra. Ezen cikkek változás nélkül a következők:

9.1.1 EME - Fiatal Műszakiak Tudományos Ülésszaka



FIATAL MŰSZAKIAK TUDOMÁNYOS ÜLÉSSZAKA XIX.

Kolozsvár, 2014. március 20–21.

SZÖVEGFELISMERÉSEN ALAPULÓ AUTOMATIKUS KÖNYVKATEGORIZÁLÓ ALKALMAZÁS

AUTOMATIC BOOK CATEGORIZATION SYSTEM BASED ON TEXT RECOGNITION

Mihályi Martin ⁽¹⁾, Windisch Gergely ⁽²⁾, Vámosy Zoltán ⁽³⁾

(1) hallgató, Óbudai Egyetem, Neumann János Informatikai kar, H-1031 Budapest, Bécsi út 96/b, djozike@gmail.hu

(2) tanársegéd, Óbudai Egyetem, Neumann János Informatikai kar, H-1031 Budapest, Bécsi út 96/b, windisch.gergely@nik.uni-obuda.hu

(3) egyetemi docens, Óbudai Egyetem, Neumann János Informatikai kar, H-1031 Budapest, Bécsi út 96/b, vamosy.zoltan@nik.uni-obuda.hu

9.1.2 Abstract

In this study, we design and implement a book categorization system. Systems based on local image features are not effective in some cases in book spine detection. This is because few book spines contain only text component. The implemented system at first detects the book spines on the image of a bookshelf, then recognizes the text on the book spines by optical character recognition (OCR) and searches in the book database. Finally the implemented system displays the results for the user.

Key words: image processing, book spine detection, text recognition

9.1.3 Összefoglalás

Ezen cikk egy könyvkategorizáló rendszert mutat be. A képjellemző alapú rendszerek egyes esetekben nem mutatnak jó hatásfokot a könyvek esetében. Ennek oka, hogy néhány könyvgerinc csak szöveget tartalmaz. Ezért adott egy szövegfelismerés alapú rendszer alkalmazása a problémára. A fejlesztett alkalmazás először egy könyvespolcra készült képről könyvgerinceket detektál, majd a rajtuk lévő szöveget szövegfelismerés segítségével felismeri és a könyv adatbázisban keresést végez. Végül a fejlesztett rendszer a felhasználó számára megjeleníti az adott könyvről további információkat.

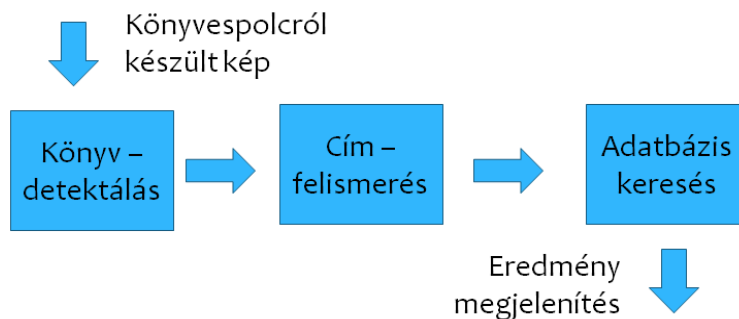
Kulcsszavak: képfeldolgozás, könyvgerinc keresés, szöveg felismerés

1. Bevezetés

Nagy mennyiségű könyv akár otthoni, akár bolti vagy könyvtári karbantartása, rendszerezése egy napjainkban is egy problémás feladat. A kézi könyvkategorizálás egy nagyon időigényes és rengeteg hibalehetőséget magában hordozó feladat. Léteznek erre már félautomata, rádió-frekvenciás (RFID) vagy vonalkód alapú megoldások, ezeknek különleges eszköz igényük van és használatához először el kell látni megfelelő azonosítóval a könyvek gerincét és ezért nem használhatók az otthoni vagy üzleti könyvespolc esetén. Megfelelő megoldás lehet egy képfeldolgozás alapú módszer. Ez esetben egy könyvespolcra mobiltelefonnal vagy akár számítógéppel készített kép alapján teljesen automatikusan végzi el a rendszer a könyvek karbantartását. A már meglévő kísérleti rendszerek ezt a képfeldolgozást képjellemző alapú módszerrel (referencia pont meghatározás, társítás) végzik, amely gyenge teljesítményt mutat olyan könyvek esetében, ahol kevés az ilyen meghatározható pont. Ilyen könyvek jellemzően azok, amelyek gerincén csak szöveges információ található. Ezen dolgozatban egy lehetséges megvalósítást mutatjuk be egy olyan rendszernek, amely szövegfelismerés alapján végzi a könyvek kategorizálását.

2. Rendszerfelépítés

A megvalósított alkalmazást három részmodulra került osztásra: egy könyvgerinc felismerő rész, egy szövegfelismerést végző rész, és egy adatbázis keresést végző rész, ahogy az 1. ábrán látható.



1. ábra. A megvalósított rendszer modul szintű felépítése

A könyvdetektáló rész feladata, hogy a könyvespolcra készült képen a könyvgerincek detektálása majd azok szerint az eredeti kép szétbontása olyan képekké, amelyeken már csak egy-egy könyv található. Ezt követően a cím felismerő modul előfeldolgozás után detektálja a képen lévő karaktereket, majd ezeken a karaktereken utófeldolgozást végez az adatbázis keresés modul és az így előálló karaktersorozattal leginkább egyező eredményt megjeleníti a felhasználó számára.

3. Könyvdetektálás modul

Ezen modul célja, hogy a könyvespolcra készült képen lévő könyveket meghatározza és könyvenként szétvágja a képet. A könyvdetektálás esetében először a képeken lévő élek kerültek meghatározásra. A legpontosabb él meghatározást a Canny éldetektáló [1] segítségével sikerült elérni. Majd ezekből az élekből Hough-transzformáció [2] segítségével vonalakat állítottunk elő. Ezt követően, mivel ismert, hogy a kép könyveket ábrázol, az egyes irányú élek előfordulásának arányából meghatározásra kerül a könyvek iránya, azaz merre található a legtöbb él, majd ha szükséges a kép elforgatásra kerül, hogy az élek függőleges irányúak legyenek. Utolsó lépésként az eredeti könyvespolcra készült kép detektált élek szerinti szétbontása történik. Fontos megjegyezni, hogy a bemeneti képen egyetlen sorban vagy oszlopban kell elhelyezkedni a

könyveknek, több sor esetén nem működőképes ez a megvalósítás. Az irodalom se jegyez olyan módszert, amely több sor vagy oszlop esetén is megfelelően működőképes lenne.

4. Címfelismerés modul

A modul célja, hogy a bemenetként érkező egy-egy könyvgerincet ábrázoló képeken meghatározzák a szövegeket. Ehhez először a képen előfeldolgozást kell végezni a jobb hatásfok elérése érdekében, ilyen szükséges előfeldolgozások a binarizáció és a szöveg orientációjának meghatározása. Ezt követően egy online karakter felismerő (OCR) rendszer elvégzi a karakterek felismerését. Majd ezt az eredményt szolgáltatja a következő modul számára.

4.1 Binarizáció

A binarizáció célja, hogy a karakterfelismerő számára kiemelje a képen a fontos részeket, azaz a betűket. Több binarizációs algoritmust is megvizsgálva (Otsu, Bradley, Niblack, Sauvola) az Otsu [3] binarizáció esetén sikerült a legnagyobb hatásfokot elérni a karakterfelismerés terén. Mint ahogy a 2. ábrán is látható Bertsekas: Introduction to Probability című könyvének gerincén végzett Otsu, Bradley [4] binarizációt, illetve ez alatt az adott binarizáció után kapott rendszer által felismert szöveg látható.



2. ábra Az egyes binarizációs algoritmusok eredménye, illetve az adott képen való szövegfelismerés eredménye

4.2 Szöveg orientáció

A szöveg orientációjának meghatározása is egy fontos kérdés, mivel a karakterfelismerő (OCR) csak a megfelelően elhelyezkedő betűket képes detektálni, azaz amelyek nem fejtetőn, vagy adott szöggel elforgatva helyezkednek el. Sajnos a könyv orientációjából nem következik a gerincen lévő betűk orientációja, mivel egyrészt a nemzetközi szabvány szerint a szöveget fentről lefele, a magyar szabvány szerint letről felfele kell a könyvgerincire írni, másrészt előfordulhat esztétikai okból ettől a szabványtól való eltérés is, például a szerző neve állítva, a cím elfordítva szerepel a könyvgerincen. Erre a problémára megoldást jelentet, hogy dilatációt végeztünk a binarizált képen, annak érdekében, hogy a detektált betűk összeérjenek. Ezek után egyszerűen meghatározhatóvá váltak az egy-egy összefüggő szövegrészek, amelyek szerint szegmentálható a kép, majd az egyes részeket különböző irányba elforgatva többször is karakterfelismerést végeztünk. Amelyik irányban a legnagyobb egyezést mutatja a az adatbázisból egy rekorddal az adott rész, az a megfelelő irány. Ez a megvalósítás azonban, a könyvek változatos színvilága miatt (sötét háttéren világos szöveg, világos háttéren sötét szöveg) nem javított jelentősen az ennél egyszerűbb és emiatt gyorsabb megoldáson, azaz amikor az egész binarizált képet négyszer végzünk karakterfelismerést (eredeti állapot, 90 fokkal való elforgatás, 180 fokkal való elforgatás, 270 fokkal való elforgatás). Ez esetben időigényesebb a karakterfelismerés (nagyobb területen alkalmazzuk), viszont a dilatáció és szegmentálás elhagyása miatt összességében gyorsabb megoldás.

4.3 Karakterfelismerés

Az alkalmazáshoz egy már létező online karakterfelismerő (OCR) engine-t használtunk. Az irodalom alapján a legmegfelelőbb ilyen rendszernek a Tesseract OCR engine [5] bizonyult, ezért

ezt használtuk. Az engine-t csak alfanumerikus karakterek felismerésére korlátoztuk (könyvek esetén kis eséllyel fordul elő más karakter).

