Debreceni Egyetem

Informatikai Kar

Szabadon választott téma a digitális képfeldolgozás témakörében
Képjavítási eljárások

Témavezető:
Dr. Habil Fazekas Attila
Egyetemi Docens

Készítette:
Tóth Imre
Programtervező - Matematikus

Debrecen
2008.

Tartalomjegyzék

1. fejezet

Bevezetés

Témaválasztásom apropóját elsősorban az orvosi „műhibák”, és az autómatizálás-, valamint a számítástechnikai algoritmusokban rejlő határok megismerésének vágya szolgáltatták (főként a képfeldolgozás területére koncentrálva). Ugyanis a családomon, ismerősi köreimban többször is előfurdult, hogy egyes betegségeket az orvosok - a legmodernebb technikák ellenére is - félrediagnosztizáltak, olyan elváltozásokat jeleztek, melyek esetleg nem is léteztek az adott személynél.

A legutóbbi eset a nagymamámnál történt (és tulajdonképpen e miatt szántam rá magam komolyabban, hogy a hibák felderítésével foglalkozzak), amikor súlyos betegségre utaló jeleket vettek észre a vérében, többszöri ellenőrzés után is. A PET-es, és endoszkópos vizsgálatok viszont ennek teljesen elltmondó eredményt szolgáltattak. Mint kiderült, nem csak a mamám eredményénél voltak félrediagnosztizálások abban az időszakban, hanem sokmindenkinél. Kiderült ugyanis, hogy a mintákat egy autómatizált számítógép elemezte, és nem egy esetben állított fel rossz diagnózist.

Ez volt az a pillanat, amikor eldöntöttem, amennyire tehetem, utánajárok annak, vajon miért tévedhet egy olyan autómatizált rendszer, amin esetleg emberéletek foroghatnak kockán.

Az orvostudományban, de az emberiség más területein is, egyre jobban kezdenek elterjedni a számítógépek. A digitális képfeldolgozás, és számítástechnika ma még elég fiatal tudomány, nagyon érzékeny a hibaforrásokra, és még nem egy tökéletesen kiforrott technológia (akárcsak a számítástechnika egyéb területei). Mi sem bizonyítja ezt jobban, minthogy nem létezik univerzális képfeldolgozó algoritmus, a megalkotott képeink nem zajmentesek, a digitális csatornák pedig nem teljesen megbízhatóak.

A képfeldolgozás gyakorlati problémáira Hajdu András tanár Úr gyakorlatain lettem figyelmes, és nagy érdeklődést váltott ki belőlem, miért is nem lehet tökéletes egy olyan technológia, melyet precíz számítóeszközökkel végeznek.

Célom ezzel a dolgozattal az volt, hogy saját magam -, valamint az olvasóval is megismertessem, milyen hibaforrások fordulhatnak elő a digitális képfeldolgozásban (elsősorban a zajokra, és torzulásokra koncentrálva), és ezeket a hibákat miként lehet eliminálni, hogy ne gyűrűzzenek tovább a feldolgozás további lépéseiben - hiszen mint tudjuk, a rossz számítások (hibák) a program végére exponenciális nagyságot ölthetnek.

Ezzel a témakörrel a képjavítási eljárások szoktak foglalkozni. Az egyes témakörökhöz programokat készítettem demonstálási / kipróbálási (teszt) célból *MatLab*, valamint *Delphi* rendszerben, így a gyakorlatban is meg lehet nézni, milyen hatékonysággal dolgoznak a manapság használatos képjavítási eljárások. Célom egy olyan dokumentum elkészítése, mely a jelenleg legelterjedtebb képjavító eljárásokat gyűjti össze, és mutatja be (hibáival együtt). A fejezetek összefoglaló jellegűek, ezért mindegyikhez külön irodalomjegyzéket készítettem.

2. fejezet

Történelmi áttekintés

A számítógéppel végzett képfeldolgozás, analízis, illetve gépi látás egy nagyon érdekes, és speciális részét teszi ki a számtógépes adatfeldolgozásnak. Ezen szakterület alapköveit az 1960-as években tették le az *MTI, Bell laboratórium, Marylandi egyetem, ...* kutatóintézeteiben. [20] Ebben az időszakban nagyon sokba került minden ezen a szakterületen való kutatás, hiszen nem állt rendelkezésre megfelelő mennyiségű, és minőségű erőforrás. Az 1970-es években viszont megjelentek az olcsóbb, és dedikált hardverek, így a számítógépes képfeldolgozás egyre inkább kezdett elterjedni. Ezek után egyes problémákat már valós időben sikerült kezelni.

