

TDK-dolgozat

Katona Melinda

Szegedi Tudományegyetem
Természettudományi és Informatikai Kar
Képfeldolgozás és Számítógépes Grafika Tanszék

Vizuális kódok hatékony detektálása

Készítette:

Katona Melinda

II. éves programtervező informatikus
MSc szakos hallgató

Témavezető:

Dr. Nyúl László

egyetemi docens

Tartalomjegyzék

Bevezetés	4
1. A vizuális kódok áttekintése	5
1.1. Történeti áttekintés	5
1.2. Főbb jellemzők	6
1.3. Felhasználási terület, csoportosítás	7
1.4. Azonosítás	7
1.5. Kapcsolódó munkák	8
2. Az alkalmazott képfeldolgozó műveletek áttekintése	10
2.1. Előfeldolgozó műveletek	10
2.2. A detektálás során használt műveletek	11
3. Szakirodalmi vonalkód detektáló algoritmusok	13
3.1. Alapvető képműveleteken alapuló eljárás	13
3.2. Futáshosszok számításán alapuló eljárás	15
3.3. Lokális küszöbölésen alapuló eljárás	17
4. Javasolt vonalkód detektáló algoritmusok	19
4.1. 1D-s vonalkódok detektálására javasolt módszer	19
4.2. QR kódok detektálására javasolt módszer	23
5. A tárgyalt algoritmusok összehasonlító elemzése	27
5.1. Tesztsorozat, tesztkörnyezet, implementáció	27
5.2. Eredmények vizsgálata	30
Összefoglalás	35
Irodalomjegyzék	36
Köszönetnyilvánítás	38

Bevezetés

Az automatikus azonosítás egyik legfontosabb és széles körben alkalmazott eleme a vizuális kódokkal történő azonosítás, mely a kereskedelmi folyamatoknak szinte elválaszthatatlan része (pl. UPC vonalkódok), és a mobil kommunikációnak köszönhetően a hétköznapi életben is egyre elterjedtebb (pl. QR kódokba ágyazott webcímek és leírások). A különböző szimbólumokkal megjelenített azonosítók teszik lehetővé a gépek számára az elektronikus leolvasást, ami nagyban segíti és gyorsítja az információ áramlását pl. a bolti pénztáraknál, raktári átvételnél, nagy sebességű feldolgozási helyeken pl. futószalagokon. Bizonyos alkalmazásokban egy-egy hibás detektálásnak komoly költségvonzatai is lehetnek. A kódok különbözősége és az alkalmazási területeik speciális problémákat vethetnek fel, így szükség van újabb és hatékonyabb eljárások kidolgozására.

Az első fejezetben tanulmányozzuk a manapság leggyakrabban használt 1D-s, illetve 2D-s vonalkód típusokat, azok vizuális jellemzőinek kiemelésével. A vonalkódok területén megvalósuló kutatásunk két évvel ezelőttig nyúlik vissza. Az első fejezetben ezeket az eredményeket is sorba vesszük. A második fejezetben áttekintjük azon alapvető képfeldolgozó műveleteket, melyekből felépíthetők a bemutatott algoritmusok. Néhány irodalmi módszer bemutatását tartalmazza a harmadik fejezet. A negyedik fejezet keretein belül bemutatjuk a saját eljárások működését. Az ötödik fejezetben végül összehasonlítjuk a tárgyalt algoritmusokat egy több ezer képből álló adatbázison, vizsgálva azok hatékonyságát. A tesztképek különböző típusú vonalkódokat, képenként egy vagy több vonalkódot tartalmaznak, illetve különféle torzításokkal terheltek.

1. fejezet

A vizuális kódok áttekintése

Ebben a fejezetben tárgyaljuk a napjainkban leggyakrabban használt 1D-s és 2D-s vonalkódokat, bemutatva azok főbb vizuális jellemzőit, melyekre az automatikus kód-detektáló módszerek is támaszkodhatnak. A dolgozat későbbi részében ismertetésre kerülő algoritmusok 1D-s, valamint QR vonalkód típusokra lettek kifejlesztve. A fejezetben az egyes vonalkód típusok ismertetését a [1] forrásra támaszkodva adtuk meg.

1.1. Történeti áttekintés

A vonalkód megjelenése az ötvenes évekre nyúlik vissza, viszont tömeges elterjedése csak 1973-ban indult meg. Bernard Silver és Norman Joseph Woodland nevéhez köthető az első vonalkód megjelenése, melyet az azt leolvasó rendszer kiépítése követett. A vonalkódok létrejötte a technika fejlődésével áll szoros összefüggésben, és nem a matematikai apparátus hiányától vagy rendelkezésre állásától tehető függővé.

A Morse kódot tekintik sokan a vonalkód elődjének, mivel Joseph Woodland és Bernard Silver ezek rajzolásából kapta az ötletet. Először arra gondoltak, hogy Morse kódokat használjanak horizontálisan nyomtatva, létrehozva keskeny és széles vonalakat, de ezt később elvetve megalkották az első vonalkódot, mely koncentrikus körökből állt. Ez vékonyabb és vastagabb körökből álló kód, ahol a vonalak és a vonalközök voltak az információhordozók, ezeket bármilyen irányból, bármilyen fénnel le lehetett olvasni. 1835-ben dolgozta ki Samuel Finley Breese Morse a rendszerét, melynek alapjául a ritmus szolgált, vagyis az idő felosztása, tagolása. Morse az angol ábécé betűit feleltette meg a rövid (pont), illetve a hosszú (vonat) jeleknek, és az ezek kombinációjából kialakított jelcsoportoknak. Ez a megfeleltetés nem esetleges volt, hanem a betűk előfordulásának gyakorisága szerint történt: a leggyakoribb „E” betű kapta például a legegyszerűbb jelet, a pontot. A vonalkódban is a Morse rövid-hosszú jelsorozataihoz hasonlóan a különböző szélességű sávokat alkalmaztak.

Egy konkrét példát tekintve az „á” betű kódja

jel - szünet - szünet - jel - szünet, számjegyekkel írva: 10010

A vonalkódokat először vasúti kocsikon, majd az amerikai posta járművein próbálták ki. Az évek során a különböző fejlesztések következtében a vonalkód rengeteg változatban megjelent, olyanok is, amelyek már szöveget is tudnak tárolni, sőt a beléjük kódolt információ is maga egy kód, amelynek megfejtéséhez szükséges az adott kereskedelmi egység adatbázisához való hozzáférés. Magyarországon az 1990-es években terjedt el, de a Skála már 1984-ben hasznosította a vonalkódtechnika nyújtotta előnyöket.

1.2. Főbb jellemzők

A vonalkódos azonosítás széles körben elterjedt a mindennapi életünkben. Számos eljárás létezik azok gyors és hatékony azonosítására. A legtöbb eljárás a kódok lokalizálására, az alkotó komponensek pozícióinak és jellemzőinek meghatározására automatikus szegmentálási technikát alkalmaz.

Vonalkód kifejezés alatt különböző típusú vizuális kódokat értünk. Jelen dolgozatban az 1D-s és QR kódok szegmentálásával fogunk foglalkozni. A vonalkódok nem olvashatóak az emberek számára, ezt az erre alkalmas dekódoló eszközök valósítják meg. A hagyományos vonalkódok felépítése azonos: egymással párhuzamos, különböző vastagságú világos és sötét közök, illetve vonalak meghatározott váltakozása fejezi ki az információt, mely gépek által optikailag leolvasható kód. Az egyes vonalkódok abban különböznek egymástól, hogy az adott karakternek milyen fekete és fehér vonalakból álló sorozatot feleltethetünk meg.

A felépítése az alábbi struktúrát követi: a vonalkód elején egy nyugalmi zóna található, ezt követi a start karakter, egy vagy több adatkarakter, opcionálisan egy vagy több ellenőrző karakter, majd a stop karakter, végül a hátsó nyugalmi zóna. Egy kód karakterkészlete azt adja meg, hogy az adott kódrendszer segítségével milyen karakterek kódolhatóak (numerikus, alfanumerikus, kis- és nagybetűk, vezérlő karakterek). A vonalkód modulmérete a vonalkódot alkotó legkeskenyebb vonal fizikai szélességét határozza meg, és általában a hüvelyk ezredrészében adják meg.

A hagyományos vonalkódok használata során felszínre kerültek az előnyeik és a felhasználók körében egyre több alkalommal igény mutatkozott olyan kódokra, melyek információ tárolására alkalmasak, és kisebb méretben képesek eltárolni több információt. Az 1D-s vonalkódok felépítésükben kötöttek, így ezen típusok esetében erre nincs lehetőség. Ezek következtében alakították ki a 2D-s kódokat.

1.3. Felhasználási terület, csoportosítás

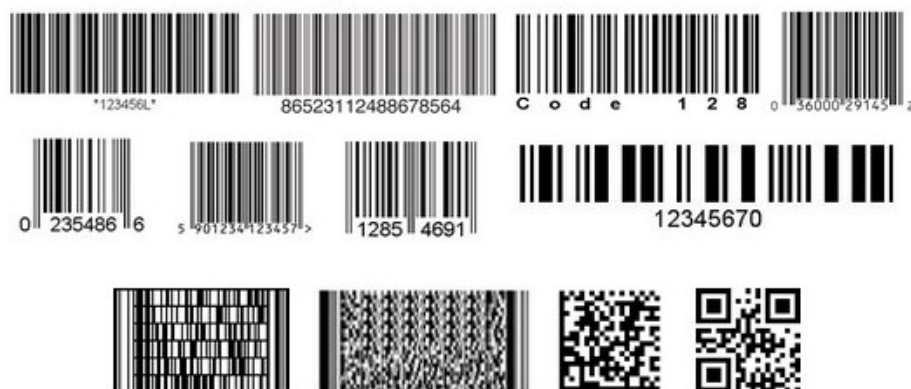
A vonalkódok leggyakoribb felhasználási területe a kereskedelem, például az áruk csomagolásán, amely lehetővé teszi az áru adatainak, úgy mint a gyártó országa, a gyári szám és a termék cikkszám gyors azonosítását. A vonalkódos azonosítás a GS1 rendszer legismertebb eleme, mely a kereskedelmi folyamatok elválaszthatatlan része. A vonalkód jelképekkel megjelenített azonosító számok teszik lehetővé a gépek számára az elektronikus leolvasást, ami nagyban segíti és gyorsítja az információ áramlását. A vonalkódok lehetnek fix vagy tetszőleges hosszúságúak. Fix hosszúságú kódok esetében a szabványban meghatározásra kerül, hogy milyen hosszú karaktersorozat kódolására képes. Ez a tetszőleges hosszúságú kódok esetén nem meghatározott.

A vonalkódok tulajdonságaik alapján csoportosíthatók. Lehetnek bináris vagy delta, diszkrét vagy folytonos, illetve egy vagy több jelentésű kódok a különböző osztályozások alapján. A bináris kódok esetében a széles vonal és vonalköz 1-est, míg a keskeny vonalak illetve vonalközök 0-t jelentenek. A delta kódok közös jellemzője, hogy a fekete vonalakhoz a szélességüknek megfelelő darabszámú 1-est rendelünk, a vonalközökhöz hasonló szempontok alapján 0-akat. Egy másik osztályozás szerint diszkrétnek nevezzük azokat a vonalkódokat, ahol a karaktereket elválasztó vonalközök nem hordoznak információt. Folytonos kódok esetén a karaktereket elválasztó vonalközök is értékes információt hordoznak. További osztályozási lehetőséget nyújt annak vizsgálata, hogy az adott vonal- és vonalközsorozat egyértelműen meghatároz-e egy adott karaktert vagy sem. A vonalkódok nagy része egyjelentésű, ami azt jelenti, hogy egyértelműen hozzárendelhető egy karakter egy vonalából álló mintázathoz. A többjelentésű kódok több karakterkészletet kódolnak, melyek közt választhatunk vezérlőkértékek segítségével. A 1.1 ábra a főbb 1D-s és 2D-s kódokat szemlélteti.

A 2D-s kódoknak két csoportját különböztetjük meg a felépítésük alapján, a halmazozott szerkezetű, és a mátrix kódokat. Szerkezetüket tekintve nem vonalából állnak, hanem sötét és világos cellákból építkeznek, melyek elrendezése mátrixszerű alakzatot követ. A QR kód a mátrix kódok csoportjába tartozik.