4.4 Utófeldolgozás

A mérések során kiderült, hogy sok esetben a karakterfelismerő a valós képen lévő betűkön kívül a megjelenő zajokat is „felismerte”. Ezen fals pozitív betűk kiszűrésére egy szótár került kiépítésre, amely értelmes szavakat tartalmaz. A feldolgozás során a karakterfelismerés eredménye mindig ezzel a szótárral kerül összevetésre, és ha egy bizonyos százalékot (80-85%) meghalad az egyezés akkor a szótárból a megfelelő szóra cserélődik a detektált szó. Az összehasonlítást longest common subsequence (LCS) problémaként közelítettük meg és az szerint került megoldásra.

5. Adatbázis keresés modul

Az adatbázis egy szerveren került eltárolásra, a kliens oldal a könyvekről készült képet beküldi a szervernek, majd onnantól minden feldolgozási lépést a szerver végez, és az eredményeket visszaküldi a kliensnek, ami azt megjeleníti a felhasználó számára. Ezen adatbázisban a keresés a 4.4 pontban leírt módszerhez hasonlóan történik, az eltérés annyi, hogy nem az eredmény nem egy szótárral kerül összevetésre, hanem az adatbázisban tárolt szerzőkkel és címekkel.

6. Eredmények

A pontosabb összehasonlítás érdekében a tesztelesek az irodalomban [6] használt tesztadatbázis alapján történtek. Az eredmények csak a szövegfelismerést tekintve közel 70%-os sikeres felismerést mutattak, azaz az esetek 70%-ában csak a könyvgerincet ábrázoló kép alapján a megfelelő könyv adatokat jelenítette meg hozzá a rendszer. Kiemelendő az adatbázis kereső pontossága. Olyan esetekben, amikor szövegfelismerő motor legalább egy a címben szereplő szót teljesen hibátlanul felismert, az adatbázis keresés 95%-os pontossággal találta meg a hozzá tartozó könyvinformációt az adatbázisban.

7. Összefoglaló

Ezen cikkben bemutattam egy képfeldolgozás alapú könyvkategorizáló rendszer lehetséges megvalósítását, amely a hasonló rendszerekkel ellentétben csak szövegfelismerés alapján végzi a kategorizálást. Részletesen bemutattam a szövegfelismerés menetét, és megállapítottam, hogy ez a megközelítés jól használható, olyan könyvek esetében, amely gerincén kevés referencia pont, csak szöveg található, illetve hogy a címfelismerés hatásfoka nagyban függ a megfelelő binarizációs algoritmustól, illetve paraméter értékektől.

Irodalom

- [1] J. Canny: *A computational approach to edge detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 1986. pp. 8(6):679–698,
- [2] S. N. Srihari, V. Govindaraju: *Analysis of textual images using the Hough transform*, Machine Vision and Applications, 1989. 2.3, pp. 141-153.
- [3] N. Otsu: *A threshold selection method from gray-level histograms*. IEEE Transactions on Systems, Man and Cybernetics 1979, 9(1), pp. 62-66
- [4] D. Bradley, G. Roth: *Adaptive thresholding using the integral image*. Journal of graphics, gpu, and game tools, 2007, 12.2: 13-21.
- [5] R. Smith: *An overview of the Tesseract OCR engine*, ICDAR 2007. Ninth International Conference on Document Analysis and Recognition, IEEE, 2007. p. 629-633.
- [6] D. Chen, S. Tsai, C.-H. Hsu, K.-Y. Kim, J. P. Singh, and B. Girod: *Building book inventories using smartphones*, ACM Multimedia (MM), October 2010. pp. 651-654

9.1.4 CINTI 2014

Building Book Catalog Using Image Processing

Gergely Windisch*, Martin Mihályi** and Zoltán Vamossy**

* Óbuda University/ Doctoral School of Applied Informatics and Applied Mathematics, Budapest, Hungary

** Óbuda University/ John von Neumann Faculty of Informatics, Budapest, Hungary

windisch.gergely@nik.uni-obuda.hu

mihalyimartin91@gmail.com

vamossy.zoltan@nik.uni-obuda.hu

Abstract—In this article we describe an automatic initialization-free method for building and managing book catalogs. At the heart of the system there is an image processing algorithm which can analyze the contents of an image taken of a bookshelf, separate the book spines into subsequent images and use an OCR algorithm on it to find the title and author of the book. The title and author information is searched for in public book databases to find all the details of the given book which can then be stored. The article describes the methods used along with accuracy measurements for the different modules of our system.

I. INTRODUCTION

Organizing large book collections can be a tedious task. The initial digitalization of the indices take a lot of time, effort and typing as most catalogue applications do not provide much in the way of inserting multitudes of books into a digital library. We propose a novel method of organizing large collections easily and automatically using a mobile phone to take images of the books, and then using image processing algorithms to find the book spines, digitalize the text and then download the necessary information from the internet. All it takes for the user to digitalize the entire catalog is aim the camera at the bookshelf, press the capture button, and the rest is done automatically. Book catalogs nowadays usually use bar codes or RFID tags to identify the books which is an acceptable solution for managing databases but those methods require an initial step – marking each book with either a bar code or an RFID tag. There are more modern approaches similar to ours where an imaging device is used to get the contents of the spine of the books and compare said spine to the previously stored database [3, 10, 11].

These approaches all have one disadvantage in common (apart from being too costly for home use in the case of RFID and barcode): they require initialization step where the barcodes and RFID tags need to be attached or the spine has to be photographed and all the information of the book has to be typed into the computer manually.

The solution we propose consists of automatic book recognition based on an image of either the spine or the cover of the book complete with OCR capabilities to extract the most important information from the picture and a possibility to search for the extracted information on the internet to get all the rest of the data about the

book, thus making the whole process of catalogize an entirebook collection (Fig. 1.). The input image cancome from a multitude of sources ranging from mobile phones to unconventional camera setups [4, 19].

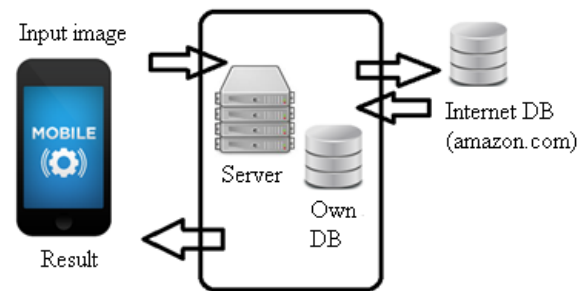


Figure 1. High level system architecture of the system

II. SYSTEM ARCHITECTURE

The workflow [16, 17] of our system is presented on Figure 2. The steps are as follows:

- Step 1. Image of Bookshelf: Acquire image of book / books. The image can be of a single book's cover or spine, or an entire bookshelf with multiple books with their spine visible.
- Step 2. Detection of Books: If there are multiple books on the image they have to be separated into multiple images to be processed.
- Step 3. Spine image in DB?: The image of the spine is compared to the database to see if it is already present. If it is, the book details are retrieved.
- Step 4. Preprocess Spine Image: Book spines and covers usually contain more texture than just the author/title of the book - background image, artificial texture, color. As for the OCR all that counts as clutter for better results it is removed at the preprocess stage, the image is binarized and the text is selected.
- Step 5. OCR: The textual information is fed into the Optical Character Recognition process which yields strings – text that can be searched for in the internet database.
- Step 6. Internet DB – OCR result: The OCR returns words, but they have to be decoded – which words belong to the author, and which are the title? Public

book databases are searched to try and identify the book.

- Step 7. Store spine img. in DB: The segmented spine image is stored in the database along with all the information received from the internet for future reference.
- Step 8. Present Result: The details of the book is presented to the user. The system itself consists of three main modules, in the next section those will be presented.

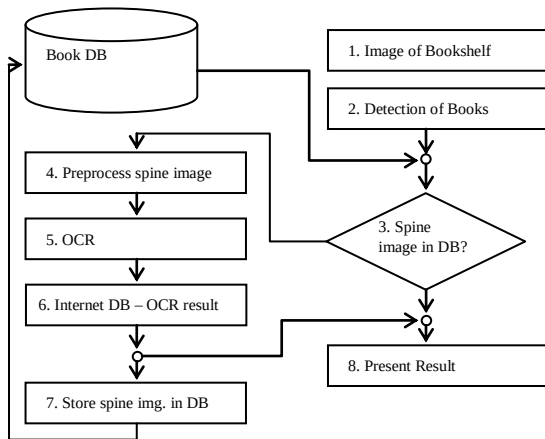


Figure 2. Workflow of the process

A. Book Detection Module

The aim of this book is to split the input image of a section of the bookshelf with multiple books on it into separate images where only one book spine is present. Our method is based on the detection of the parallel lines that separate the books from each other on an image. Edge detection is the first step, after thorough comparisons we have chosen the Canny edge detector as it was the most accurate on our testing set. The edge detection results in a binarized image where the white



Figure 3. Image of bookshelf after identification of book edges

points represent edge pixels in the original image. Usually after the edge detection there are more edges than the book edges that we are interested in so the next step is finding the edges that we are interested in (that is edges that separate books, not edges that are present because of the texture of any given book). The assumption is that the lines we are interested in are parallel. Hough transformation can be used to find such parallel lines, this is the algorithm we execute on the edge image. Once the parallel lines are identified the image can be separated along these lines and we get images of each book spine.

Using this method it is also possible to measure the slope of the parallel lines and rotating the image back to vertical/horizontal position that is better for the next steps. It is important to note here that on the original input image there can only be one row/column of books. Multiple rows above each other would render the method useless as it wouldn't be able to distinguish books directly above each other. This is common in all similar applications, there is no mention in the literature of a method that works with multiple rows.

A different approach to the image separation is the use of a finite state automate (FSA) as proposed in [fsa] that is also based on edge detection. Horizontal and vertical edges are separated into two different images. The images are then analyzed at 5-10 pixel wide bars at a time and the number of horizontal and vertical edges are counted. Depending on the number of edge pixels inside a region it is possible to specify what can be found at that region in the image. The method makes a distinction of the following cases: left/right boundary, left/right spine, title and background. There is an FSA representation of these states where only certain state changes are possible and that two features make this method accurate.

B. Title recognition module

In the previous step the original input image of the bookshelf has been segmented into smaller images, each image containing just one book spine, the next step is finding the text on the spine and converting it to real text using optical character recognition. The raw image has to be preprocessed for the OCR to work properly. It has to be binarized with all the clutter (texture, background image). The image is binarized and the text orientation is identified.

1) Binarization

The aim of the binarization step is to remove everything else but the title from the spine image so that the OCR can recognize it more easily without having problems with the rest of the image (see Figure 4).