Hazánkban a '80-as évek elején jutottak túl a számítógépes képfeldolgozásban a kísérleti kutatások szintjén. Ennek hatására kezdtek elterjedni a különbőző, általános célú kisgépekre, illetve micro-, és személyi számítógépekre írt digitális képfeldolgó rendszerek. Ezen kívül rengeteg dedikált eszközt jelent meg egy-egy konkrét feladat elvégzésére: pl. PET, űrfelvételek különböző célú elemzése, tomográf elemzés, hőkamerás felvételek elemzése, stb.

Régen a képfeldolgozás magas „költségvetése", és a feldolgozás lassúsága miatt csak kevesen foglalkozhattak ezzel a szakterülettel. A magas árnak két oka is volt:
- Az eletronikai eszközöknek nehézkes, és drága beszerzése (a képfeldolgozási problémák megoldásához speciális hardverelemek kellenek).
- Régebben a programok egy rendszerre voltak dedikálva, ezért nem rendelkeztek a hordozhatóság tulajdonságával. Amik az egyik rendszeren működtek, nem biztos, hogy a másik rendszerre is átvihetők lettek volna módosítás nélkül. Ennek oka, hogy a programok assembly nyelven voltak megírva, ami elsősorban a gyorsaság miatt kellett, az akkori lassú, és drága számító-, és tárolókapacitás végett. Viszont így a hordozhatóságról le kellett mondani az esetleges speciális megoldások miatt [3].

Az utóbbi időben végbemenő - és manapság is tartó - technikai és ideológiai fej-lődés hatására újabb, és újabb rendszerek jelennek meg. Manapság minden „modern kori" programozási nyelvben vannak képfeldolgozást elősegítő modulok. [7]

Valamint mára már léteznek kizárólag képfeldolgozással foglalkozó programozási rendszerek (VisiQuest, Simulink ...), melyen nagyban megkönnyítik az egyes alkalmazások gyors, könnyű implementálását.

A digitális képfeldolgozás célja, hogy valahogy értelmezzük -, elemezzük a képet; információt nyerjünk ki belőle. Ennek az az oka, hogy már régen is észlelték, hogy bizonyos problémák analóg, manuális elemzése (pl. atomfizikában, radarfelvételek esetén, ...) nem lehetséges, részben a pontosság megkövetelése, részben pedig az adatok sokassága miatt sem. Legelső lépésként a képet rögzíteni, és digitalizálni kell. A mintavételező, és kvantáló rendszer jelentősen bedolyásolhatja a feldolgozásra kerülő kép minőségét. Amennyiben a képet feldolgozás céljából továbbítani kell (pl. műholdak által készített képek esetében a földi központ felé), a továbbítás során a képen torzulások, és zajok keletkezhetnek. [2]
Feldolgozás előtt ezeket a torzulásokat meg kell szüntetni, különben a feldolgozás végére hibás eredményeket kapunk. A cél az, hogy olyan matematikai modellt alkossunk a torzító hatásokról, mely segítségével könnyen meghatározhatjuk az inverz transzformációját. Ezek után olyan transzformációkat kell végrehajtani a digitalizált képen, mely hatására az eddiginél kedvezőbb tulajdonságú képet kapjunk, miközben a fontos információknak nem változtatjuk meg.

Teljes körű, univerzális modell mgalkotása reménytelen, csak egyes részproblémákra lehet megoldást kidolgozni. Egy új algoritmus kialakítása problémafüggő. Egyes módszerek nagyon hatékonyan működnek bizonyos alkalmazásoknál, más feladatoknél ezek a módszerek elképzelhető, hogy teljesen használhatatlanok, pontosan amiatt, mert az emberi értékelő számára kell kedvező látványt létrehozni.