1.4. Azonosítás

A vonalkód lokalizáló eljárásoknak két fő célja a sebesség és a hatékonyság. Ipari környezetben a hatékonyság kritikus tényező. Egy-egy hibás detektálásnak komoly költségvonzatai lehetnek. A feldolgozási sebesség a másodlagos elvárt összetevője a detektornak. Mobil készülékek esetében a hatékonyság kevésbé kritikus azóta, amióta a felhasználó és a telefon interakcióban vannak egymással, így újabb és újabb felvételek készülhetnek a detektálni kívánt kódról. Tehát, ez esetben egy gyors procedura a kívánatos. Számptalan



1.1. ábra. Főbb vonalkód típusok (balról jobbra). Felső sor (1D kódok): Code39, Codabar, Code128, UPC-A; Középső sor (1D kódok): UPC-E, EAN-13, EAN-8, I2of5; Alsó sor (2D kódok): Codablock, PDF417, Data Matrix, QR.

technika használható a vonalkódok digitális képekről való lokalizálására és dekódolására a vonalszkenneléstől [2, 3] a morfológiai megközelítéseken keresztül [4, 5, 6, 7] a konvex burkon át [8] egészen a waveletek-ig [9].

1.5. Kapcsolódó munkák

Az automatikus azonosítás különféle vonalkód típusokkal történő megvalósításával több éve foglalkozunk. A kitűzött cél a korábbi munkákban is a hatékonyság maximalizálása volt, tehát gyors, de ugyanakkor pontos eljárások kidolgozása. A vizsgálataink a morfológiai műveleteket alkalmazó szakirodalmi és saját fejlesztésű algoritmusokra terjedtek ki. Célzottan egyszerű és gyors eljárások kerültek kiválasztásra, melyek a [10, 11, 12] cikkekben szereplő algoritmusok voltak. Ezen módszerek implementálásra kerültek, hogy összehasonlíthatóvá válhassanak az általunk fejlesztett eljárásokkal.

A kidolgozott eljárások morfológiai völgydetektáláson alapszanak, melynek alkalmazásával a vonalkódot alkotó párhuzamos sávok kiemelhetők. A konkrét detektálási fázisban bináris képpel történnek a képmanipulációs műveletek. A [6]-ben ismertetett módszerben a binarizálást követően, a horizontális pásztázást a fals régiók csökkentésére terület alapú küszöbölés követte. Az így kapott képen különböző szerkesztőelemmel előbb dilataciót, majd eróziót hajtottunk végre, mely műveleteket amiatt alkalmaztunk, hogy az összetartozó szegmensek egy régiót alkossanak, de azon területek, melyek szükségtelenül összeolvadtak, szétválasztásra kerüljenek. Végezetül, annak kiszűrésére, hogy csak valószínűleg vonalkód régiók legyenek az eredményképen, újabb terület alapú küszöbölést alkalmaztunk.

A [7] cikkben található módszer az előzőtől abban tér el, hogy a bináris kép előállítását

követő lépéssorban, valamint a felhasznált szerkesztőelemek alakjában és méretében. A binarizált képre távolságtérképet számolva, majd a soronkénti átlaggal küszöbölve jól szegmentálható a vonalkód régió, mivel az azt alkotó sávok közel helyezkednek el egymáshoz. A további lépések megegyeznek az előzőleg ismertetett eljárásával.

2. fejezet

Az alkalmazott képfeldolgozó műveletek áttekintése

A fejezet keretein belül ismertetjük az előfeldolgozás és detektálás során használt képműveleteket, melyek a következő fejezetben bemutatott eljárások során felhasználásra kerültek.

2.1. Előfeldolgozó műveletek

Az ismertetésre kerülő algoritmusok az előfeldolgozás során Gauss-simítást, különféle szűrőket, kontrasztszéthúzást, hisztogram kiegyenlítést, élesítés alkalmaznak.

Gauss simítás alatt a digitális képnek egy Gauss-szűrő függvénnyel vett konvolúcióját értjük,

$$h = f * G_{\sigma}, \quad (2.1)$$

ahol az f a kiindulási kép a G_{σ} a Gauss-szűrő, a h a simított kép, illetve a $*$ a konvolúció műveletét jelöli [13]. Ezenfelül meg kell adni szórát is, mely a Gauss-függvény egy paramétere, ezt a σ jelöli, mely az elmosás mértékét befolyásolja.

Egy másik, képjavításra alkalmas művelet a képélesítés. Az élesítésre különféle lehetőségek adóttak. Jelen dolgozatban úgynevezett életlen maszkkal tettük mindezt. Ez egy finoman befolyásolható szűrő. Először egy életlen képet készít a felvételtől, majd ezt kivonja az eredetiből. Három paraméterrel rendelkezik, melyek mindegyike az outputot határozza meg. Megadható az élesítés mértéke, egy küszöbérték és az életlenség mértéke is. Minél nagyobb értéket állítunk be az élesítés értékének, annál élesebb képet eredményez. A küszöbérték azt határozza meg, hogy mekkora kontrasztugrást, azaz mennyire éles határvonalat tekintsen minimumnak a szűrő az élesítés alatt. Így a kieresződő szemcsézetet, vagy a zajt kimaszkolhatjuk a szűrő hatásköréből. Minél nagyobb értéket

adunk meg, annál hatékonyabb lesz az élesítés.

A kontraszt széthúzás az intenzitások egy megadott $[o_{min}, o_{max}]$ intervallumát skálázza egy $[n_{min}, n_{max}]$ intervallumba:

$$s = (r - o_{min}) \frac{n_{max} - n_{min}}{o_{max} - o_{min}} + n_{min}, \quad (2.2)$$

ahol s az új intenzitás, r a régi intenzitás. A kontraszt széthúzás helyett hisztogram kiegyenlítés is alkalmazható.

Hisztogram kiegyenlítés során a képet úgy transzformáljuk, hogy az világosságértékek egyenletes eloszlásúvá váljanak. Ez az eljárás kevésbé kontrasztos képek esetében hasznos, amikor az intenzitásértékek az intenzitástartomány egy részére koncentrálnak.

Az előfeldolgozás során számos műveletsorban szokás alkalmazni különféle szűrőket. A dolgozatban ezek közül kettő kerül felhasználásra, a LoG szűrés, valamint egy lokális szórás számításra alapuló szűrő eljárás.

A LoG (Laplacian of Gaussian) operátor a szűréssel együtt élkimelést is végez. A Laplace operátor definíciója az alábbiaképpen írható fel:

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}, \quad (2.3)$$

ahol az $f(x, y)$ a Laplace szűrt értéket jelenti az adott pontban. A Laplace szűrő az intenzitásváltozásokra, hibákra is nagyon erősen reagál. Simítás nélkül, önmagában nem túl eredményes, tehát előtte Gauss simítás alkalmazása a szűrt érték kiszámítása előtt éldetektort eredményez. A konvolúció asszociativitásából következően:

$$\nabla^2(f * G_\sigma) = \underbrace{(\nabla^2 G_\sigma)}_{\text{LoG}} * f. \quad (2.4)$$

Lokális szűrők estében az adott pixel egy meghatározott környezetének vizsgálata következik be, mely alapján kerül kiszámításra annak új értéke. Jelen dolgozatban egy adott képpont 3×3 -as környezetében lévő intenzitásértékek szórása határozza meg a pixel értékét.

2.2. A detektálás során használt műveletek

Az ismertetésre kerülő eljárások nagy része magában foglal valamiféle morfológiai operációt. A morfológiai műveletek bemenete egy ponthalmaz és egy elemi minta, a szerkesztőelem. A műveletek tárgyalását két olyan módszerrel kezdjük, amik a későbbi összetett eljárások alapelemei. Ez a két művelet duális tulajdonságú, így az egyik a másikból származtatható, de az alaphalmazra különböző hatással bírnak, ennek következtében külön

definiálják őket.

Legyen $A = [a(i, j)]$ egy többszintű kép és jelölje az S szerkesztőelem tartományát (S -be eső pontok halmazát) D_s . Ekkor a $B = A \oplus S = [b(i, j)]$ többszintű képet eredményező dilatációt a következő formula írja le:

$$b(i, j) = \max_{(u, v) \in D_s} \{a(i - u, j - v)\}. \quad (2.5)$$

A $C = A \ominus S = [c(i, j)]$ többszintű erózió formulája:

$$c(i, j) = \min_{(u, v) \in D_s} \{a(i + u, j + v)\}. \quad (2.6)$$

Minták keresésére használható a mintaillesztés. [14] Az alapvető módszert tekintve adott egy konvolúciós maszk, ami a minta, és ennek helyét szeretnénk kimutatni a keresési képtérben. A konvolúciós kimenet értéke azokon a pozíciókon lesz magas, ahol az illesztett minta és az input kép megegyeznek. A módszer implementálása általában úgy történik, hogy a keresési képtér egy adott részét használják sablonnak.

3. fejezet

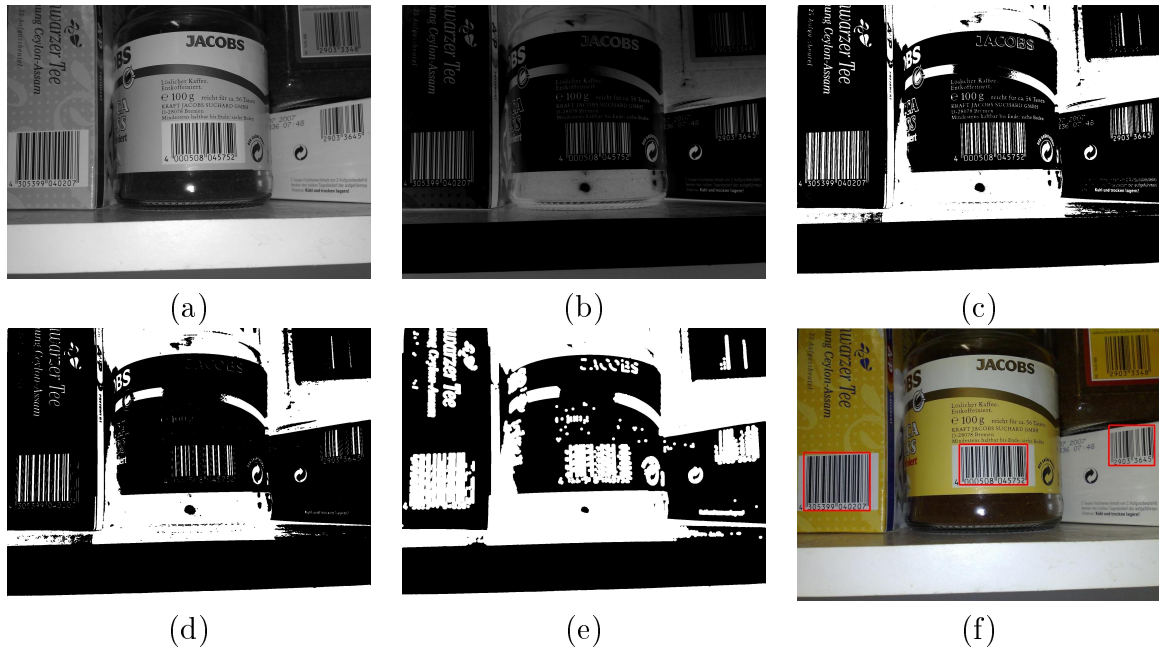
Szakirodalmi vonalkód detektáló algoritmusok

3.1. Alapvető képműveleteken alapuló eljárás

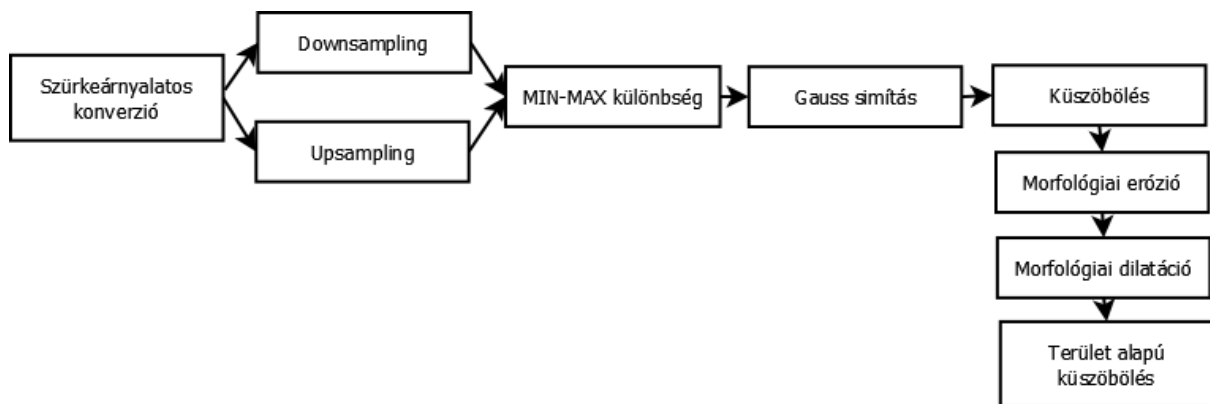
Ezen alfejezet keretein belül Daw-Tung Lin és Chin-Lin Lin [15] cikkében szereplő algoritmus kerül ismertetésre. Az eljárás alapvető képfeldolgozó műveleteken alapszik. Az előfeldolgozás során első lépés a bementi kép szürkeárnyalatossá kvantálása. Ehhez az általános konverziót alkalmazták a szerzők, tehát az egyes színcsatornák különböző súllyal vett összegei adják a szürkességi értéket. Ezt a konverziót mutatja a 3.1(a) ábra.