Figure 4.: Binarization examples: Otsu / Bradley

There are many known binarization methods available, after testing multiple algorithms, like Otsu, Bradley, Niblack, Sauvola, we chose the Otsu binarization method for its robustness and accuracy. Figure 2 shows the results of binarization of the spine of the book "Bertsekas: Introduction to Probability". The result of Otsu binarization can be seen on the left of the image, the right side was created by the Bradley binarization. After the binarization further processing is necessary as there are still some particles left that are not part of the title.

2) Text orientation

Identifying the orientation of the text is also an important step as the OCR can only produce valid results for text that are horizontal and the bottom of the letters are pointing downwards. Rotating the book to a horizontal position is not enough as the books can be upside down – and even if it isn't, there are no universal standards that

describe how the text is written on the spine. It can be rotated + or – 90 degrees. It is also possible that the title of the book is written vertically and the author’s name appears horizontally for design reasons.

We came up with different solutions for the problem at hand. Using dilate on the image (which connects disconnected pixels thus making larger connected areas where it is easier to find the orientation) it is possible to make a connected segment of the adjacent letters thus making it easier to find the text and determine the orientation. The connected parts of the image can be further segmented into sub images and analyzed for content. OCR can be executed on these sub segments and the result can be compared to text already in database or in a dictionary and the orientation that yields the best result can be chosen as the orientation of that text.

The other possibility is to skip all the dilate and subsegmenting of the image and simply rotate the image 4 times, each time with 90 degrees, and do execute an OCR algorithm at each step. In this case the OCR takes longer because it has to analyze a larger image, but as there is no dilation and segmentation steps the overall performance is better in this case. Our measurements had shown that there is no significant benefit accuracy-wise of using the first, more complex method (mainly due to the differences in the color schemes of the books – light background with light / dark text, dark background with light/dark text and anything in between), so for the better performance (process time) we chose the latter option.

3) Optical Character Recognition

The heart of the application is the optical character recognition, this is the algorithm that has the most effect on the overall quality and accuracy of the system. We chose the Tesseract OCR engine [ocr] as it currently provides the best performance and its license permits using it in applications. We have limited the engine to alphanumeric characters as that is the most common in author names and book titles.

4) Post process

Our measurements have shown that on real images the OCR results are influence by the noise around the text. Noise is recognized as characters resulting in false positive hits, so we have devised a post process method that looks the hits up in a dictionary and if there is a similarity with a word, the OCR word is replaced to the one found in the dictionary. We use an 85% similarity as the threshold for the directory search. The string comparison is done using the Length of Longest Common Substring [llcs] method. Another possibility for the post process we have tried is to use Google to find similar words. When a Google search is performed Google compares the the search string to similar searches and is able to spot spelling mistakes. Our idea was to use this service with our results. Unfortunately there is no public API for this word comparison so we initiate a Google search and see what it results. Our measurements show that the method is mostly effective only when the OCR result resembles a common spelling mistake. That is so because Google’s spelling error database is populated by words misspelled by people using their search engine.

For more random mistakes it did not provide a benefit. At the current stage of the research we are allowing both methods, it can be configured which one should be used.

5) Database query module

The database is stored on a server, the client sends the input image to the server where all the processing takes place. The result is then sent back to the client. The same methods as described earlier is executed.

III. MEASUREMENTS

All the components in the system has undergone performance testing to see how accurate the book identification is.

A. Orientation detection

This test shows the accuracy of the orientation detection. A test case can have three different outcomes: hit, false negative, false positive. A hit is when the image had to be rotated and was rotated, false positive is when it shouldn’t have been rotated but has been, false negative is when it should have been rotated, but it hasn’t been.

TABLE I.
Oreintation Detection

No. of test cases	Hit	FP	FN
55	48	5	2

Table I sums the result. The results show that on the test set it worked well at around 85% of the cases.

B. Book Detection

This test shows the accuracy of finding the edges of books. The tests were conducted on our own images taken of bookselves. The Total Edges parameter is the number of edges the method should have been able to identify (for an example see Figure 1). Hit is the number of edges it identified correctly

1) Hough Transformation

Table II. contains the identification accuracy of the Hough transformation.

TABLE II.
Hough transformation result

Name of test set	Total Edges	Hit	FP	FN
Bookshelf 1	7	7	0	0
Bookshelf 2	25	22	9	3
Bookshelf 3	62	51	6	11
Bookshelf 4	23	21	12	2
Total	117	101 (86%)	27 (23%)	16 (13%)

2) Finite State Automate

On Table III. the accuracy of finding edges using the FSA method is shown.

TABLE III.
Finite state automaton result

Name of test set	Total Edges	Hit	FP	FN
Bookshelf 1	7	7	0	0
Bookshelf 2	25	24	2	1
Bookshelf 3	62	52	2	10
Bookshelf 4	23	19	13	4
Total	117	102 (89%)	17 (15%)	15 (11%)

According to these results there is no significant difference between the two methods, but further study is required to find the best at most situations.

C. OCR results

1) OCR without pre-processing

Table IV sums the results of executing the OCR algorithm on the raw book spine image without any preprocessing. That means the OCR has to cope with colors, different texture, anything that happen to be on the books or on the image as an artifact. Hit means that the correct character has been identified, false positive (FP) means that a character has been found where there hasn't been a character to be found, false negative (FN) means that a character has not been identified.

TABLE IV.

OCR Without preprocessing

Name of test set	Total # of chars.	Hit	FP	FN
Bookshelf 1	39	15	1	24
Bookshelf 2	18	10	5	8
Bookshelf 3	19	3	0	16
Bookshelf 4	20	1	1	19
Bookshelf 5	29	4	2	25
Total	125	33 (25.1%)	9 (6.8%)	92 (70.2%)

As can be seen from Table IV the accuracy is very poor for raw images. That is somewhat to be expected though, the OCR algorithms are designed to work on text images of homogeneous background and text with a distinctly different color. We have conducted measurements on the same raw images using commercial OCR applications to see if they yield the same accuracy or if there is some problem on our part with the implementation. The applications in question were ABBY Fine Reader 11 and OmniPage Recognita.

TABLE V.

Commercial OCR

	Perfect	Hit	Miss	Orientation
OmniPage	3 (20%)	2 (13.3%)	9 (60%)	1 (6.6%)
ABBY	9 (60%)	3 (20%)	1 (6.6%)	1 (6.6%)

The commercial applications received the segmented images of book spines as they do not have segmentation capabilities. OmniPage was on the same accuracy level as our own application, ABBY Fine Reader performed significantly better which shows that there is still room to improve.

2) OCR with preprocessing

As expected, the OCR module does not perform well on the raw images as such algorithms are not designed for those images. Preprocessing is essential. In our solution we propose binarization to get an image that is more suitable for the OCR algorithm. Different binarization algorithms were implemented, Table VII sums the results.

TABLE VI.

Binarization results

	Hit	Miss
Otsu	23	12
Bradley	7	27
Niblack	8	26
Sauvola	9	25

Table VII shows whether after the binarization the OCR has been able to identify the text (we have set an 80% threshold for similarity) or not. Hit means that the OCR was successful on the binarized image, miss means that it hasn't been successful – that is the binarization algorithm has left too much clutter or has destroyed part of the text as well. As for the binarization results: Otsu binarization has proven to be the most effective, however the Sauvola method is also interesting because on 8 of the 9 hits this has been the only method that had actually done something useful. We propose applying both the Otsu and the Sauvola binarization on complex images and have both results be analyzed afterwards.

3) OCR with manual preprocessing

In order to find the maximum accuracy of the OCR we have carried out a measurement where the preprocessing of the image has been done manually using an image manipulation software. Everything but the text has been removed and it has been binarized. The idea behind the measurement has been to see if given the right conditions the system could actually achieve over 95% accuracy. Table VIII shows the results.

TABLE VII.

OCR With manual preprocessing

	Hit	Miss
OCR	34	1

As is apparent from Table VIII, using manual preprocessing our OCR algorithm has been able to achieve 97.1% accuracy so it is worthwhile to keep working on a more effective preprocess algorithm.

D. Overall System Performance

Table VIII sums the overall performance of the system. The full workflow has been measured, the input was an image taken of the bookshelf, the end result was the text written on the spine. Successful recognition means that the text on the spine was recognized successfully. Unsuccessful means that the text was not found.

TABLE VIII.

Overall system performance

# of test cases	Successful	Unsuccessful
30	18 (60%)	12 (40%)

IV. CONCLUSION

In this paper we have introduced the system that has been devised for automatic, initialization free book cataloging. The input of for the system is an image taken of a bookshelf which is segmented into smaller images containing only one book spine per image, it is preprocessed and an OCR algorithm finds all the text, which is fed into a database search to find the title and author, and then using that data all the details of the book are downloaded from the internet. Different algorithms have been implemented and tested for the steps, the measurement results can be found in Section III. The current accuracy of the whole system is approximately

60% so there is still room for improvement with different optimization methods [12-15]. Using different color spaces could also improve the performance [5].