A képjavítások célja sokszor nem egy szebb, hanem egy könnyebben feldolgozható kép előállítása, melyen újabb műveleteket lehet végrehajtani. A képfeldolgozási eljárásokat két fő csoportba lehet rangsorolni: lokálisak, vagy globálisak, annak függvényében, hogy a képek a képpontok egy csoportját, vagy az összeset egyidejűleg feldolgozzuk.
Egy másik csoportosítási mód szerint újabb két csoportba sorolhatjuk a képjavítási eljárásokat aszerint, hogy képtérben (általában világosságkódokkal), vagy frekvenciatérben (a kép Fourier transzformáltjával) dolgozunk. (A frekvenciatérben végrehajtott javítások általában a konvolúciós elméleten alapszanak.)
Egy harmadik csoportosítási mód a hibatípusokat veszi figyelembe. E szerint három kategóriába sorolhatjuk a kijavítandó hibákat: zajok a képhez keveredése (különböző simító eljárások), kontraszthibák (különböző hisztogram alapú eljárások), és élek elmosódása (él-, sávkiemelés). Meg kell jegyezni, hogy ugyanaz a módszer más paraméterekkel más - más célra alkalmazható. Az eljárások zöme ad-hoc jellegű, heurisztikus megfontoláson alapszik. [3] Lássuk tehát azokat a módszereket, melyek megkönnyítik a képek későbbi feldolgozását oly módon, hogy eliminálják a „haszontalan" információkat/zajokat. Először az képtérben végezhető műveleteket tekintjük át, majd megnézzük, frekvenciatérben milyen eljárásokat lehet használni képjavításra.

3. fejezet

A zaj

Előfordulhat, hogy egy kép torzulni fog elkészítés, továbbítás, vagy feldolgozás alatt. A kép minőségének meghatározásával megállapíthatjuk a torzulás fokát. Az, hogy milyen minőségű kép szükséges egy faladathoz, azt az alkalmazás/feladatkör határozza meg [12].

A kép minőségének meghatározása két módon lehetséges: szubjektív, vagy objektív módon. A szubjektív módszert leginkább a televíziós technologiában alkalmazzák, ahol az alapvető kritérium egy kiválasztott professzionális, és laikus csoport néző számára az észlelés. Ők egy képről egy sor kritérium, és megfelelő jelek hatására értesülnek. A szubjektív módszerekről való részletesebb leírás megtalálható a Pratt által elkészített irodalomban [17].

Számunkra a képminőség meghatározásában érdekesebbek az objektív kvantitatív módszerek. Ettől a módszertől olyan hatást várnak el, mint a szubjektív módszertől, valamint egy egyszerű becslési eljárást, ami után a kvantitatív képminőség jelzőt fel lehessen használni a paraméter optimalizáció során. Az $f(x, y)$ kép minőségét általában úgy becsülik meg, hogy összehasonlítják egy $g(x, y)$ referencia képpel. Egyes össehasonlító metódusok egyszerűen a kvadratikus középérték különbséget használják mértékegységként: $\int\int(g-f)^2 dxdy$. Ezzel az a probléma, hogy ennél a módszernél nem tudjuk megkülönböztetni a pár nagyobb különbséget a sok aprótól. Ezért a kvadratikus középérték különbség helyett az abszolút középérték különbséget, vagy egyszerűen a maximális abszolút különbséget érdemes használni. Egy másik megoldás lehet az f, és g közötti korreláció vizsgálata.

Egy másik mérési módszer lehet a képen a kis, vagy közeli objektumok felbontásának mérése. Ennél a módszernél párhuzamos fekete-fehér csíkokat használnak a mérésre; az egy miliméterre eső fekete és fehér párok száma határozza meg a felbontást.

Igazából a képeket néha véletlenszerű hiba/hibák torzítják, amit zajnak nevezünk. A zaj keletkezhet a kép rögzítésekor, átvitelkor, vagy a kép feldolgozása esetén; függhet a kép tartalmától, de előfordulhat, hogy teljesen független tőle. A MatLab rendszer beépített zajszimulálóval rendelkezik, azaz segítségével könnyedén meg tudjuk vizsgálni a különféle zajok hatását, és függvényeket írhatunk/alkalmazhatunk eltávolításukra.

Zajt egy képre az *imnoise* beépített függvény segítségével megfelelő felparaméterezésével tehetünk.