A vonalkódot alkotó sávok és a háttér közt nagy kontrasztbeli különbség mutatkozik. Emiatt bármilyen élkiemelő művelet alkalmazása javíthatja a képminőséget. A cikk írói a többszintű kép alul-, illetve túlmintavételezett eredményképek különbségének számításával tették mindezt. Az egyes pixelértékek kiszámítása ablakozásos technikával történt. Az alulmintavételezett képen és túlmintavételezett képen 4×4 -es blokkokat véve, az adott blokkban az alulmintavételezett minimum kivonásra kerül a túlmintavételezett maximumból. Ennek eredménye látható a 3.1(b) ábrán. Az így kapott élkiemelt képen zajszűrés céljából Gauss simítást végeztek, majd megszabva egy 33-as konstans értéket, küszöbölték, előállítva ezzel a bináris képet, mely a detektálási folyamat alapjául szolgál. A 3.1(c) ábra illusztrálja ezt a lépést.

Az előfeldolgozást követően a tényleges vonalkód területek feltérképezése a következő lépés. A képen előfordulhatnak kis területű objektumok, melyek biztosan nem vonalkód szegmensek, ezért eltávolíthatóak, illetve összeérhetnek olyan objektumok, melyek nem egy régióba tartoznak. Ehhez morfológiai eróziót alkalmaztak az eljárás kidolgozói 5×5 -ös négyzet alakú szerkesztőelemmel. Mivel ez szétszakíthat összetartozó területeket is, ezért, hogy ezek mégis egy objektumot alkossanak, morfológiai dilatációt hajtottak végre 5 sugarú, kör alakú szerkesztőelemmel. Ezen két művelet végrehajtásának eredményét



3.1. ábra. Az alapvető képfeldolgozó műveleteket alkalmazó eljárás köztes lépéseinek eredményképei. (a) szürkeárnyaltos kép, (b) min-max különbségkép, (c) küszöbölés eredménye, (d) morfológiai erózió, (e) morfológiai dilatáció, (f) terület alapú küszöbölést követő végső eredménykép



3.2. ábra. A lokalizálás folyamatának grafikus megjelenítése

szemlélteti a 3.1(d), illetve 3.1(e) ábra.

Az eljárás végső lépésében terület alapú küszöböléssel a vélhetően fals régiók törlésre kerülnek. Minden szegmens területe meghatározásra kerül, majd minden olyan objektum eltávolítása szükséges, melynek területe kisebb, mint 725 pixel. Az eljárás lépéseit összefoglaló folyamatábra a 3.2 ábrán látható.

3.2. Futáshosszok számításán alapuló eljárás

Az alábbiakban ismertetett módszerben Daw-Tung Lin és Chin-Lin Lin [16] olyan eljárást alkottak meg, mely a futáshosszok számításán alapszik.

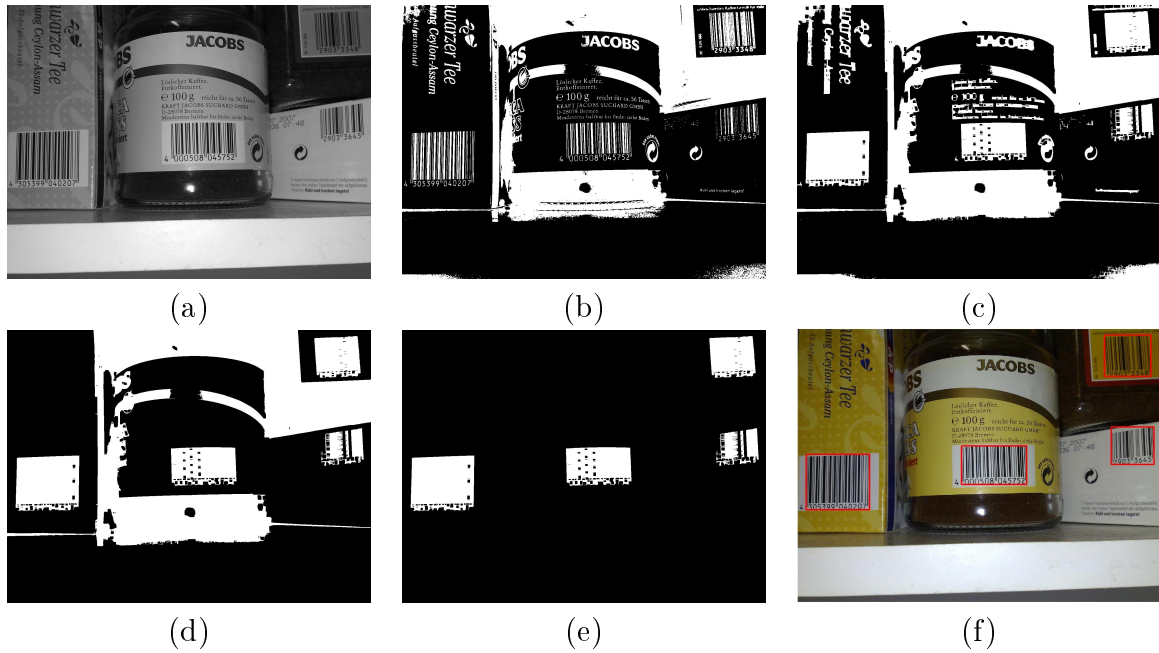
Az algoritmus az inputban kapott képet amennyiben az nem szürkeárnyaltos, előbb kvantálással azzá alakítja. Hasonlóan az előző alfejezetben ismertetett eljáráshoz, úgy itt is az egyes színcsatornák különböző súllyal vett összegei adják a szürkeségi értéket. A binarizált kép előállításához, mely szükséges a további feldolgozáshoz, iteratív küszöbölést alkalmaztak, melynek lépései:

1. T küszöbérték inicializálása
2. A kép intenzitástartományának két részre particionálása. G_1 tartalmazza azokat a pixeleket, melyek értéke nagyobb, mint T és G_2 tartalmazza az összes többi pontot
3. T_1 és T_2 a G_1 , illetve G_2 átlagintenzitásainak meghatározása
4. Új T érték számítása a következőképpen: $T = (T_1 + T_2)/2$
5. A 2-4 lépések ismétlése, míg a T értéke nem kisebb, mint egy előre definiált T_0 értéke

Amennyiben a T és T_0 értékekre előírt elvárás teljesül, a kapott T értékkel küszöböljük a képet. Minden olyan pixel előtérpont lesz, melynek intenzitása kisebb vagy egyenlő, mint a számított T érték. Ezzel megkapható a binarizált eredménykép. A 3.3(b) ábra mutatja a küszöbölés eredményét.

A következő lépés határolja be leginkább, hogy mely területek lehetnek lehetséges vonalkódot alkotó sávterületek. Úgynevezett `mod_RLSA` algoritmust dolgoztak a cikk szerzői, melyben futáshosszok kerülnek kiszámításra. A művelet sor az alábbi lépésekből épül fel:

1. Szabályalkotás minden sorra. Legyen R_1 a szabályok által meghatározott output kép
 - azon 0 értékkel rendelkező pixelek értéke 1, melynél az egymást követő 0-ák száma $\leq$ egy előre definiált C konstans
 - ha 1, akkor változatlan.
2. Az 1. lépésben megadott szabályok kiterjesztése az oszlopokra. Az így kapott kép legyen R_2



3.3. ábra. A futáshosszok számításán alapuló eljárás köztes lépéseinek eredményképei. (a) szürkeárnyaltos kép, (b) iteratív küszöbölés eredménye, (c) futáshosszok számítása, (d) terület alapú küszöbölés, (e) szélesség, magasság alapú küszöbölés eredménye, (f) eredménykép

3. Az R_1 és R_2 képek kombinációja az alábbiaképpen:

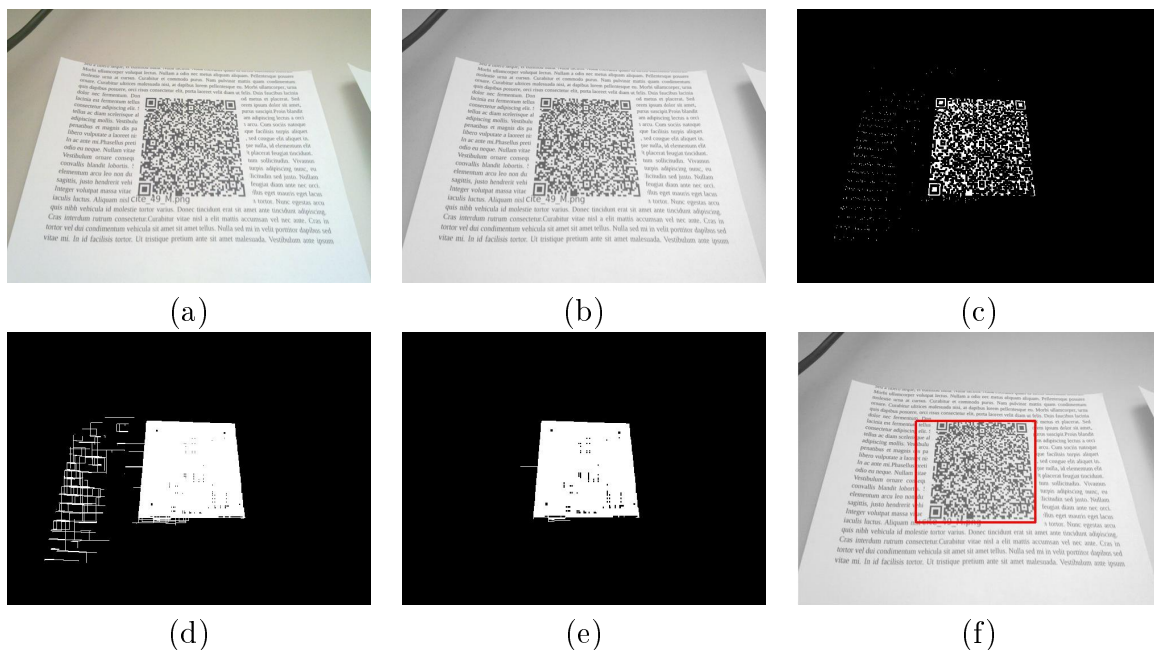
$$R(x, y) = \begin{cases} 1 & R_1(x, y) = 1 \text{ és/vagy } R_2(x, y) = 1, \\ 0 & \text{egyébként.} \end{cases}$$

Az eredményt a 3.3(c) ábra szemlélteti.

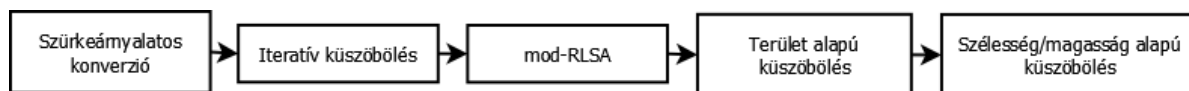
A futáshossz számítással kapott képen még előfordulhatnak nem valós vonalkód területek, ezért a keresési tér szűkítésére terület alapú küszöbölést alkalmaztak. Minden olyan régió eltávolításra került, melynek területe nem haladta meg a 300 pixelt, mint ahogyan ez látható a 3.3(d) ábrán is.