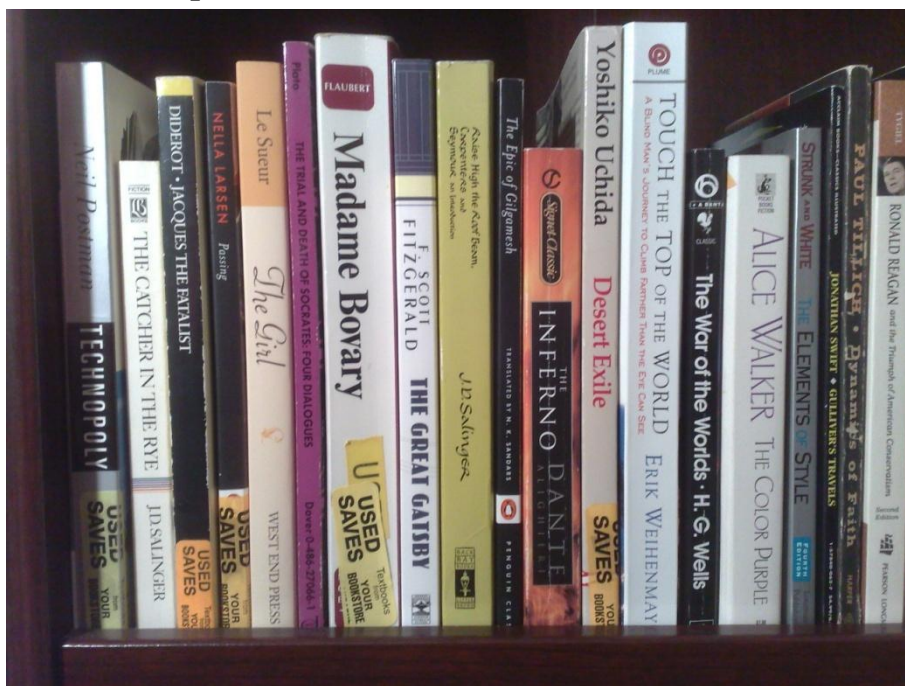
REFERENCES

- [1] R. Smith. "An Overview of the Tesseract OCR Engine", *ICDAR*. 2007. Parana, Brazil, pp. 629–633
- [2] M. Paterson and V. Dančik. "Longest common subsequences", *Proc. 19th MFCS, Lecture Notes in Computer Science*, Vol. 841, Springer, Berlin, 1994, pp. 127–142.
- [3] E. Taira, S. Uchida, H. Sakoe, "A Model-based book boundary detection technique for bookshelf image analysis", *Asian Conference on Computer Vision (ACCV2004, Jeju Island, Korea)* vol.2, pp. 634-639.
- [4] T. Hashimoto, T. Suzuki, H. Aoshima, A. Rövid "Multi-Camera-based High Precision Measurement Approach for Surface Acquisition", *Acta Polytechnica Hungarica*, 10(8) 2013, pp. 139-152.
- [5] M. Apró, D. Novaković, S. Pál, S. Dedijer, N. Milić "Colour Space Selection for Entropy-based Image Segmentation of Folded Substrate Images" *Acta Polytechnica Hungarica*, 10(1), 2013, pp. 43-62.
- [6] J. Canny, "A computational approach to edge detection", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6)1986. pp. 679–698
- [7] S. N. Srihari, V. Govindaraju, "Analysis of textual images using the Hough transform", *Machine Vision and Applications*, 1989. 2.3, pp. 141-153.
- [8] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man and Cybernetics* 1979, 9(1), pp. 62-66
- [9] D. Bradley, G. Roth, "Adaptive thresholding using the integral image", *Journal of graphics, GPU, and game tools*, 2007, 12.2, pp. 13-21.
- [10] D. Lee, Y. Chang, J. Archibald, and C. Pitzak. "Matching book-spine images for library shelf-reading process automation". In *Proc. IEEE International Conference on Automation Science and Engineering (CASE'08)*, Arlington, 2008. pp. 738–743.
- [11] D. Chen, S. Tsai, B. Girod, C-H. Hsu, K-H. Kim, J. P. Singh, "Bulding Book Inventories using Smartphones!", *ACM Conference of Multimedia (MM)*, 2010. Italy pp. 651-654.
- [12] M. Ge, Q.-G. Wang, M.-S. Chiu, T.-H. Lee, C.-C. Hang, and K.-H. Teo, "An effective technique for batch process optimization with application to crystallization", *Chemical Engineering Research and Design*, vol. 78, no. 1, 2000, pp. 99-106.
- [13] S. Preitl, R.-E. Precup, J. Fodor, and B. Bede, "Iterative feedback tuning in fuzzy control systems. Theory and applications", *Acta Polytechnica Hungarica*, vol. 3, no. 3, 2006, pp. 81–96.
- [14] M. Z. Ali, K. Alkhatib, and Y. Tashtoush, "Cultural algorithms: Emerging social structures for the solution of complex optimization problems", *International Journal of Artificial Intelligence*, 2013, vol. 11, no. A13, pp. 20-42.
- [15] M. Mavrovouniotis and S. Yang, "Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors", *Applied Soft Computing*, vol. 13, no. 10, 2013, pp. 4023-4037.
- [16] J. Tick, Z. Kovacs, F. Friedler, "Synthesis of Optimal Workflow Structure" *Journal of Universal Computer Science* vol. 12: (9) 2006, pp. 1385-1392
- [17] J. Tick, "Potential Application of P-Graph-Based Workflow in Logistics" In: *Aspects of Computational Intelligence: Theory and Applications: Revised and Selected Papers of the 15th IEEE International Conference on Intelligent Engineering Systems INES 2011*. Springer Verlag, 2013. pp. 293-303
- [18] S. Szénási, "Medical Image Segmentation with Split-and-Merge Method", *LINDI 2013: 5th IEEE International Symposium on Logistics and Industrial Informatics*, Wildau, 2013. pp. 137-140.
- [19] Gy Györök, T Orosz, M Makó, T Treiber, „To Achieve Circuit Robustness by Co-operation of FPAA and Embedded Microcontroller" *SACI 2013 – 8th International Symposium on Applied Computational Intelligence and Informatics*, Timisoara, România, 2013. pp. 315-320.
- [20] A. Rövid, L. Szeidl, Sz. Sergyán, P. Várlaki, "HOSVD-Wavelet Based Framework for Multidimensional Data Approximation", *Proceedings of the ICCV 2003 IEEE International Conference on Computational Cybernetics*, Tihany, Hungary, 2013. pp. 29-33.

9.2 Könyvgerinc detektálás

Ebben a fejezetben a könyvgerinc detektáló modul bemeneti képeit, illetve ezen bementi képekre kapott eredményeket szemléltetem. Az eredmények ugyancsak képeken kerülnek bemutatásra a jobb szemléltetés végett.