A zajt általában az előfordulási valószínűségével szoktunk jellemezni. A zaj egyik típusa a *fehér* zaj, melynek állandó S spektrumereje van, ami azt jelenti, hogy az intenzitása nem csökken a frekvencia növekedésével. A fehér zajt gyakran a kép torzulásának legrosszabb megközelítésére használják, az egyszerű kiszámíthatósága miatt gyakran használják.

A *Gaussi* zaj egy másik nagyon elterjedt zaj megközelítő. Egy Gaussi (normál) eloszlású véletlen változó valószínűség sűrűségét a Gaussi görbe határozza meg. Egy dimenzióban a sűrűségfüggvény a következőképpen néz ki:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}},$$

ahol μ a véletlen változó középértéke, σ pedig az elhajlása. Az esetek többségében a Gaussi zajjal való megközelítés nagyon jó a zaj becslésére. MatLab-ban is van lehetőség Gaussi zaj előállítására, ehhez mindössze az imnosie függvényt a 'gaussian', a középérték, és az elhajlás paraméterekkel kell ellátni [14].

Ha a képet valahova továbbítani kell, olyan zaj keletkezhet, mely független a kép tartalmától. Hasonló zaj keletkezik a vidicon kamerákban is. Ezt a jel-független torzulást *additív* zajnak nevezzük, és a következő modellel lehet jellemezni

$$f(x, y) = g(x, y) + \nu(x, y),$$

ahol a ν zaj, és az eredeti g kép egymástól független változók.

A zaj erőssége sok esetben függ a jel erősségétől. Amennyiben a zaj erőssége nagyságrendekkel erősebb, mint a jel erőssége, akkor a következőt mondhatjuk ki:

$$f = g + \nu g = g(1 + \nu) \approx g\nu.$$

Ez a modell a *multiplikatív* zajt írja le. Jó példa a multiplikatív zajra a televíziós rasztertorzulás, mely a TV-ben lévő soroktól függ; egy sor közelében a zaj maximális, két sor között pedig minimális. Másik példa a multiplikatív zajra a film anyagának romlása, mivel a fényérzékeny emulziónál véges nagyságú ezüstport használnak. MatLab-ban a 'speckle', és egy eltérés paraméter megadásával lehet ilyen zajt előállítani.

Kvantálási zaj akkor keletkezik, amikor a kvantálás során kevés szintet adunk meg, például monochrome kép esetén 50-et. Ilyenkor hamis kontúrok jelennek meg. Ezek a zajok nagyon egyszerűen eltüntethetőek, például nem egyforma kvantálási intervallumok bevezetésével.

Az *impulzív* zaj olyan egymásttól független zajos pixellel torzítja el a képet, melynek világosságkód értéke jelentősen eltér a szomszéd képpontokétól.

A *só és bors* zaj egy telített impulzív zaj, például amikor fekete és/vagy fehér képpontokkal rontjuk el a képet. A só és bors zaj a bináris képeket is el tudják rontani. A MatLab-ban az imnoise függvényt a 'salt & pepper', valamint a zaj sűrűségének (azaz a képen előforduló zaj százalékának) megadásával lehet meghívni.

A *periódikus* zaj általában a kép megszerzése során keletkezik eletronikus/elektromechanikus interferencia miatt. Ezt a fajta zajt leginkább a frekvencia tartományban szokták szűrni.

Ezen kívül beépített függvények segítségével előállíthatunk Poisson eloszlást követő zajt a 'poisson' paraméter megadásával. Ennek az az érdekessége, hogy az adatokból alkotja meg a zajt, nem pedig a zajt adja hozzá a képhez.
Amennyiben semmit se tudunk a zaj természetéről, lokális előfeldolgozó metódusok alkalmazása javasolt. Amennyiben a zaj paraméterei nagyjából ismertek, kép helyreállító technikákat is lehet használni. A zajok paramétereit leginkább a kép Fourier spektrumát elemezve becsülik meg. [13]

4. fejezet

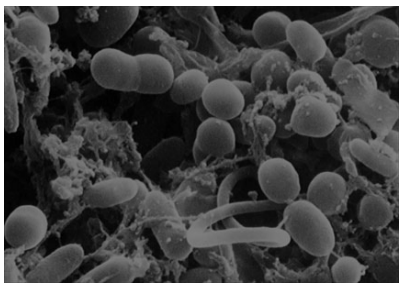
Hisztogram alapú eljárások

A nem megfelelő fénysűrűségből, illetve a fényveszteségből adódó kontrasztszegénység még ma is az egyik leggyakoribb képhiba, napjainkban is. Ez szerencsé esetben az egész képen egyenletesen jelentkezik, de előfordulhat, hogy a kép bizonyos részein is változik. A hisztogram alapú transzformációk célja a kontraszt növelése a világosságkódok eloszlásának megváltoztatásával [3].