Ismeretes, hogy a vonalkódok négyzet vagy téglalap alakúak lehetnek, emiatt az algoritmus utolsó lépésében megvizsgálásra kerül a régió szélessége, illetve magassága. Abban az esetben, ha az objektum szélessége ≤ 512 pixelnél és a magassága ≤ 384 pixelnél, akkor azt vonalkódterületnek tekinti az eljárás, egyéb esetben eldobásra kerül. Ennek eredményét illusztrálja a 3.3(e) ábra.

Az eljárás 2D-s kódok lokalizálására is alkalmas, ahogyan a 3.4 ábra is mutatja. Az algoritmus működését összefoglaló folyamatábra látható a 3.5 ábrán.



3.4. ábra. A futáshosszok számításán alapuló eljárás lépéseinek eredményképei. (a) input kép, (b) szürkeárnyaltos konverzió (c) iteratív küszöbölés, (d) futáshosszok számítása, (e) terület alapú küszöbölés, (f) eredménykép



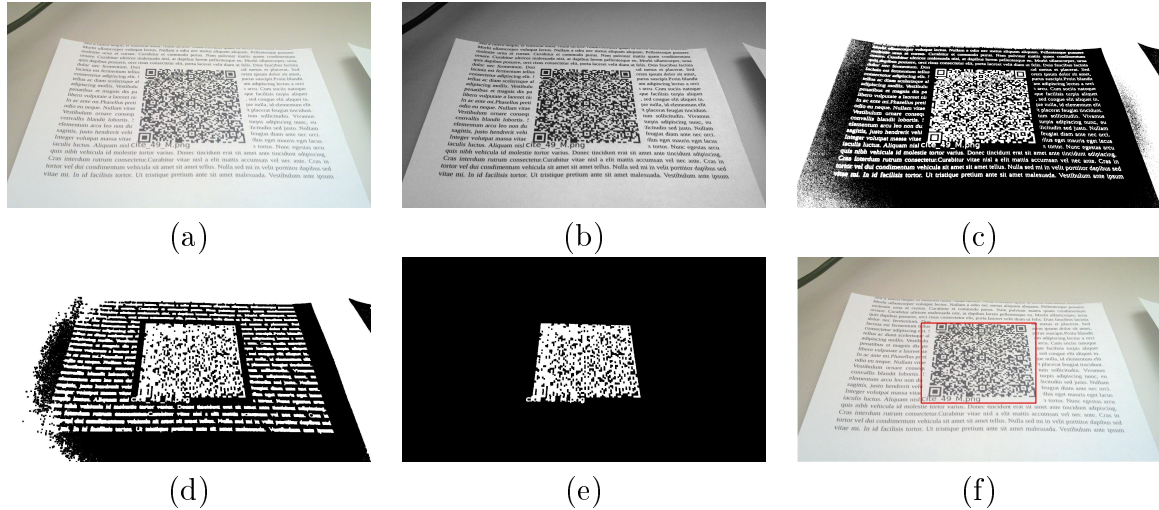
3.5. ábra. A lokalizálás folyamatának grafikus megjelenítése

3.3. Lokális küszöbölésen alapuló eljárás

Ezen alfejezetben bemutatásra kerül a [17] cikkben kidolgozott vonalkód lokalizáló módszer. A cikkben a szerzők több fázisra osztották az elvégzendő feladatokat, melyek közül a detektálás folyamata a előfeldolgozás szakaszában definiált. Fontos megemlíteni, hogy a módszer QR kódok felismerésére alkalmas.

A folyamat első lépéseként YUV színtérkonverziót hajtottak végre a bemeneti képen, melynek az Y (fényesség) csatornája került felhasználásra többszintű kép előállítására 3.6(b).

Az eljárás legfontosabb lépése a megfelelő küszöbérték megválasztása a bináris kép létrehozásához. Mindez úgy történik, hogy a kép középpontjához legközelebb elhelyezkedő területen 9 darab 60×60 -as területű blokkot határozunk meg. Ezen blokkok átlagos intenzitásértékeinek minimuma adja a küszöbértéket. Tehát minden olyan pixel előtérpont lesz, melynek a szürkességi értéke a jól meghatározott küszöbszám feletti értéket veszi fel. A küszöbölt eredményképet szemlélteti a 3.6(c) ábra.



3.6. ábra. A lokális küszöbölésen alapuló eljárás köztes lépéseinek eredményképei. (a) bemeneti kép, (b) szürkeárnyaltos konverzió (c) küszöbölés eredménye, (d) morfológiai dilatació, (e) terület alapú küszöbölés, (f) eredménykép



3.7. ábra. A lokalizálás folyamatának grafikus megjelenítése

A binarizálást követően újramintavételezésre kerül a kép, a képméret felére való redukálással. Ezt követi a morfológiai nyitás 7×1 -es szerkesztőelemmel (3.6(d)). Az így kapott eredménykép tartalmazhat fals szegmenseket, ezért a szerzők terület alapú küszöbölést alkalmaztak, melynek kimenetele a 3.6 (e) ábrán látható. A bemeneti képen behatárolt QR kód régiót illusztrálja a 3.6(f) ábra. Összefoglaló folyamatábra látható a 3.7 ábrán.

4. fejezet

Javasolt vonalkód detektáló algoritmusok

Ebben a fejezetben két vonalkód detektáló algoritmust tárgyalunk. Mind a két procedúra más módszert vesz a helymeghatározás alapjául.

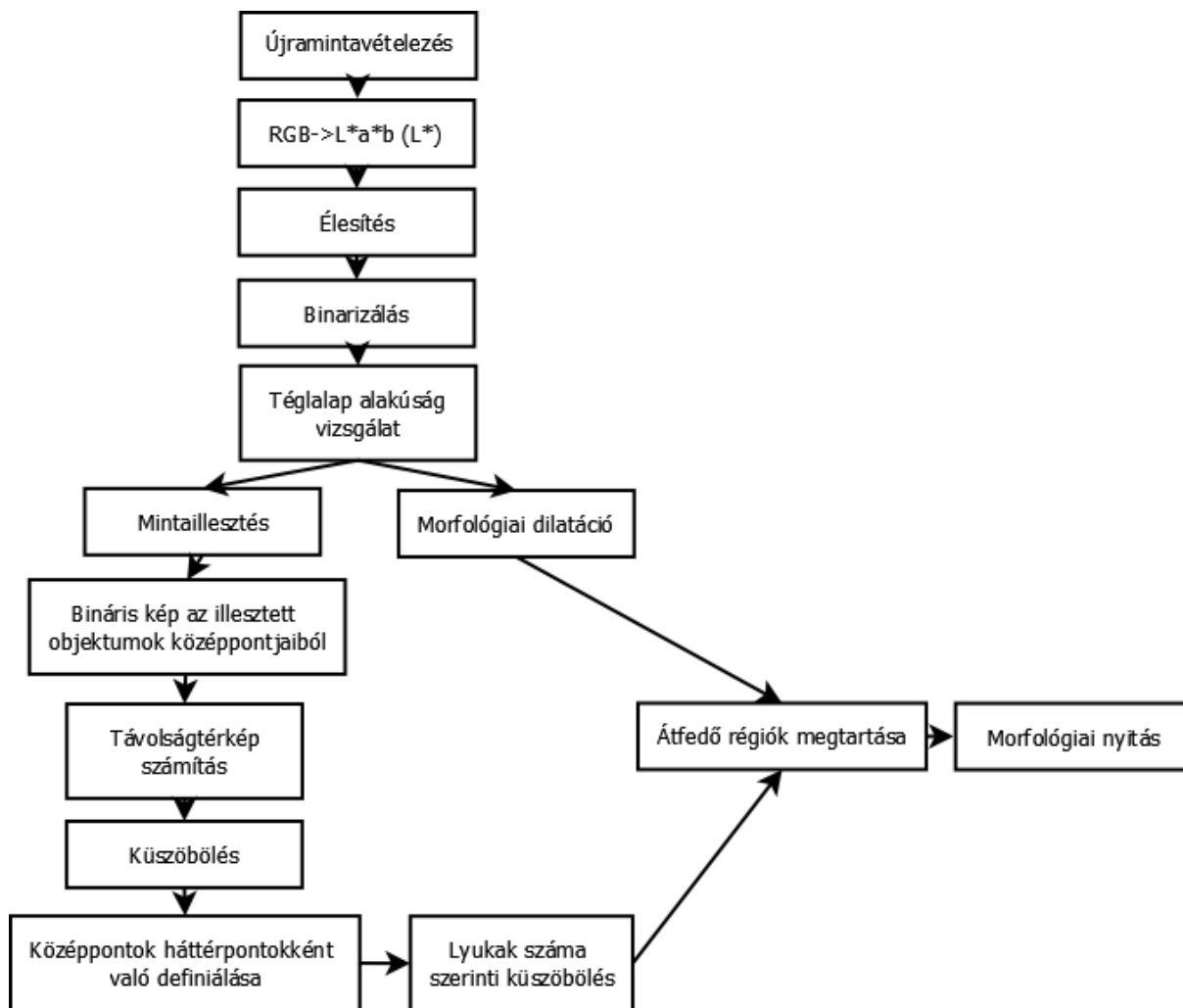
A digitális képek minősége manapság kevésbé kifogásolható, bár gyenge minőségű felvételek is készülnek. A minőségromlás oka pl. lehet maga a digitalizáló eszköz, illetve a környezet okozta nehézségek. Mindezek következményeként gyakran szükséges a kép minőségének javítása a konkrét feldolgozás előtt.

A tárgyalásra kerülő algoritmusok mindegyike szürkeárnyaltos, majd a detektálás fázisában bináris képpel dolgozik. Ennek az oka, hogy amennyiben az algoritmusok robusztusságát kívánjuk növelni, akkor a színintenzitások figyelmen kívül hagyhatóak, mivel manapság már színes kódok is használatban vannak és ha adott színtartományra szűkítünk, akkor információt, jelen esetben kódszegmenst veszíthetünk.

4.1. 1D-s vonalkódok detektálására javasolt módszer

Az eddigi alfejezetekben egy-egy szakirodalmi eljárást ismertettünk. Most egy új, saját fejlesztésű algoritmust mutatunk be 1D-s vonalkódok lokalizálására. Az algoritmus alapjául a mintaillesztés szolgál, melynek más műveletekkel való kombinálásával hatékonyan állapítható meg diszkrét képeken a vonalkódok helye. Az eljárás lépéseit összefoglaló folyamatábra a 4.1 ábrán látható.