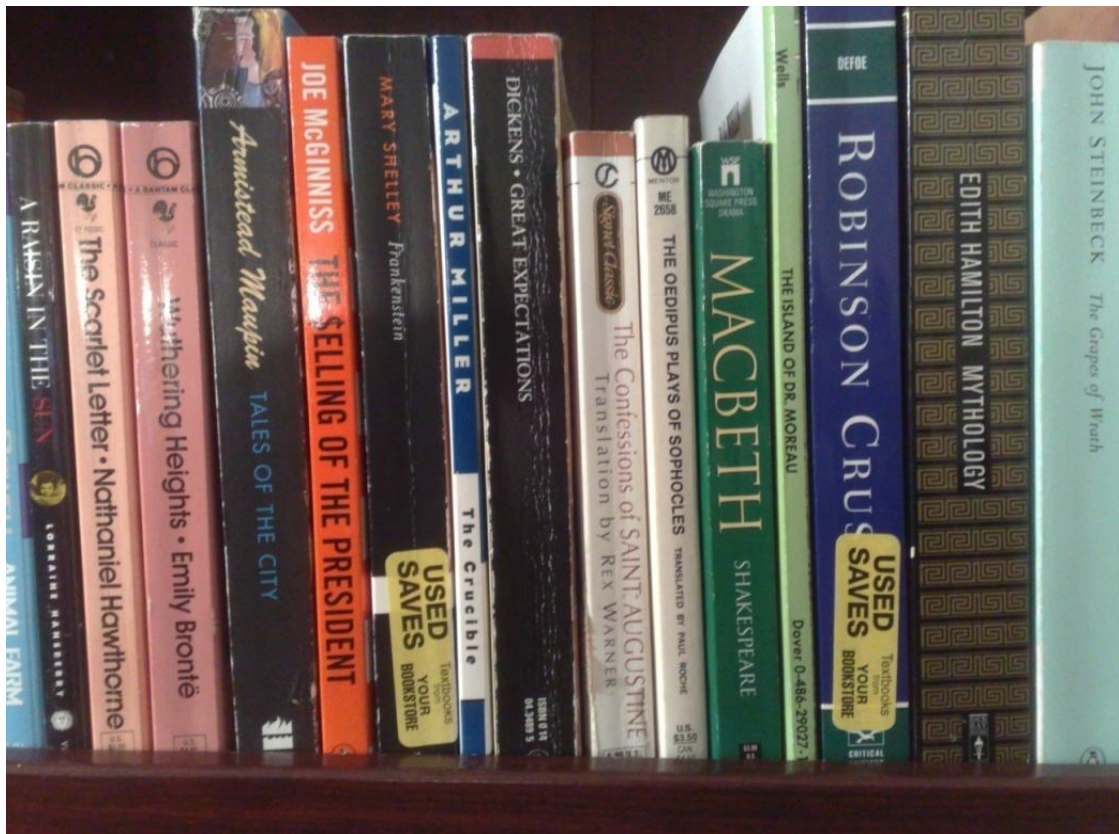
9.2.1 Bemeneti képek



1. tesztkép



2. tesztkép

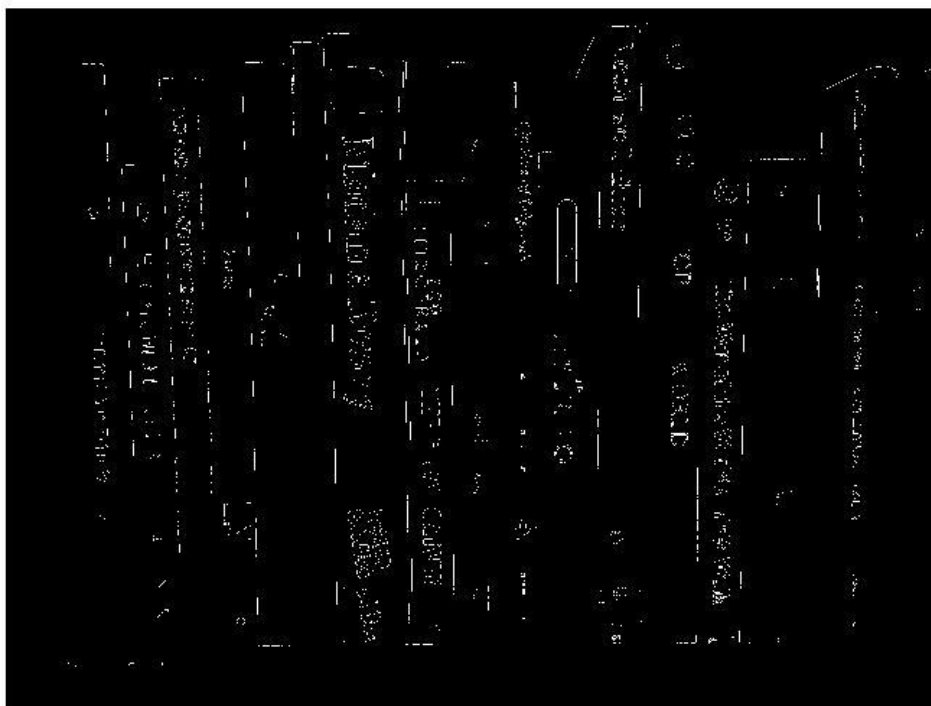


3. tesztkép

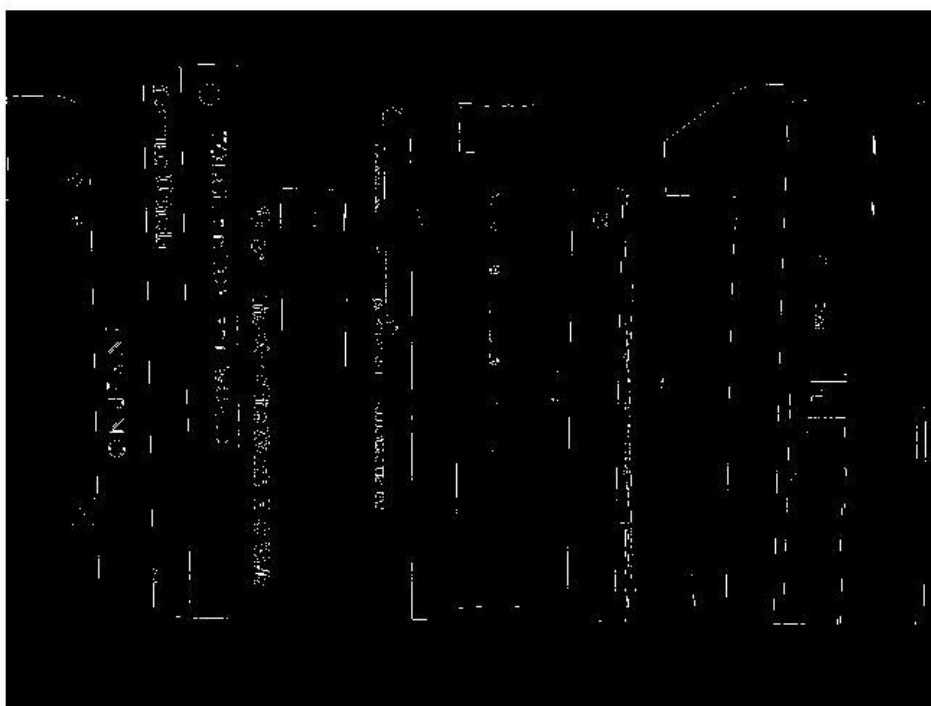


4. tesztkép

9.2.2 Canny éldetektáló



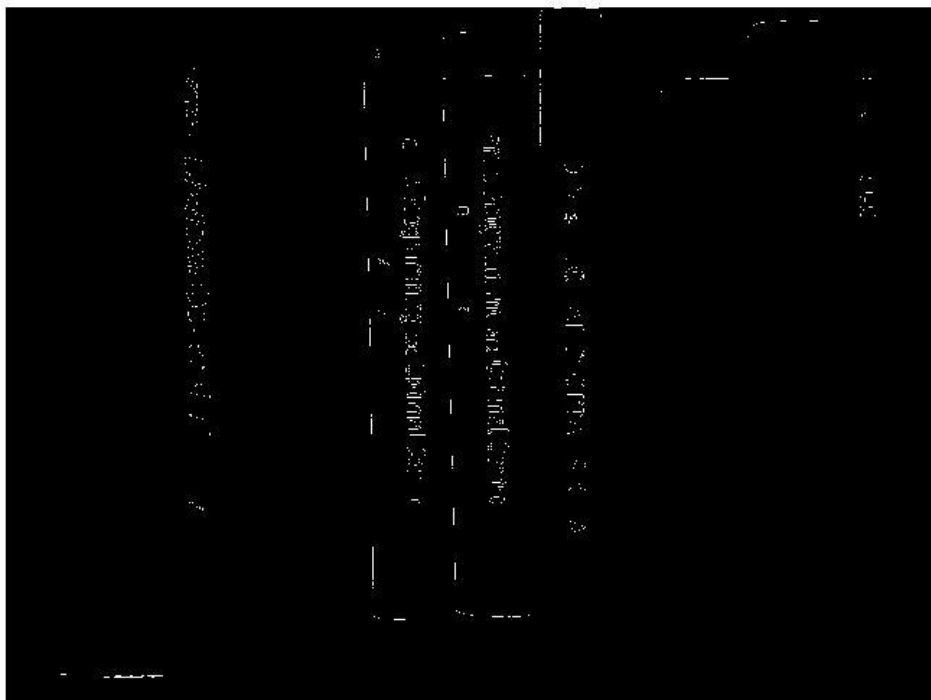
5. tesztkép



6. tesztkép



7. teszkép

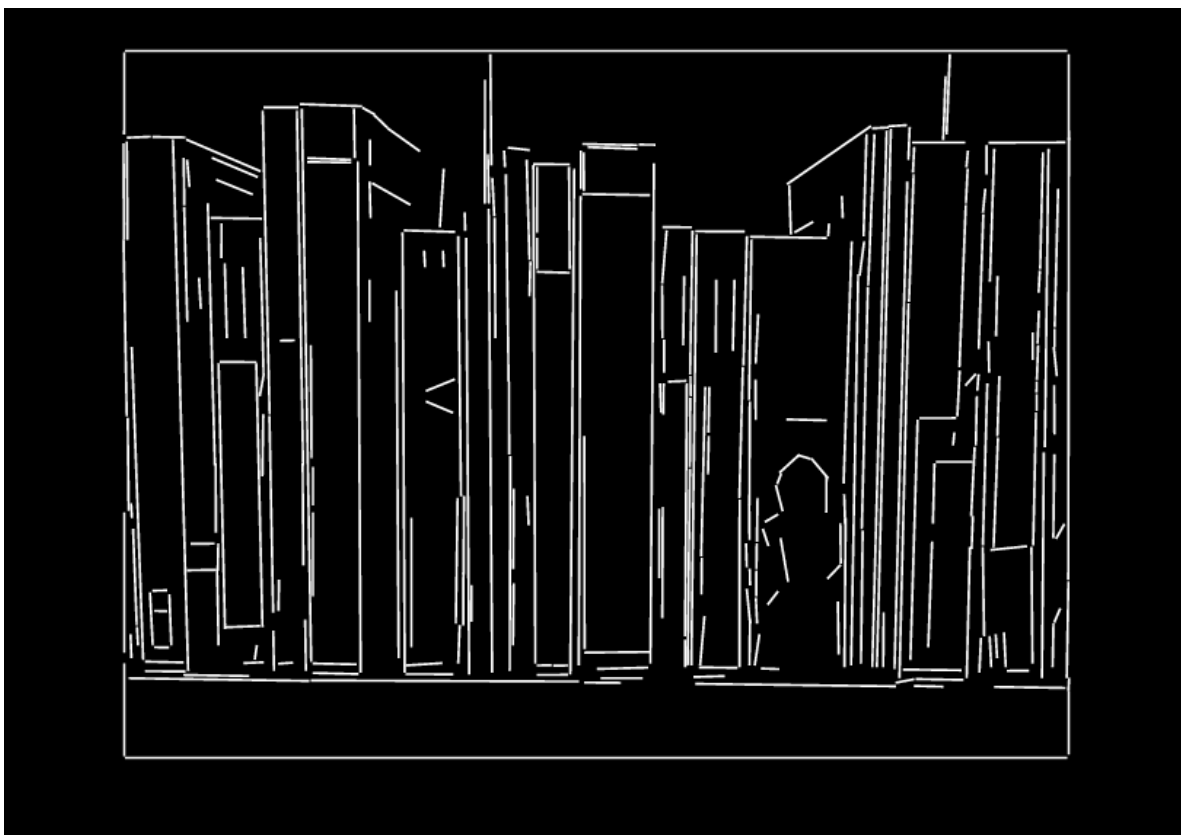


8. teszkép

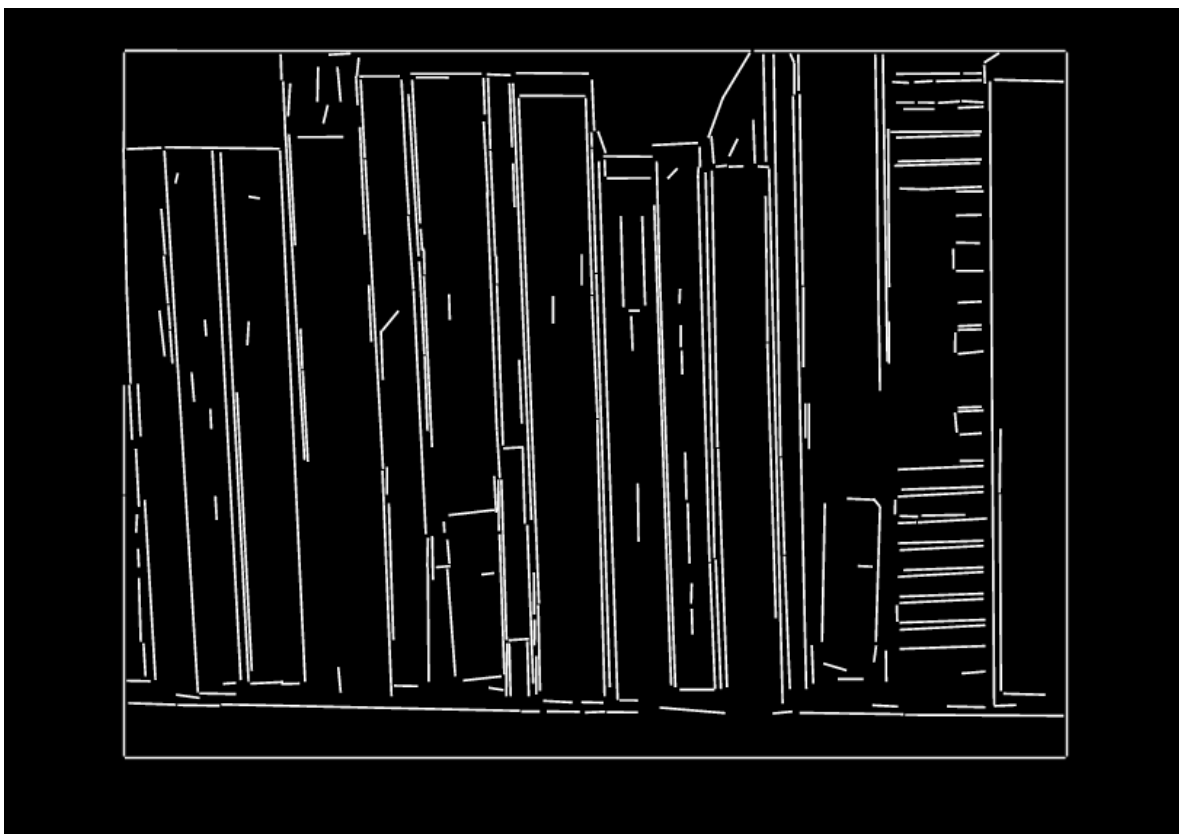
9.2.3 LSD- Line Segment Detection



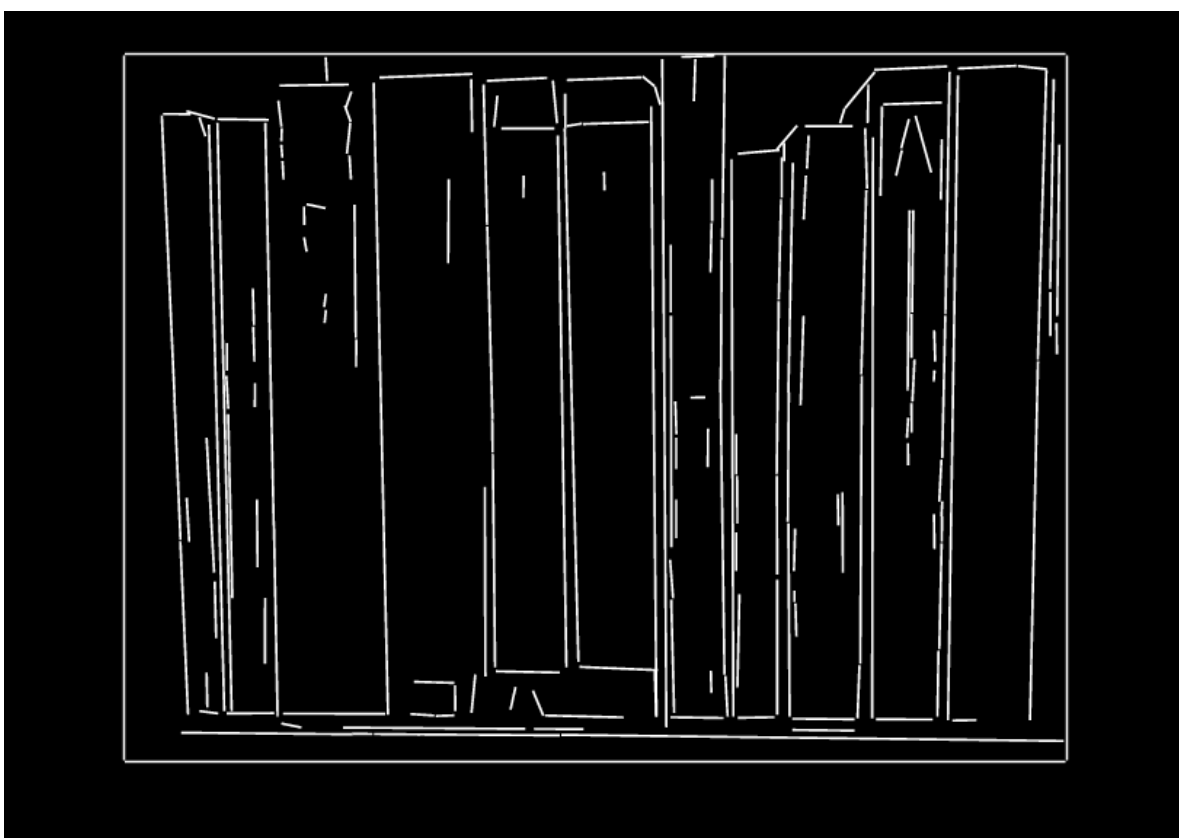
9. tesztkép