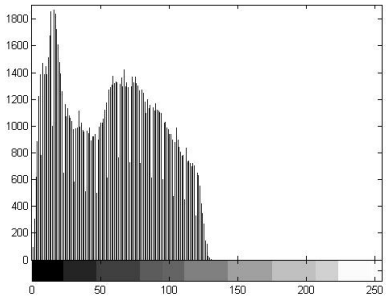
4.1. A hisztogram

Egy kép hisztogramja $h_f(z)$ a képen előforduló z világosságkódú gyakoriságot szolgáltatja - más szóval a világosságkód eloszlását szolgáltatja egy adott (szürgeskálás) képen. A képfeldolgozásban a hisztogram egy kép globális leírását jelenti. A hisztogramot leggyakrabban egy két dimenziós koordináta rendszerben ábrázoljuk, mint egy lépcső függvényt, ahol az x tengely jelöli a világosség-érték kódokat, az y tengely pedig a relatív gyakoriságot.

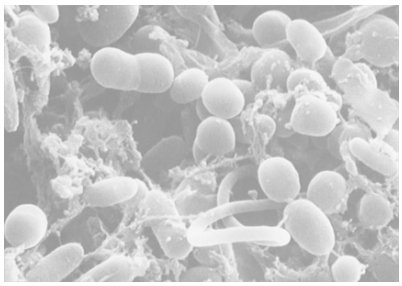
Egy kép hisztogramját nagyon könnyen, és gyorsan lehet szoftveresen számolni, ezért nagyon népszerű a valós idejű képfeldolgozás területén. Valamely tetszőleges felvétel hisztogramjának alapján nagyon sok általános jellemző megtudható róla, pl. világos összhatsú, vagy közel kétszintű, stb. Azon kívül, hogy a hisztogram statisztikát ad a képről, más képfeldolgozási alkalmazásokban is előszeretettel használják, például szegmentáláskor.



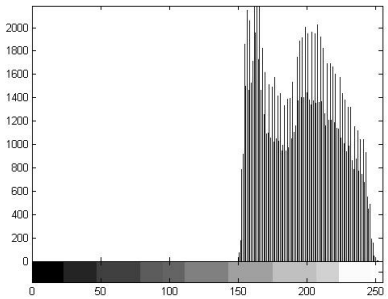
(a)



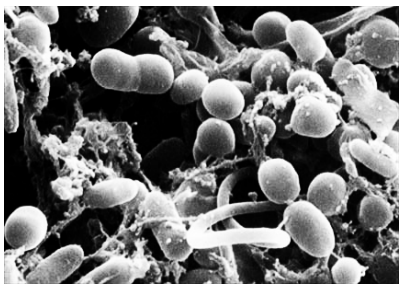
(b)



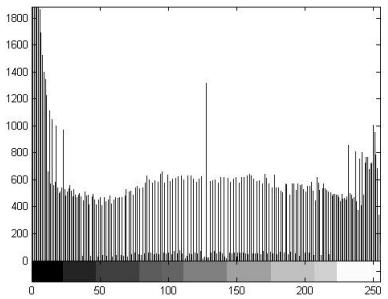
(c)



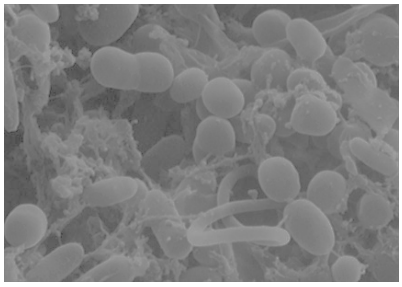
(d)