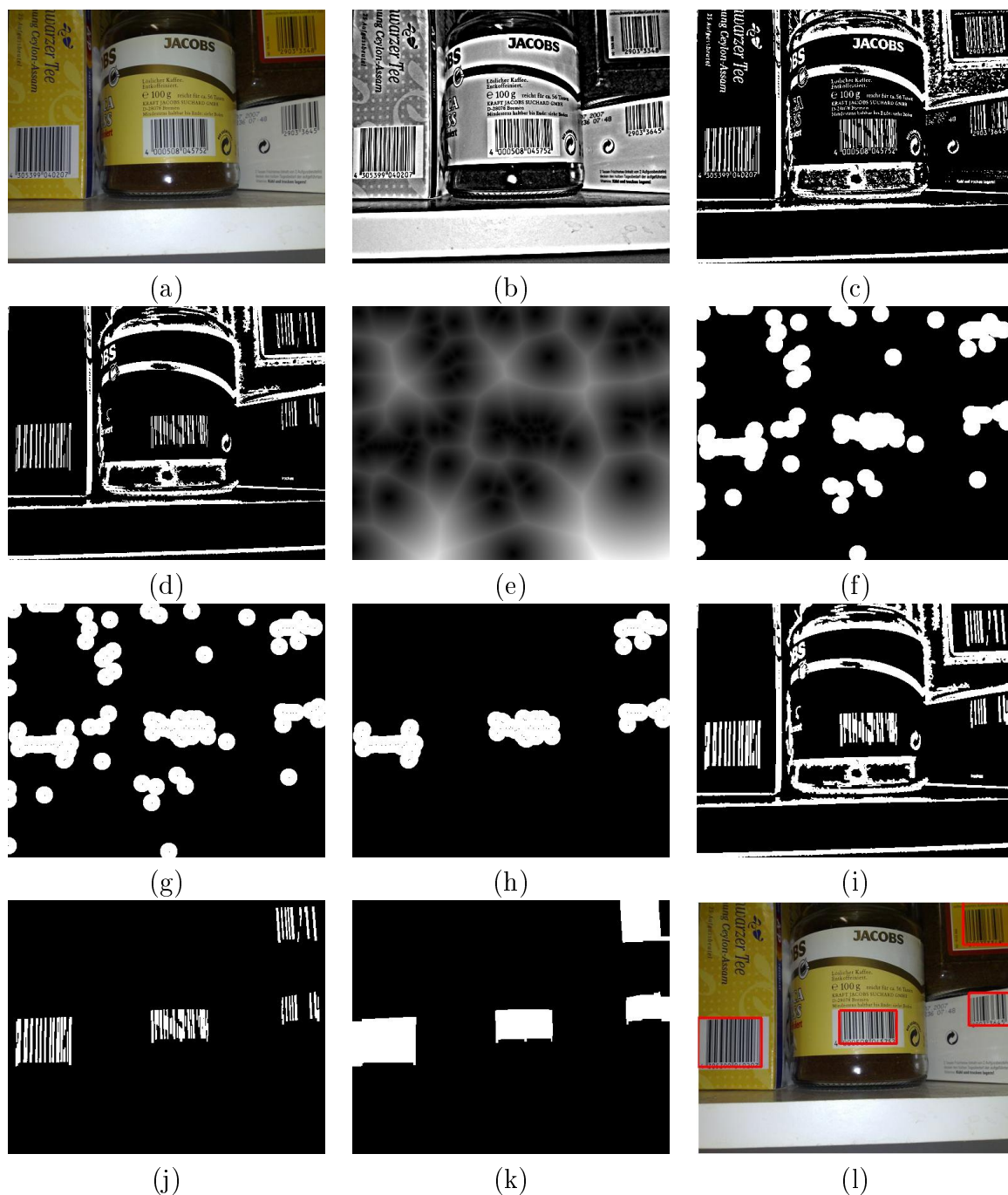
Ismeretes, hogy a mintaillesztés elvégzése időigényes feladat, így a műveletsor első lépéseként újramintavételeztük a képet. A képméretet konstans 400×500 -as méretűre redukáltuk. Ez a képméret már lehetővé teszi, hogy érdemleges szegmentálási eredményhez jussunk, illetve megadhatóak hozzá olyan, a feldolgozáshoz használt konstans értékek, melyekkel jól behatárolhatóak különféle méretű, orientáltságú, darabszámú vonalkódok



4.1. ábra. A lokalizálás folyamatának grafikus megjelenítése

és a futási idő is nagyban csökkenthető. Empirikus tapasztalatok során bebizonyosodott, hogy a L^*a^*b színtérkonverzióval a későbbi fázisokban jobb eredmények érhetőek el. Tehát az újramintavételezést követően a kapott RGB képet $L^*a^*b^*$ [13] színtérbe konvertáltuk. A későbbi feldolgozásban már csak az L^* (fényesség) csatornát használjuk és az így kapott többszintű képpel dolgozunk. Az inputban kapott képek nem ideálisak, emiatt képjavító módszer alkalmazása javasolt. Jelen esetben élesítettük a bemeneti képet, melynek eredményét a 4.2(b) ábra szemlélteti.

A detektálás folyamata bináris képekre épül, ezért az élesített képen meghatározva egy globális küszöbértéket, azt binarizáltuk. A megfigyelések azt mutatták, hogy a vonalkódot alkotó párhuzamos sávok intenzitásértéke alacsony, így a küszöbszámot konstans 1-ben határoztuk meg. Ennek eredménye látható a 4.2(c) ábrán. A képek minőségét a nem megfelelő felvétel során zaj is csökkentheti. A binarizált képen ezek kiszűrésére méret alapú küszöbölést végeztünk. Mindez, a képen megjelenő kis területtel rendelkező



4.2. ábra. A javasolt módszer köztes lépései. (a) input kép, (b) élesítés után, (c) binarizálás, (d) téglalap alakúság vizsgálat, (e) távolságtérkép, (f) küszöbölt távolságtérkép, (g) illesztett pontok az objektumokban, (h) lyukak szerinti küszöbölés, (i) a (d) eredményképe dilatació után, (j) a (h) utáni illeszkedő objektumok az (i) képen, (k) morfológiai nyitás, (l) detektált kódok



4.3. ábra. A mintaillesztéshez használt sablon

régiókat eltávolítja, szűkítve ezzel a lehetséges vonalkód területek számát és a könnyítve a lokalizálást. Ezt követően úgynevezett téglalap alakúságot vizsgáltunk, mivel ismeretes, hogy a vonalkódot alkotó sávok téglalap alakúak. Ehhez az egyes objektumok szélességi, illetve magassági paramétereinek egymáshoz viszonyított értékeit tekintettük. Abban az esetben, ha a két érték közötti különbség egy adott határt meghaladt, úgy az adott objektum eldobásra került. A mintaillesztéshez előkészített képet a 4.2(d) ábra illusztrálja.

Ahogy az 1. fejezetben is említettük, a vonalkódokat meghatározott felépítés jellemzi. Emiatt használható hatékonyan a mintaillesztés ezen alakzatok keresésére a képtérben. Annak analógiájára, hogy a kód sávjai párhuzamosak, az illesztéshez használt minta is két párhuzamos vonalat tartalmaz. Ezt láthatjuk a 4.3 ábrán. A sablon eredeti mérete 90x390. A sablonillesztést megelőzően ezt újrámintavételeztünk egy kisebb mintapéldány készítésére.

A sablonillesztés a képméret és a minta mérete függvényében növelheti a futási időt képtérben. Ennek gyorsítására frekvenciatérben is elvégezhető a művelet, mivel a képtérben történő konvolúció ekvivalens a frekvenciatérben történő pontonkénti szorzással. Ehhez a bemeneti szürkeárnyaltos kép és az illesztendő minta Fourier térben vett képe szükséges, majd a ezeknek pontonkénti szorzata határozza meg, hogy mennyire illeszkednek. A mintát 170°-ig 10°-onként tekintve és mindezek összegezett eredményét vettük figyelembe a további feldolgozásban, tehát

$$\sum_{i=0}^{17} FFT(I) \cdot FFT(R_i * T), \quad (4.1)$$

ahol I az input kép, R_i ($0 \leq i \leq 17$) az aktuális forgatómátrix, T az illesztendő képet jelöli. FFT alatt pedig a megadott képek gyors Fourier transzformáltját értjük. Amiatt elegendő 170°-ig venni, mert használt minta szimmetrikus.

A kapott összegképet az átlaggal küszöböltük, majd az így kapott objektumok közép-pontjainak koordinátáit eltároltuk.

Az újrámintavételezett kép méretével azonos, új, bináris képet hoztunk létre, melyben az objektumpontok az előző lépésben eltárolt középpontok. Az eredményképen az egy klaszterbe tartozó pontok jól elkülönültek, így a megadott képre távolságtérképet számoltunk. Ennek eredménye látható a 4.2(e) ábrán. A pontok egymástól vett távolságát

figyelembe véve küszöbölést alkalmaztunk, amit követően olyan szegmensek maradtak a képen, melyekben az azokat alkotó pontok közel helyezkedtek el egymáshoz. Annak meghatározására, hogy az adott objektum területén az eredeti képen vonalkód helyezkedik-e el, előbb az eltárolt illesztésből eredő középpontokat háttérpontokként definiáltuk újra a küszöbölt képen, ahogyan ezt a 4.2(g) ábra illusztrálja. Ezt követően újabb küszöbölést hajtottunk végre, de jelen esetben az egyes régiókban található lyukak száma szerint. A priori ismeretként szolgált, hogy egy vonalkód legalább nyolc sávból áll, ezért csak azon objektumok kerültek megtartásra, melyek legalább ennyi lyukat tartalmaztak. A feldolgozás további lépéséhez a megmaradt objektumokban lévő 1 pixelnyi kiterjedésű lyukak koordinátáit eltároltuk. Ezt láthatjuk a 4.2(h) ábrán. Ezáltal erősen lecsökkentettük a lehetséges kód régiókat.

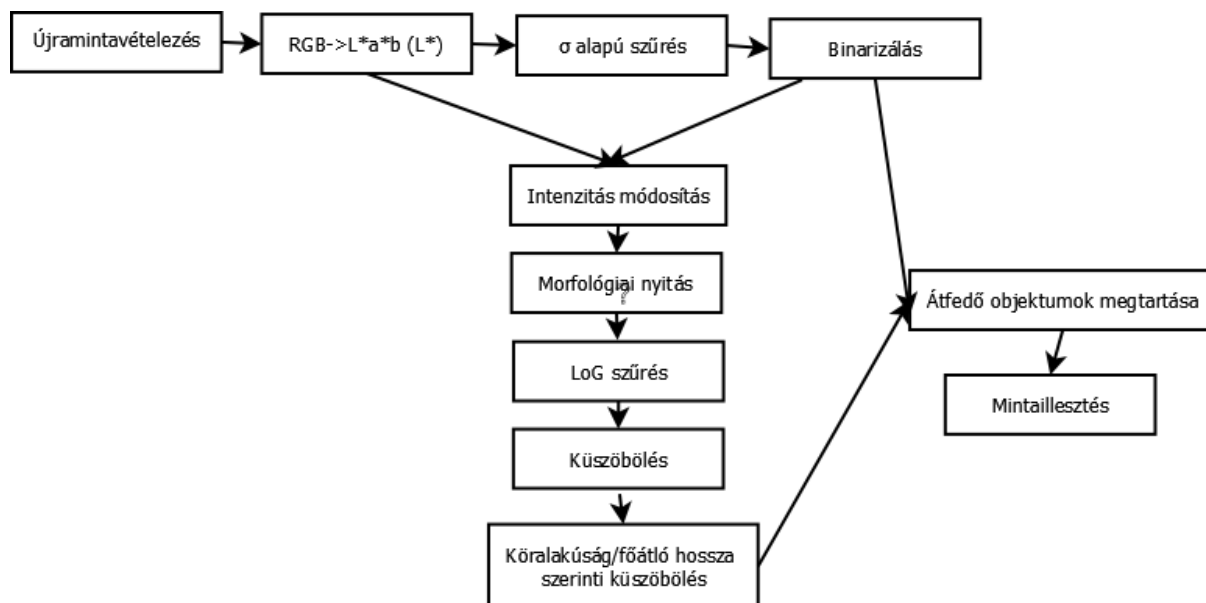
A következő lépésben az eredeti kép binarizált változatán, amit már korábban megadtunk, egy négyzet alakú, 2×2 -es méretű szerkesztőelemmel dilatációt alkalmaztunk. Erre amiatt volt szükség és azért ilyen kis szerkesztőelemmel, hogy az összetartozó régiók összetartozzanak, de fals szegmensek ne olvadjanak össze. A 4.2(i) ábra szemlélteti ezt a lépést. A meglévő lyukak koordinátáit felhasználva a dilatált képen csak azon objektumokat tartottuk meg, melyeknek az adott koordinátpontján objektumpont helyezkedett el, ahogyan a 4.2(j) ábrán is látható.

A kapott eredményképen az egyes vonalkódot alkotó sávok jól elkülöníthetőek. Ahhoz, hogy ezek egy egységként szerepeljenek, morfológiai nyitás alkalmaztunk négyzet alakú szerkesztőelemmel. A méretét annak függvényében határoztuk meg, hogy mekkora volt a sávok közötti maximális távolság. Ezáltal biztosított, hogy minden vonalkódot alkotó sáv egy régiót alkosson. Ezt szemlélteti a 4.2(k) ábra.

4.2. QR kódok detektálására javasolt módszer

A mindennapokban sűrűn használatos vonalkód típus a QR kód. Számos alkalmazási terület felsorakoztatható, úgy mint például a kiterjesztett valóság, ahol ennek a 2D-s kódnak a lokalizálása valamiféle szerepet tölthet be. Ebben az alfejezetben bemutatunk egy új, saját fejlesztésű eljárást a detektálási feladatra. Az eljárás lépéseit összefoglaló folyamatábra a 4.4 ábrán látható.