10. tesztkép



11. tesztkép



12. tesztkép

9.2.4 Canny éldetektáló és Hough Transzformáció

A képeken látható függőleges sárga vonalak jelölik a detektált könyv széleket.



13. tesztkép



14. tesztkép



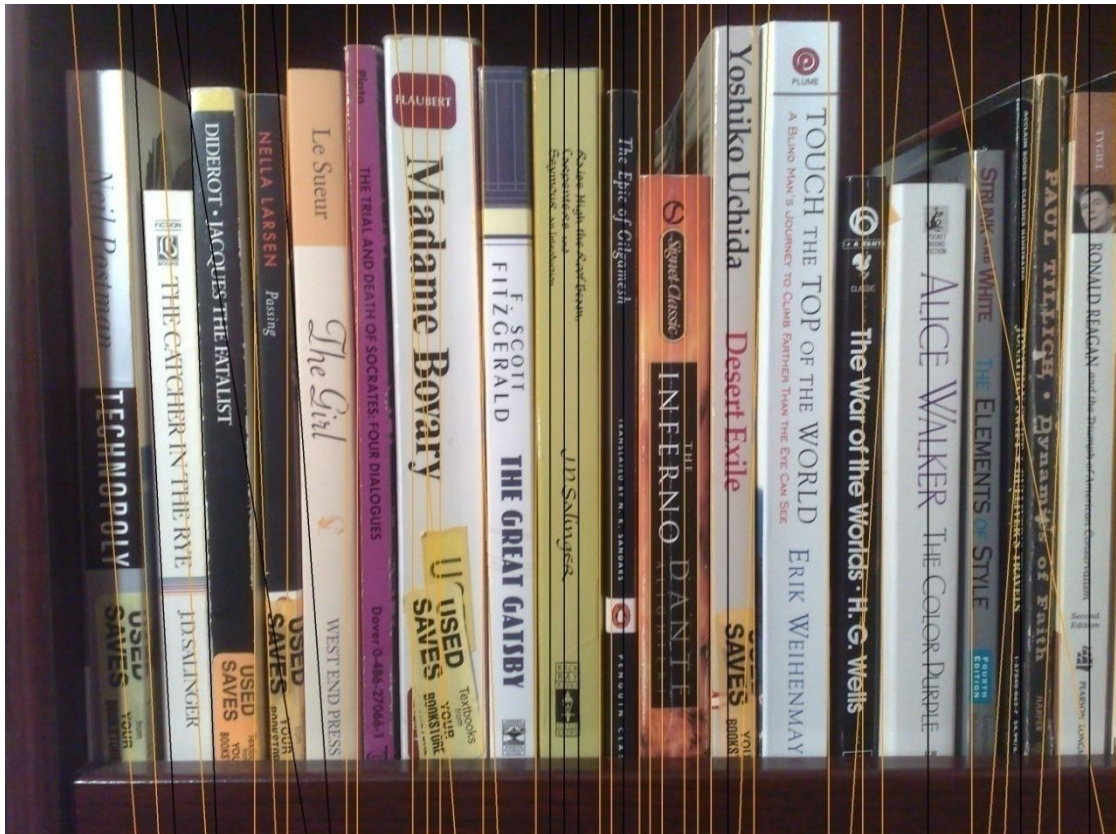
15. tesztkép



16. tesztkép

9.2.5 Canny éldetektáló és FSM

A detektált könyv széleket a sárga és fekete vonalak jelzik. A sárga vonalak a rendszer által valós felismerésnek értékelt élek, a feketék a fals pozitívnak vélt éleket jelentik.