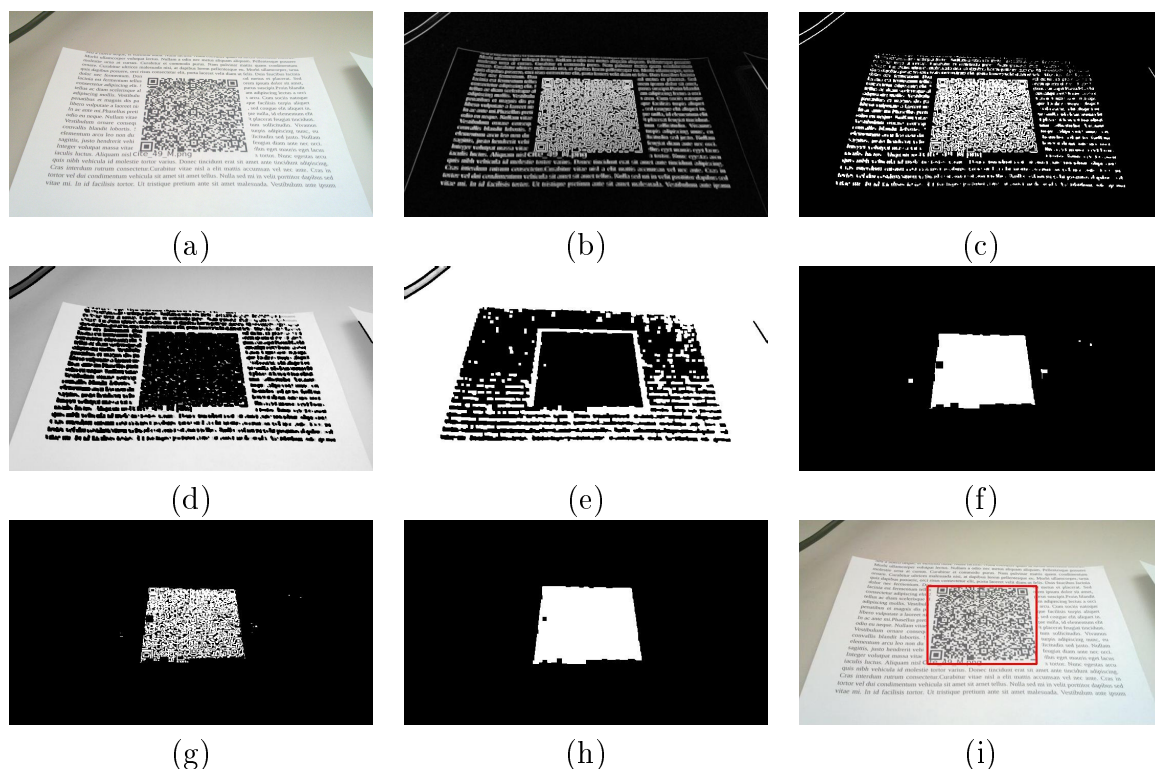
Először újraminavételeztük a bementi képet, mivel szükségtelennek bizonyult az eredeti (viszonylag nagy felbontású) képmérettel való feldolgozás, illetve a nagyobb képméret pixelenkénti manipulálása a futási időt is növeli. A képet, amennyiben annak szélessége, illetve magassága meghaladta az 1000 pixelt, úgy $400 \times W$, egyébként $600 \times W$ felbontásra redukáltuk, ahol a W úgy kerül meghatározásra, hogy az eredeti méretarányok ne változzanak.



4.4. ábra. A lokalizálás folyamatának grafikus megjelenítése

Hasonlóan a már ismertetett 1D-s kódokra fejlesztett saját módszerhez, következő lépésként a többszintű kép előállításához $L^*a^*b^*$ színtérkonverziót követően, a fényesség csatorna került használatra. Ismeretes, hogy a QR kódok rendelkeznek úgynevezett csendes zónával. Ez rendszerint magas intenzitásértékekkel telített keretet ad magának a kódnak, így amennyiben körülötte nem hasonló világosságértékekkel rendelkező pixelek vannak, úgy jól elszeparálható a háttértől. Ellenkező esetben az előtérobjektum beleolvad a háttérbe, vagy nem odatartozó szegmensek kerülnek az előtérbe, ami fals detektáláshoz vezethet. Ennek elkerülése érdekében a többszintű képen lokális szűrést végeztünk, melyben az adott pixelpozíció 3×3 -as környezetében az intenzitásértékek közötti szórást számítottuk, majd ennek értékét vette fel az adott pixel, mint intenzitásérték. Ezek az értékek $[0, 1]$ intervallumba normáltak (4.5(b)). Az így kapott képből egyszerű küszöbölési technikával bináris képet állítottunk elő (4.5(c)). A küszöbértéket az intervallum felénél határoztuk meg. A kapott bináris képet felhasználva az eredeti szürkeárnyalatos képen minden olyan képkoordináta intenzitásértékét 0-ra módosítottuk, ahol a bináris képen objektumpontot találtunk (4.5(d)), segítve ezzel a további feldolgozást.

Ezt követően, annak érdekében, hogy intenzitásértékben egy jól meghatározott környezetben a pixelértékek minimális vagy maximális szürkeségi értéket vegyenek fel, ezáltal elősegítsék az összetartozó szegmensek szegmentálását, többszintű morfológiai nyitást végeztünk. Az így kapott képen a QR kód terület jól láthatóan elkülöníthető, mint ahogyan a 4.5(e) ábrán is látható. A következő lépésben egy LoG szűrést hajtottunk végre,



4.5. ábra. A javasolt módszer köztes lépései. (a) input kép, (b) elválasztó vonal kirajzolása, (c) binarizálás, (d) intenzitásértékek módosítása, (e) morfológiai nyitás, (f) küszöbölt LoG szűrt kép, (g) utófeldolgozás, (h)-(i) eredménykép

majd annak eredményét küszöböltük, mivel a feldolgozás későbbi fázisaiban már bináris képpel dolgoztunk tovább (4.5(f)).

Az így kapott képen található még téves régiók, ezért további feldolgozásra van szükség. Lehetséges, hogy a képen olyan objektumok is egy régiót alkotnak, melyek nem szorosan összetartozóak, ezért egy kisebb szerkesztőelemmel morfológiai eróziót hajtottunk végre a képen, mely 4.5(f) ábrán látható. Ezután, hogy szűkítsük a keresési teret, megvizsgáltuk az objektumok köralakúságát és főátlójának hosszát, melyek alapján további területeket dobtunk el, aminek következtében már csak a QR kód régió maradt a képen.

A további műveletek sorozata felfogható utófeldolgozási folyamatként. Az 1D-s saját algoritmus esetében az egész művelet sor a mintaillesztés eredményére épült, míg jelen esetben a valós vonalkód régiók meghatározására, verifikálás céljából került alkalmazásra. Ehhez a legutolsó művelet eredményét, valamint az első lépések egyikében binarizált szűrt képet használtuk fel. Minden olyan objektumpontot megtartottunk a szűrt képből, mely koordinátákon a különbözőzők jellemzők vizsgálata után kapott képen is objektumpontot tartalmazott. A 4.5(g) ábra illusztrálja ennek a műveletnek az eredményét.

A 4.6 ábrán látható mintát felhasználva mintaillesztést végeztünk az előző lépésben



4.6. ábra. A mintaillesztéshez használt minta



4.7. ábra. A javasolt algoritmus néhány további eredményképe

kapott eredményképen. A minta méretét az eredeti (708×708) 0,8% -ra redukáltuk és ezt használtuk fel.

Az (4.1) formulát alkalmazva, majd az ott leírt küszöbölést felhasználva kiszűrtük a valósnak vélt vonalkód területeket. QR kódok esetében nem szimmetrikus sablont használtunk fel, de elegendő volt annak ugyanazon elforgatottjait venni, mint 1D-s esetben, mivel ezzel is lefedésre kerül az összes lehetséges orientáció a minta megjelenését illetően. A 4.5(h) ábra és 4.5(i) a kapott eredményképeket mutatja. Az algoritmus működésének további eredményképeit szemlélteti a 4.7 ábra.

5. fejezet

A tárgyalt algoritmusok összehasonlító elemzése

Ezen fejezet keretei közt a korábban ismertetett algoritmusok hatékonyságát vetjük össze adott jellemzők mellett. A dolgozat ezen részében átfogó képet adunk az egyes algoritmusok különböző torzításokkal terhelt képeken való viselkedéséről, illetve a tanulmányozzuk, hogy a hatékonyságukat egyes jellemzők hogyan módosítják. A későbbiekben az egyes algoritmusokra való hivatkozáshoz betűszavakat vezettünk be, melyek a következők:

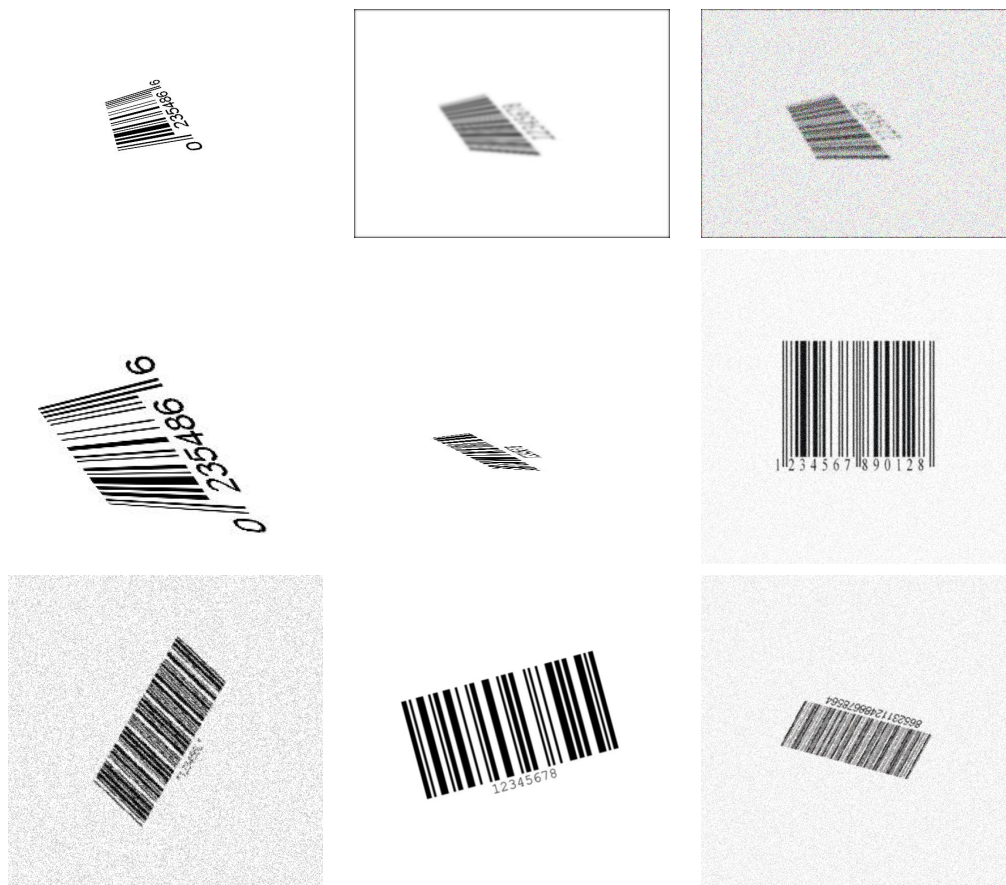
- LINB (Daw-Tung Lin és Chin-Lin Lin alapvető képműveleteket használó eljárása)
- LINM (Daw-Tung Lin és Chin-Lin Lin futáshosszok számításán alapuló eljárása)
- OHBUCHI (Eisaku Ohbuchi, Hiroshi Hanaizumi és Lim Ah Hock lokális küszöbölésen alapuló eljárása)
- Saját1D
- SajátQR

5.1. Tesztsorozat, tesztkörnyezet, implementáció

Teszthalmaz 1D-s kódok lokalizálására

Digitálisan tesztképeket generáltunk az fejezetben bemutatott 1D-s vonalkód típusokat alapul véve. Mindehhez elegendő volt az egyes típusú kódokból egyet kiválasztani. A tesztképek egy vonalkódot tartalmaznak ezekből, illetve különféle torzításokkal terheltek.