17. tesztkép



18. tesztkép

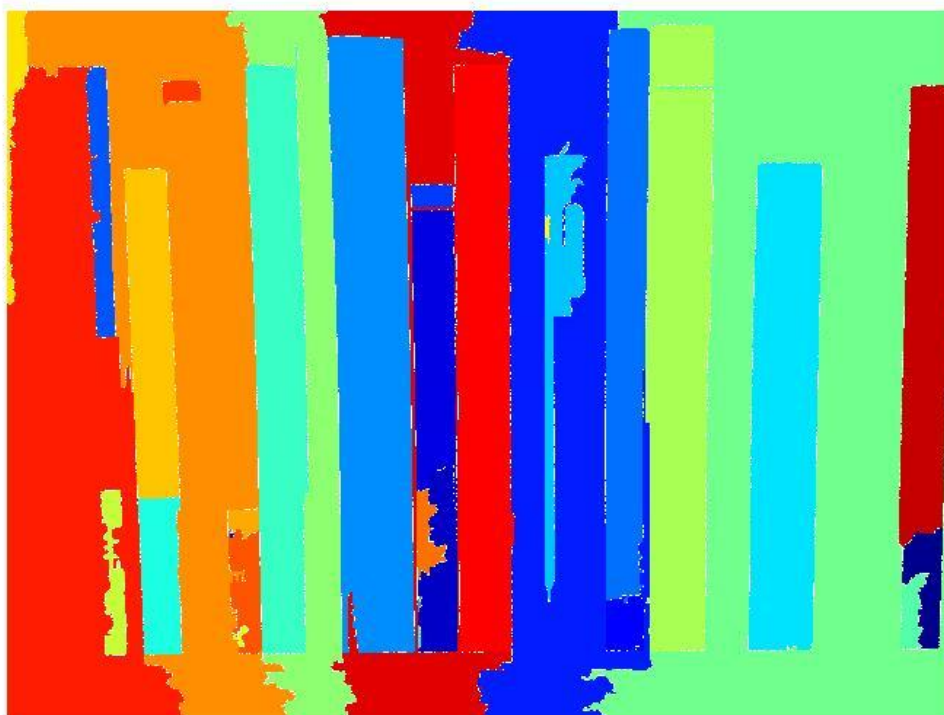


19. tesztkép

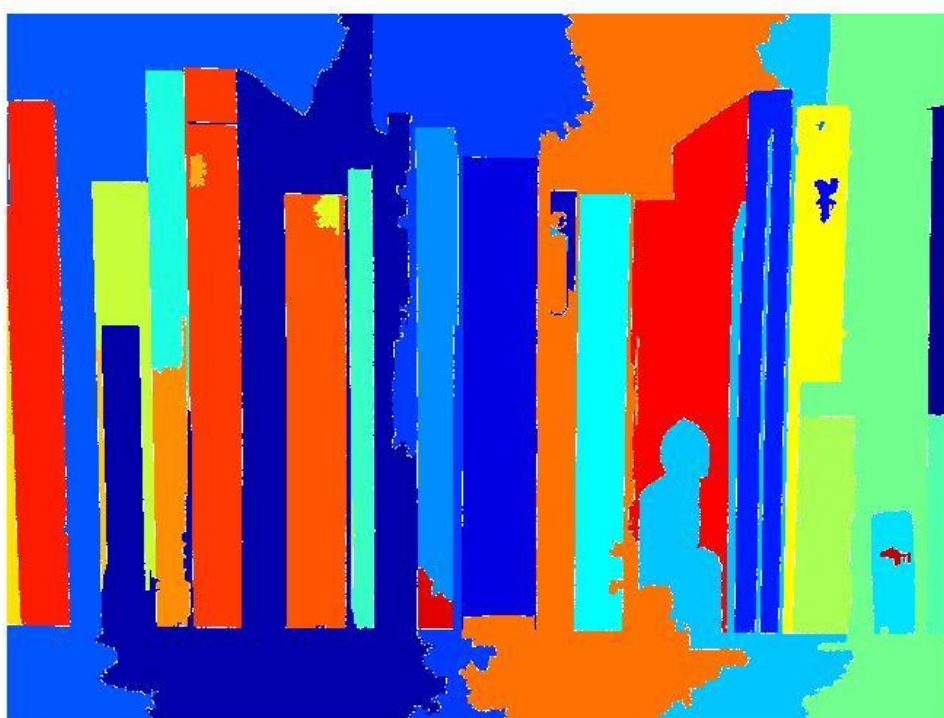


20. tesztkép

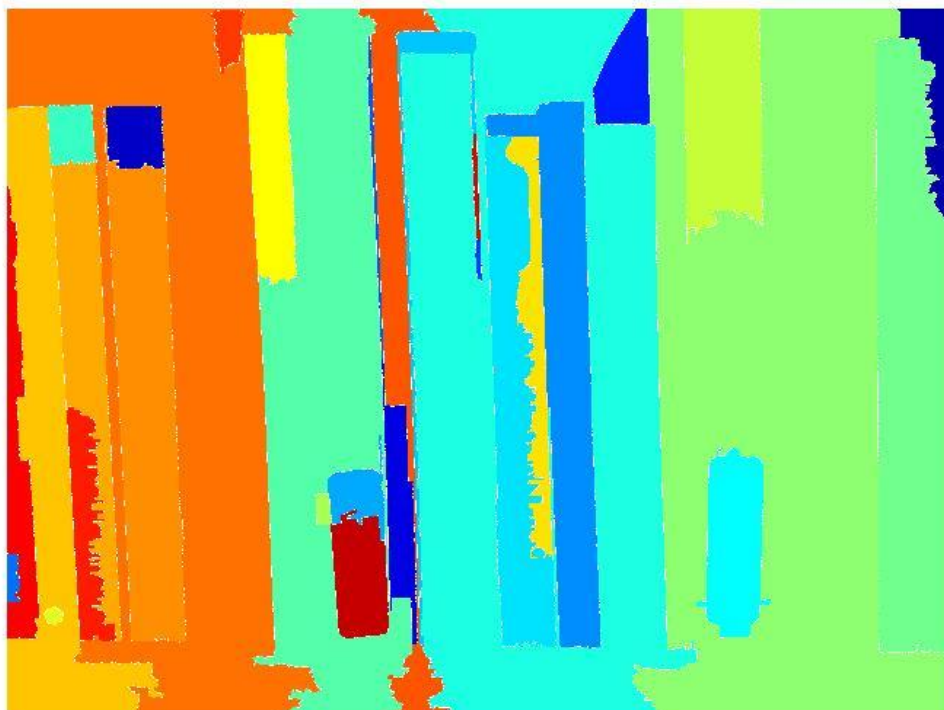
9.2.6 Watershed algoritmus



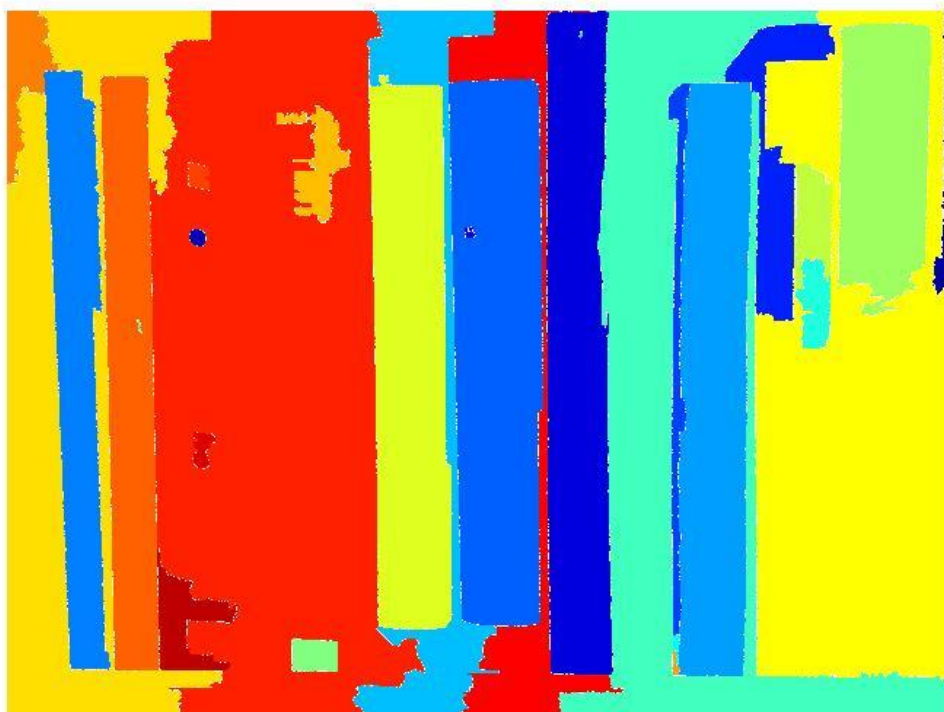
21. tesztkép



22. tesztkép



23. tesztkép



24. tesztkép

9.2.7 Canny éldetektáló, nyitás- zárás

Opening-closing by reconstruction (lobrcbr)



25. tesztkép

Opening-closing by reconstruction (lobrcbr)



26. tesztkép

Opening-closing by reconstruction (l0brcbr)



27. tesztkép

Opening-closing by reconstruction (l0brcbr)