Az általunk generált képek a következő tulajdonságokkal rendelkeznek: a képeket 15°-onként elforgattunk 0°-tól 180°-ig, Gauss szűrést alkalmaztunk 3×3 -as maszkkal és 6 különböző σ -val, illetve additív zajt is hozzáadtunk a képekhez 10% -os léptékkal 0%



5.1. ábra. Generált minta 1D tesztképek különböző torzítások mellett

és 50% intervallumban. Összegezve tehát 8 különféle típusú vonalkódból 12-féle orientációban, 6 különböző Gauss szűrővel és 6 eltérő mértékű additív zajjal terhelve és perspektív torzulással ellátva, összesen közel 15000 képet generáltunk. A 5.1 ábrán látható néhány különböző torzítással terhelt, generált tesztkép.

További 4220 vonalkódot tartalmazó valós életbeli képet is felhasználtunk a The WWU Muenster adatbázisból. Ezen képeken semmiféle manipulációt nem hajtottunk végre, de torzításokkal terheltek, nem ideálisak. Az adatbázisból származó képekre a 5.2 ábrán láthatunk példákat.

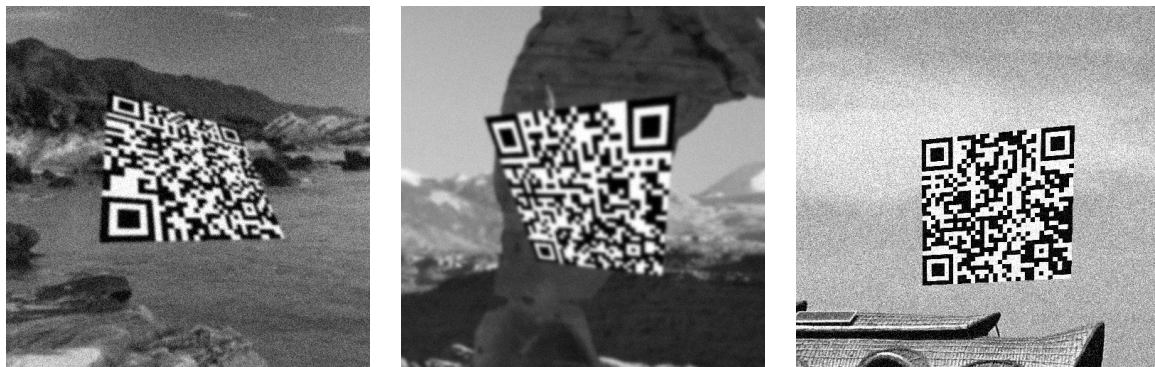
Teszthalmaz QR kódok lokalizálására

A QR kód detektáló algoritmusok kiértékelésére meglévő képi adatbázisok kerültek felhasználásra. Közel 1400 valós tesztképből álló adatbázis [18], valamint 10000 szintetikus tesztkép állt a rendelkezésünkre. A 5.3 ábrán szintetikus, a 5.4 ábrán valós képek láthatóak a teszhalmazokból.

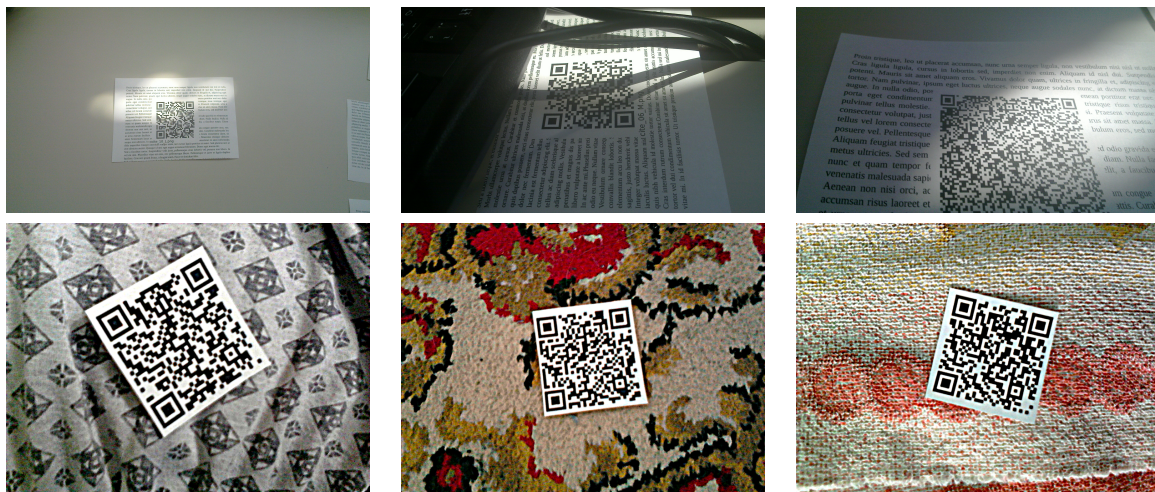
Az eljárásokat MATLAB-ban implementáltuk, Image Processing Toolbox segítségével.



5.2. ábra. Valós 1D-s tesztképek



5.3. ábra. Szintetikus QR tesztképek



5.4. ábra. Valós QR tesztképek

A kiértékelést egy Intel(R) Core(TM) i7-4700MQ 2.40Ghz CPU-val rendelkező számítógéppel hajtottuk végre.

5.2. Eredmények vizsgálata

Jelen alfejezetben azt tárgyaljuk, hogy az egyes algoritmusok egymáshoz viszonyítva milyen hatékonyak voltak különböző jellemzőkkel ellátott képeken, továbbá milyen futási időt produkáltak. A tesztelés során minden egyes tesztképet illetően általunk előre definiált jellemzők kerültek kinyerésre, melyek a jelen összehasonlítás alapjául szolgálnak. Az algoritmusok hatékonyságának kiszámításához a Jaccard-féle hasonlósági együtthatót vettük alapul, ahol a tényleges és a detektált kód régiók befoglaló téglalapjának átfedését vizsgáltuk. Az együttható az alábbiaképpen írható fel:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}, \quad (5.1)$$

ahol A a valódi kód, míg B a detektált kód befoglaló téglalapját jelöli. Ennek alapján azon egyedek fognak true pozitív példának megfelelni, tehát valós kódterületnek bizonyulni, melyeknél a detektált objektum legalább 90%-ban fedésben áll az előre meghatározott területtel. Minden olyan objektum, mely detektált, de ennél kisebb, vagy egyáltalán nincs fedésben semmilyen vonalkód területtel, azokat fals pozitív példának fogjuk tekinteni. Fals negatív példák lesznek azok, melyek ugyan vonalkód területek, de nem kerültek detektálásra.

Jelölje TP a true pozitív, FP a fals pozitív, TN a true negatív, illetőleg FN a fals negatív értéket. Pontosság, lefedettség, illetve hatékonyság mérőszámokkal fogjuk jellemezni az egyes eljárásokat. Ezek a következőképpen kerülnek kiszámításra [19] :

$$\text{Pontosság}(P) = \frac{TP}{TP + FP} \quad (5.2)$$

$$\text{Lefedettség}(L) = \frac{TP}{TP + FN} \quad (5.3)$$

$$\text{Hatékonyság}(H) = \frac{TP + TN}{TP + TN + FP + FN}. \quad (5.4)$$

Pontosság alatt tehát azt értjük, hogy a valós kódszegmensek és a fals detektált területek milyen arányban vannak egymáshoz képest. Lefedettségi mérőszám a detektált területek mekkora arányban valós területek. Hatékonyság alatt pedig az e két mérőszám kombinációjából adódó detektálási jóságot kaphatjuk meg. Ezek az értékek százalékos formában adóttak.

Az eredmények alapján megállapítható, hogy az összes vizsgált algoritmus hatékonyan képes vonalkód detektálásra.

Korábban említettük, hogy a generált tesztképek különböző mértékű hozzáadott zajt, simítást, elforgatást, nyújtást, illetve perspektív torzulást tartalmaznak. Az elforgatás,

nyújtás, perspektív torzulás nem érinti az egész képet, csak a vonalkódot. A tapasztalat azt mutatta, hogy az eljárások az elforgatásra, nyújtásra, perspektív torzulásra invariánsak, ezért a dolgozat további részében csak a zaj és simítás jellemzőket vizsgáljuk az egyes eljárások esetében.

Az egyes vonalkód típusokat változatos szerkezeti felépítés jellemzi, ennek következtében a lokalizálásuk sem ugyanolyan hatékonyságú. Az is megfigyelhető, hogy a képek különféle torzításokkal terhelt eseteiben az algoritmusok veszítenek a hatékonyságukból. A dolgozatban bemutatásra kerültek 1D-s vonalkódokat, valamint QR kódokat lokalizáló eljárások is. Az elért eredményeket külön alfejezetben vizsgáljuk.

5.2.1. 1D-s vonalkód detektálók eredményei

Az egyes eljárások hatékonyságát különféle szempontok alapján vizsgáltuk. A továbbiakban az elért eredményeket tárgyaljuk részletesen.

A 5.1 táblázat a szintetikus tesztadatbázisban megtalálható 8 különböző vonalkód típus szerinti mérőszámokat mutatja.

5.1. táblázat. Az algoritmusok 1D-s vonalkód típusonkénti hatékonysága (H), lefedettsége (L), valamint pontossága (P) (százalékban).

	LINB			LINM			Saját1D		
	H	P	L	H	P	L	H	P	L
Codabar	98.8	99.8	99.1	98.6	100.0	98.6	93.7	99.7	96.2
Code128	88.7	89.0	99.6	100.0	100.0	100.0	98.0	99.6	99.4
Code39	75.7	75.7	100.0	80.7	100.0	80.7	94.5	99.0	97.0
EAN-8	96.6	96.6	100.0	100.0	100.0	100.0	98.0	99.7	99.7
EAN-13	89.8	90.1	99.7	100.0	100.0	100.0	99.2	99.2	100.0
I2of5	98.4	98.5	100.0	75.4	100.0	75.4	99.6	99.7	99.9
UPC-A	88.9	89.0	100.0	96.2	100.0	96.2	97.2	98.0	99.0
UPC-E	98.2	98.3	100.0	100.0	100.0	100.0	99.4	100.0	99.4
Összesen	88.5	97.3	99.8	79.0	100.0	79.0	97.4	99.6	98.9

Előfordulnak olyan kódtípusok, melyeknél szembetűnően alacsonyabb a hatékonyság minden eljárás esetében. Ennek az oka az adott típust jellemző tulajdonságok lehetnek. Az I2of5 esetében például ismeretes, hogy a kód nem rögzített szélességű, tehát az x szerinti kiterjedése bármekkora lehet, ami több kisebb régióra való szakadását eredményezheti szegmentáláskor. Az is látható, hogy a fix szélességű kódok detektálása szinte az összes esetben 100% közeli.

A következőkben azt vizsgáljuk, hogy az egyes torzításokat egyenként tekintve milyen mértékű a hatékonyságvesztés, amennyiben ez releváns. Elsőként a különböző mértékű

hozzáadott zaj függvényében tekintjük a számértékeket. A 5.2 táblázatból jól látszik, hogy az alapvető képműveleteket alkalmazó LINB algoritmus a másik két módszerhez képest rendkívül zajérzékeny.

5.2. táblázat. Az algoritmusok hozzáadott zaj szerint hatékonysága (H), lefedettsége (L), valamint pontossága (P) (százalékban) 1D-s tesztképeken.

	LINB			LINM			Saját1D		
	H	P	L	H	P	L	H	P	L
0	88.6	88.8	100.0	93.8	100.0	93.8	98.2	99.0	99.2
0.5	85.2	85.4	100.0	93.8	100.0	93.8	98.3	99.0	99.2
1	84.2	84.5	100.0	93.7	100.0	93.7	98.3	99.0	99.2
1.5	83.6	83.8	100.0	93.7	100.0	93.7	98.3	99.0	99.2
2	82.2	82.4	100.0	93.7	100.0	93.7	98.3	99.0	99.2
2.5	81.9	82.4	99.6	93.2	99.4	93.7	98.2	99.0	99.2
Összesen	86.0	86.7	99.3	93.6	99.9	93.7	98.7	98.8	99.9

A zajszint növekedésével folyamatosan csökken annak hatékonysága. Habár a lefedettsége szinte minden esetben 100%-os, ami azt jelenti, hogy fals negatív vonalkód szegmensek nem kerültek detektálásra, de mindemellett a pontosság alacsonyabb volt. Ebből következően a hatékonyságából is veszített. A LINM és a Saját eljárás mondhatni konstans eredménnyel képes detektálni a különböző zajszinteken. A hozzáadott zaj utáni elemzést követően tanulmányozzuk a simítás mértékének függvényében az algoritmusokat. Érdekes megfigyelni a 5.3 táblázatban, hogy az előbb a zajérzékenysége miatt kiemelt LINB eljárás a σ érték növekedésével nagyobb hatékonyságra képes.