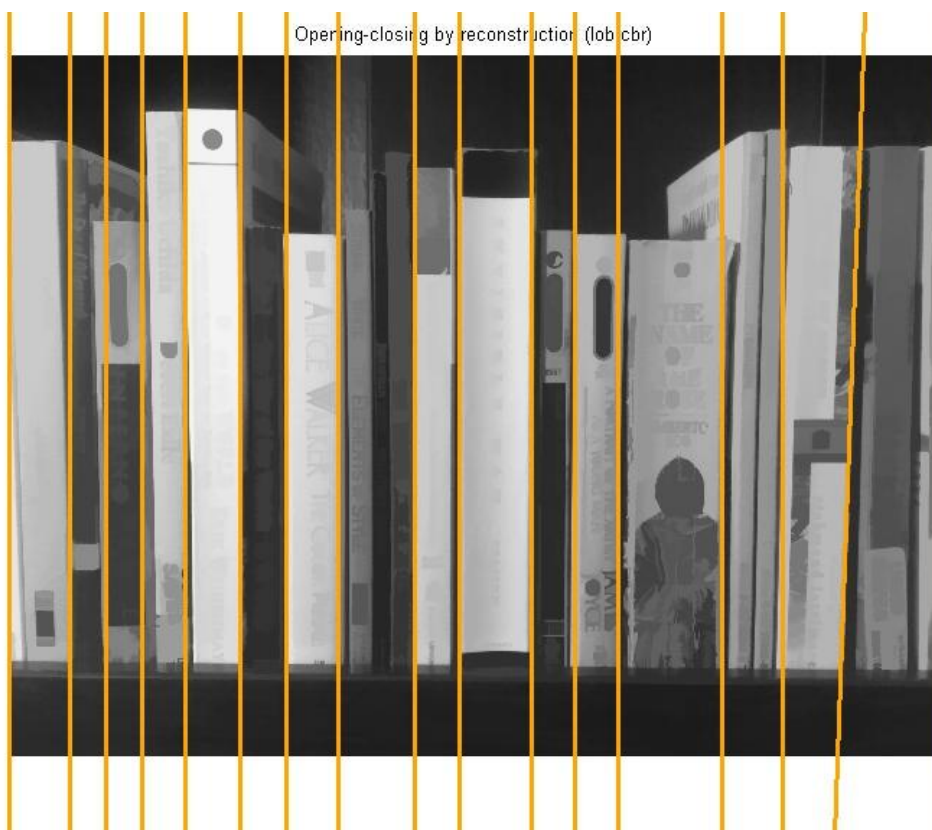


28. tesztkép

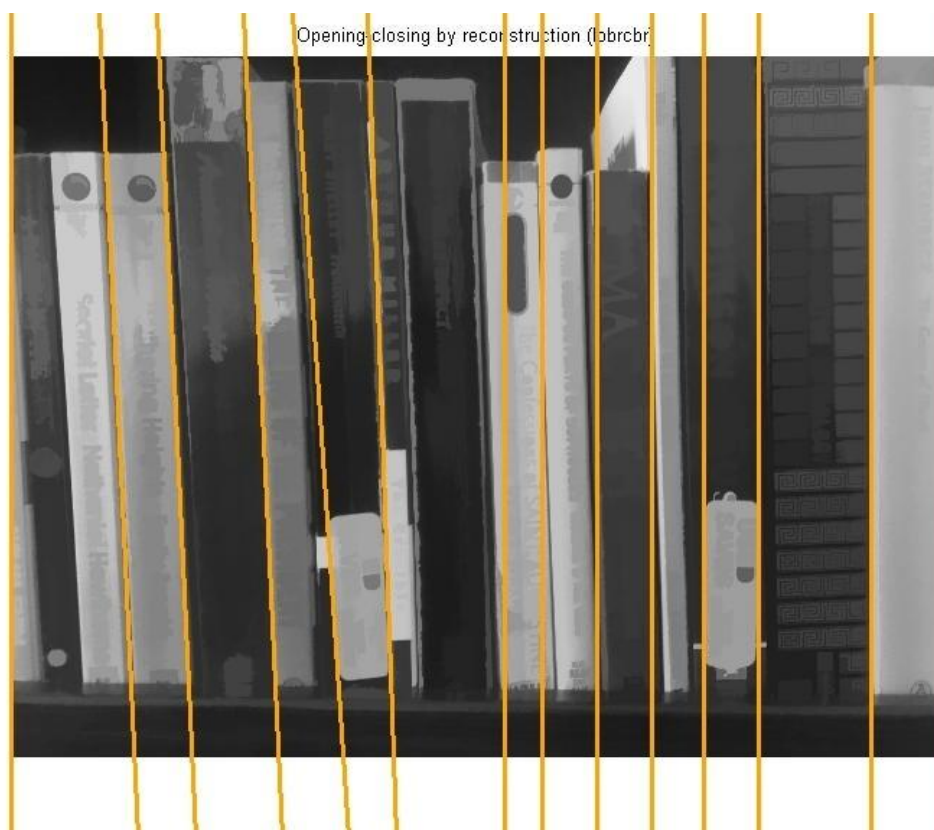
9.2.8 Canny éldetektáló, nyitás - zárás, Hough Transzformáció



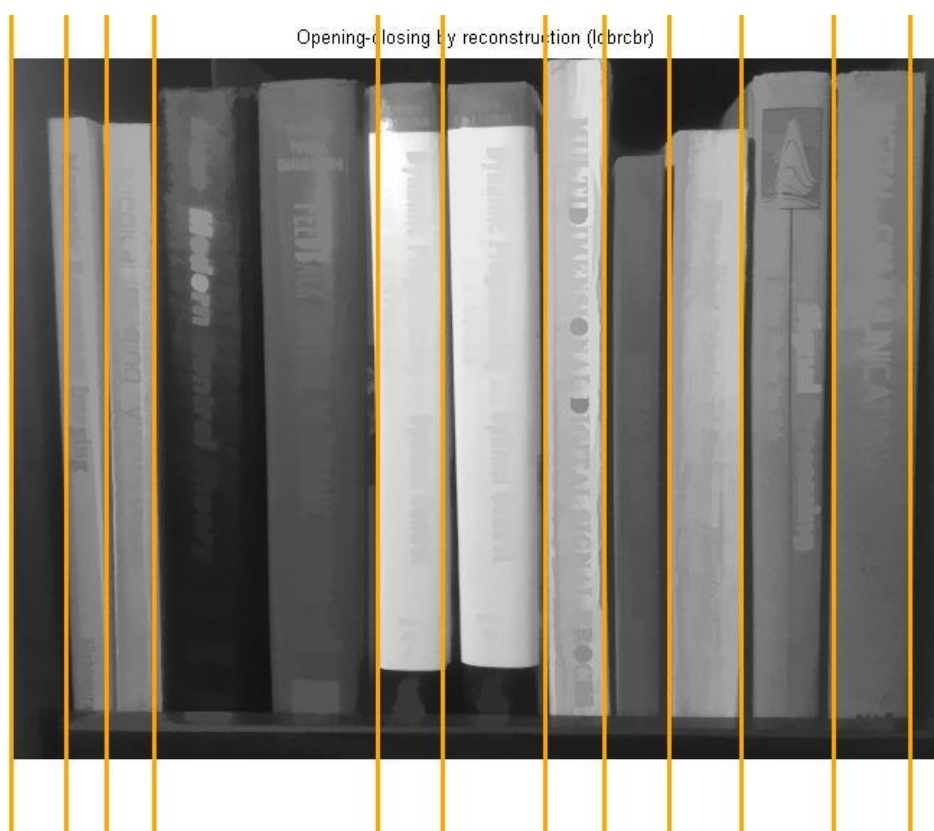
29. tesztkép



30. tesztkép



31. tesztkép



32. tesztkép

9.3 Részletes teszteredmények – Szegmentálás

Ebben a fejezetben számszerűsítve ismertetem az egyes szegmentálási eljárások eredményességét a mintaképeken. Külön kitérek a futási idők vizsgálatára is.

Pozitív találatok: Azon felismert élek, amelyek valóban könyvgerinc szélek és ezt megfelelően detektálta is az algoritmus.

Fals Pozitív találatok: Azon felismert élek, amelyek az algoritmus szerint könyvgerincek szélei azonban a valóságban nem azok.

Negatív találatok: Azon találatok, amelyek a képen a könyvgerinc szélén valóban elhelyezkedő élek, azonban ezt az algoritmus nem volt képes megfelelően felismerni.

Futási idő: A művelet elvégzéséhez szükséges idő. Minden egyes teszt azonos rendszeren, azonos beállításokkal lett elvégezve.

Művelet	Pozitív találatok	Fals Pozitív	Negatív találatok	Futási idő
1. tesztkép				
Canny érdetektálás, Hugh Transzformáció				
Eredeti képen	16	22	2	4382 ms
Kicsinyített képen	13	7	5	1024 ms
Nyitás-zárás morfológiai művelet után	14	0	4	459 ms
LSD – Line Segment Detection				
Nyitás-zárás morfológiai művelet után	18	1	0	230 ms
2. tesztkép				
Canny érdetektálás, Hugh Transzformáció				
Eredeti képen	16	25	2	4123 ms
Kicsinyített képen	16	4	2	902 ms
Nyitás-zárás morfológiai művelet után	15	0	3	352 ms
LSD – Line Segment Detection				
Nyitás-zárás morfológiai művelet után	18	0	0	240 ms
10. tesztkép				
Canny érdetektálás, Hugh Transzformáció				
Eredeti képen	14	26	0	3981 ms
Kicsinyített képen	10	10	4	936 ms
Nyitás-zárás morfológiai művelet után	11	1	3	331 ms
LSD – Line Segment Detection				
Nyitás-zárás morfológiai művelet után	14	0	0	250 ms
12. tesztkép				
Canny érdetektálás, Hugh Transzformáció				
Eredeti képen	6	11	6	4420 ms
Kicsinyített képen	9	7	3	882 ms
Nyitás-zárás morfológiai művelet után	10	1	2	323 ms
LSD – Line Segment Detection				
Nyitás-zárás morfológiai művelet után	12	2	0	240 ms

9.4 Binarizáció

Ebben a fejezetben néhány teszt képen szemléltetem a különböző binarizációs algoritmusok eredményességét a már előre szegmentált, egy-egy könyvgerincet tartalmazó tesztképeket.

9.4.1 Mintaképek

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

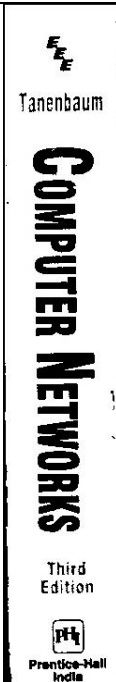
9.4.2 Optimális eredmény

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

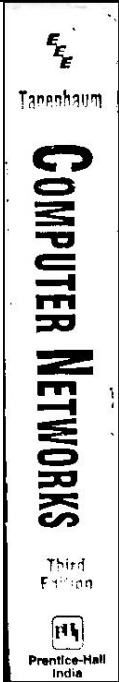
9.4.3 Otsu binarizáció

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

9.4.4 Niblack binarizáció

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

9.4.5 Bradley binarizáció

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

9.4.6 Sauvola binarizáció

				
Introduction to probability	Learning OpenCV	Neural Networks	Computer Networks	Data structures and algorithms

9.5 Részletes teszteredmények – Binarizáció

Ebben a fejezetben számszerűsítve ismertetem az egyes szegmentálási eljárások eredményességét a mintaképeken. Külön kitérek a futási idők vizsgálatára is.

Találatok: A megfelelően elfordított képen felismert pontos szöveg.

Szótár találatok: A felismert pontos szöveg alapján a szótárban társított cím.

Módszer	Találatok	Szótár találat
Introduction to probability		
Otsu	1lllllll INFRODUClION ro PROBABILITY ti iitiili My	Bertsekas Introduction to Probability
Bradley	l tEEiiEiiiitJTiiJEtiii2t2 MI flt. INN? tlt tgggmmMllllllMlillg1 itG7	Foley Computer Graphics
Niblack	1lllllllliaiuuiiiiii0iiiiiaatiitli M'	Landau Mechanics
Sauvola	b wip4. i'lu1fi1At withiiiiiiiiiigijt s ii1iii1 Dimitn P. Bemehl john burmam	EinsteinSpecialandGeneralRelativity
Learning OpenCV		
Otsu	lr Learning OpenCV ttedtmt	BradskiLearningOpenCV
Bradley	I 1lllllMllll tlllllllllllllllllllri	FlierlVideoCodingwithSuperimposedMo tionCompensatedSignals
Niblack	itali%hiriii2 IP V ki?! BMEtrs m mkw 1 A§ Tec	GibsonDigitalCompressionforMultimedi a
Sauvola	I . V Mjdmx w Learning OpenCV liarhlc	BradskiLearningOpenCV
Neural Networks		
Otsu	s.ll NEURAL NETWORKS C	Haykin Neural Networks
Bradley	lli NEURAL NETWORKS	Haykin Neural Networks
Niblack	.t NEURAL NETWORKS tft	Haykin Neural Networks
Sauvola	IMF! o . rr	Feyn Manged
Computer Networks		
Otsu	w COMPUTER Nnwnxs i l	TanenbaumComputerNetworks
Bradley	lllltl Comma Nnwnxs I Itl	AhoDataStructuresandAlgorithms
Niblack	w Comrursn !IMtlll! iti til	TanenbaumComputerNetworks
Sauvola		Sikertelen
Data structures and algorithms		
Otsu	f mrrmirilr..eie.z.ee..t. iiiarNllltllllil&	Feyn Manged
Bradley		Sikertelen
Niblack	rrrrrmirre.e.ie..zeee stiiirNlllllMV	Feyn Manged
Sauvola	nun rrnucru Ann unonmms illEmuuuuu A	Feyn Manged