Hasonló megállapítás tehető a másik két algoritmusra a simítás tekintetében is, mint a különböző zajszintek melletti viselkedésükben, miszerint nagyon csekély mértékben azért változik a hatékonysági rátájuk.

Ahogy már említettük, a tesztelés során valós képek is kiértékelésre kerültek. Az elért eredményeket tartalmazza a 5.4 táblázat. A felhasznált adatbázis interpolált tesztképek sorából állt. Az eredeti képméret 2592×1944 volt. Megállapítható, hogy az egyes algoritmusok hatékonyan képesek valós esetben is a vonalkódok detektálására. Érdekes megfigyelni, hogy a szakirodalmi eljárások 1024×768 -as képméret esetében számottevően kisebb hatékonysággal működtek. Valós esetben az is látható ezeknél a módszereknél, hogy nagy mértékben veszítettek a pontosságukból, tehát több fals detektálás következett be.

Az eredmények tanulmányozásának fontos része a futási idő vizsgálata is. A 5.5 táblázatban összesítve láthatók az elért időigények szintetikus képekre, valamint valós

képekre való bontásban. Elmondható, hogy az egyes eljárások gyorsan képesek detektálni a vonalkód szegmenseket, valamint a szórás sem számottevő.

5.3. táblázat. Az algoritmusok hozzáadott simítás szerint hatékonysága (H), lefedettsége (L), valamint pontossága (P) (százalékban) 1D-s vonalkódot tartalmazó képek esetén.

	LINB			LINM			Saját1D		
	H	P	L	H	P	L	H	P	L
0	83.1	83.4	100.0	93.8	100.0	93.8	98.8	99.0	99.8
10	83.5	83.8	100.0	93.7	100.0	93.7	99.0	99.0	99.8
20	83.9	84.2	100.0	93.7	100.0	93.7	98.9	99.0	99.8
30	83.3	83.6	100.0	93.7	100.0	93.7	98.9	99.0	99.8
40	85.0	85.3	100.0	93.0	100.0	93.0	98.5	98.7	99.8
50	86.6	86.9	100.0	93.3	99.5	93.0	98.7	98.8	99.8
Összesen	84.5	84.5	100.0	93.7	99.9	93.8	98.6	98.8	99.8

5.4. táblázat. Az algoritmusok hatékonysága (H), lefedettsége (L), valamint pontossága (P) (százalékban) 1D-s kódokat tartalmazó valós tesztképeken.

	LINB			LINM			Saját1D		
	H	P	L	H	P	L	H	P	L
800 × 600	96.7	69.2	96.7	91.8	68.0	91.8	97.4	99.0	97.4
1024 × 768	86.4	69.3	86.4	81.2	67.9	81.2	98.4	98.9	98.4
1600 × 1200	96.0	68.6	96.0	90.0	67.2	90.0	97.7	98.0	97.7
2592 × 1944	95.9	68.5	95.9	89.7	67.1	89.7	98.8	99.1	98.8
Összesen	93.8	68.9	93.8	88.1	67.6	88.2	97.0	98.1	97.0

5.5. táblázat. Az algoritmusok futási ideje (átlag (sec) ± szórás)

	LINB	LINM	Saját1D
szintetikus	0.724 ± 0.214	0.974 ± 0.148	0.625 ± 0.150
valós	0.981 ± 0.281	0.795 ± 0.726	0.683 ± 0.283

5.2.2. QR kód detektálók eredményei

A QR kódokat lokalizáló eljárások aspektusából is megvizsgáltuk az eredményeket. Ahogy a tesztadatbázis ismertetésénél említettük, meglévő tesztalmanach használtunk fel a

kiértékelés során. Hasonlóan, mint az 1D-s tesztalmaz kiértékelésének esetében, itt is külön tanulmányoztuk a szintetikus és valós tesztképeken való futtatással kapott mérőszámokat.

Az 1D-s esethez képest mind a szakirodalmi eljárások, mind a saját javasolt algoritmus kisebb hatékonysággal képes detektálni a kívánt területet, ugyanakkor még ez az eredmény is megfelelő. Látható a 5.6 táblázatban, hogy közel ugyanakkora hatékonysággal képesek a valós tesztképeket detektálni, mint a szintetikusakat. Jelen esetben is, ha a fő szempontot figyelembe vesszük, ami a hatékonyság, akkor a saját algoritmus teljesít a legjobban.

5.6. táblázat. Az algoritmusok hatékonysága (H), lefedettsége (L), valamint pontossága (P) (százalékban) valós és szintetikus QR kódot tartalmazó tesztképeken

	OHBUCHI			LINM			SajátQR		
	H	P	L	H	P	L	H	P	L
szintetikus	87.3	100.0	83.7	88.9	93.4	94.9	89.5	99.9	89.5
valós	83.6	95.0	87.5	84.5	94.0	89.3	89.0	95.7	92.6

A futási idők jelen esetben sem elhanyagolható tényezők az algoritmusok jóságának megállapításában. A 5.7 táblázatban összefoglalásra kerül szintetikus, valamint valós esetben, hogy az egyes eljárások mekkora időigénnyel rendelkeznek. OHBUCHI algoritmus láthatóan nagyon gyors, ugyanakkor, ha megvizsgáljuk a 5.6 akkor elmondható, hogy a gyorsaság ára a hatékonyság csökkenése. Valamint valós tesztképek esetében a detektálási idők közti szórás nagy.

Az elért eredményekből következik, hogy hatékony vonalkód detektáló algoritmusok kerültek összehasonlításra, melyek különböző alapelven működnek és a javasolt módszerek mutatták a legnagyobb hatékonyságot a vizsgált szempontok alapján.

5.7. táblázat. Az egyes eljárások futási ideje (átlag (sec) $\pm$ szórás)

	OHBUCHI	LINM	SajátQR
szintetikus	0.011 $\pm$ 0.007	1.004 $\pm$ 0.073	0.267 $\pm$ 0.023
valós	0.237 $\pm$ 2.225	1.225 $\pm$ 0.809	0.419 $\pm$ 1.346

Összefoglalás

A dolgozatban ismertettük a vonalkód detektálás három szakirodalomban található módszerét, valamint két új, saját fejlesztésű algoritmust. A szakirodalmi algoritmusok egyike nem specializálódott 1D-s, valamint 2D-s kódtípusokra, hanem hatékonyan detektálja mindkettőt, viszont a további bemutatásra került módszerek igen. Egy-egy szakirodalmi módszert mutattunk be, melyek egyike 1D-s, míg a másik QR kódok lokalizálására fejlesztették ki, illetve még egy eljárást, mely mindkét esetben működőképes. Ehhez hasonlóan a saját eljárások egyike klasszikus 1D kódtípus felismerésére alkalmas, míg a másik a 2D-s kódok közé tartozó QR kódokra. Az eljárások pontosságának és futási idejének vizsgálatára létrehoztunk közel 15000, valamint 10000 tesztképet magában foglaló adatbázist.

Az algoritmusok pontosságát vizsgáltuk különféle szempontokat tekintve. A digitálisan generált 1D-s képekhez különböző mértékű simítást, zajt rendeltünk. A tesztképeken található kódok elforgatásra, nyújtásra is kerültek, illetve perspektív torzulást is szenvedtek, de a későbbiekben a tesztelés során világossá vált, hogy ezen algoritmusok egyike sem függ ezen tényezőktől, így ez az összehasonlítás során nem kerültek elemzésre. A QR kódot tartalmazó tesztképek némelyike perspektíven torzult, zajjal terhelt, valamint a nem megfelelő képalkotásból származóan elmosódott.

A tesztelés során megmutatkozott az is, hogy van olyan irodalmi algoritmus, mely nagyon zajérzékeny, a saját algoritmusaink csekély mértékben veszítenek hatékonyságukból, mint a többi tárgyalt eljárás. Tanulmányozva a futási időket és az algoritmusok pontosságát mindkét kódtípus esetében, beláttuk, hogy a saját algoritmusunk hatékonysága nagyobb mind ideális képek esetében, mind a képek különböző torzításokkal terhelt esetében, mint a többi ismertett irodalmi eljárásé. Ezen túlmenően az futási idők eredményeit tekintve, valós idejű implementáció is kivitelezhető.

Irodalomjegyzék

- [1] R. C. Palmer, *The Bar Code Book: Reading, Printing, and Specification of Bar Code Symbols*. Helmers Pub, 1995.
- [2] R. Adelmann, „Toolkit for Bar Code Recognition and Resolving on Camera,” in *Phones - Jump Starting the Internet of Things. In: Informatik 2006 workshop on Mobile and Embedded Interactive Systems*, 2006.
- [3] J.-H. Li, Y. Wang, Y. Chen, M. Zhang, G.-C. Wang, and P. Li, „Morphological Segmentation of 2-D barcode Gray Scale Image,” *JCP*, vol. 8, pp. 2461–2468, 2013.
- [4] P. Bodnár and L. G. Nyúl, „Improving Barcode Detection with Combination of Simple Detectors,” in *Signal Image Technology and Internet Based Systems (SITIS), 2012 Eighth International Conference on*, 2012, pp. 300–306.
- [5] —, „Efficient Barcode Detection with Texture Analysis,” in *Signal Processing, Pattern Recognition, and Applications, Proceedings of the Ninth IASTED International Conference on*, 2012, pp. 51–57.
- [6] M. Katona and L. G. Nyúl, „Efficient 1D and 2D Barcode Detection Using Mathematical Morphology,” in *Mathematical Morphology and Its Applications to Signal and Image Processing*, ser. Lecture Notes in Computer Science, 2013, vol. 7883, pp. 464–475.
- [7] —, „A Novel Method for Accurate and Efficient Barcode Detection with Morphological Operations,” in *Signal Image Technology and Internet Based Systems (SITIS), 2012 Eighth International Conference on*, 2012, pp. 307–314.
- [8] S. Kong, „QR Code Image Correction based on Corner Detection and Convex Hull Algorithm,” *Journal of Multimedia*, vol. 8, pp. 662–668, 2013.
- [9] S. Wachenfeld, S. Terlunen, and X. Jiang, „Robust recognition of 1-d barcodes using camera phones,” in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, 2008, pp. 1–4.

- [10] X. Q. James Juett, „Barcode Localization Using Bottom-hat Filter,” *NSF Research Experience for Undergraduates*, 2005.
- [11] T. R. Tuinstra, „Reading Barcodes from Digital Imagery,” 2006.
- [12] E. Tekin and J. M. Coughlan, „An Algorithm Enabling Blind Users to Find and Read Barcodes,” in *WACV*. IEEE Computer Society, 2009, pp. 1–8.
- [13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Prentice Hall, 2007.
- [14] R. Brunelli, *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley Publishing, 2009.
- [15] D.-T. Lin and C.-L. Lin, „Multi-symbology and Multiple 1D/2D Barcodes Extraction Framework,” in *Proceedings of the 17th International Conference on Advances in Multimedia Modeling - Volume Part II*, 2011, pp. 401–410.
- [16] —, „Automatic Location for Multi-symbology and Multiple 1D and 2D Barcodes,” *Journal of Marine Science and Technology*, vol. 21, pp. 663–668, 2013.
- [17] E. Ohbuchi, H. Hanaizumi, and L. A. Hock, „Barcode Readers Using the Camera Device in Mobile Phones,” in *Proceedings of the 2004 International Conference on Cyberworlds*, ser. CW '04, 2004, pp. 260–265.
- [18] G. Sörös and C. Flörkemeier, „Blur-resistant Joint 1D and 2D Barcode Localization for Smartphones,” in *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia*, 2013, pp. 1–8.
- [19] T. M. Mitchell, *Machine Learning*. McGraw-Hill, Inc., 1997.

Köszönetnyilvánítás

A kutatás a TÁMOP 4.2.4.A/2-11-1-2012-0001 azonosító számú Nemzeti Kiválóság Program – Hazai hallgatói, illetve kutatói személyi támogatást biztosító rendszer kidolgozása és működtetése országos program című kiemelt projekt keretében zajlott. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.