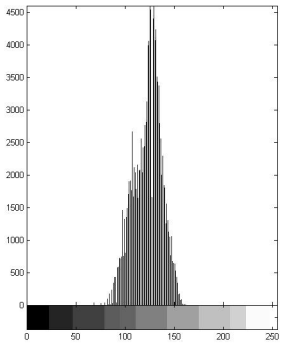
(e)



(f)



(g)



(h)

4.1. ábra. Képek, és azok hisztogramjai. Az (a) ábrán egy sötét-, a (c) ábrán egy világos-, az (e) ábrán egy magas kontrasztú, a (g) ábrán egy alacsony kontrasztú kép látható, mellettük pedig hisztogramjaik. Forrás: http://www.divediscover.whoi.edu, Chemosynthetic baktérium, 1977.

Általában véve viszont a hisztogram az egyetlen globális információ a képről, mely a rendelkezésünkre szokott állni. Mint látszik, sokminden leolvasható a grafikonról. A sötét kép hisztogramján a komponensek a szürke-skála (x tengely) a bal oldalán (sötét) koncentrálódnak, míg világos képen a jobb (világos) oldalon. Az alacsony kontraszttal rendelkező kép hisztogramján a komponensek középen, egy vékony „csíkban" koncetrálódnak, a skála közepén; a magas kontraszttal rendelkező kép hisztogramja pedig elterülő, nagy sávot foglal el a szürke-skálán. A következő feladatokhoz szokták felhasználni a hisztogramot: kép megfelelő megvilágításának beállításához, szürkeskálás transzformációkhoz, képszegmentáláshoz, stb.

Megjegyzés: több, különböző képnek lehet ugyanolyan hisztogramja; valamint a hisztogram nem változik, ha a képen egy objektum helyzete megváltozik. A *Matlab* beépített hisztogram megjelnítő eljárással rendelkezik (*imhist*), melyen elég csak a képet, és a skálázási paraméter(eke)t megadni.
Egy digitális kép hisztogramjának rengeteg lokális minimuma, és lokális maximuma szokott lenni, ami a további feldolgozást esetleg képes megnehezíteni. Ezt a problémát a hisztogram lokális simításával lehet kiküszöbölni, ami a szomszédos hisztogram-elemek átlagolásán alapszik. Az új hisztogramot a következő egyenlet segítségével kapjuk meg:

$$h'_f(z) = \frac{1}{2K+1} \sum_{i=-K}^{K} h_f(z+i),$$

ahol K egy konstans, annak meghatározására szolgál, hogy hány szomszédos elem figyelembe vételével határozzuk meg a simítást. Léteznek egyéb simító technikák is, pl. a Gauss módszer hisztogram simító eljárás.

4.2. Hisztogram-transzformáció

Reprezentálja az r változó a kép pontjainak világosság kódjait. Az elméleti tárgyalásokban r értékeit úgy tekintjük, hogy azok normalizáltak a $[0, 1]$ intervallumba ($0 \leq r \leq 1$), ahol az $r = 0$ a fekete, míg $r = 1$ a fehér színt reprezentálja. Ebben az esetben definiáljuk a hisztogram-transzormációt a következő képpen: $s = T(r)$.
Ennek a transzformációnak két feltételt kell teljesítenie [8]:

- $T(r)$ egyértékű, és a $[0, 1]$ intervalumban monoton növekvő
- $0 \leq T(r) \leq 1$, ha $0 \leq r \leq 1$

Az első feltétel biztosítja a szürkeségi skála helyes sorrendjét feketétől fehérig. A második feltétel pedig egy olyan leképezést garantál, amely konzisztens a pixelértékek megengedett tartományával.
Az nyilvánvaló, hogy az $r = T^{-1}(s)$, $0 \leq s \leq 1$ inverz transzformáció is kielégíti a fenti két feltételt.

A képpontok világosságkódjai valószínűségi eloszlást alkotnak. Jelöljük az eredeti kép, illetve a transzformált kép hisztogramját $p_r(r)$-el, valamint $p_s(s)$-el. Ha ismerjük a $p_r(r)$ és $T(r)$ értékeit, valamint $T^{-1}(s)$ teljesíti a fenti két feltételt, akkor $p_s(s)$-t a következő összefüggésből számíthatjuk ki:

$$p_s(s) = \left[p_r(r) \frac{dr}{ds} \right]_{r=T^{-1}(s)}$$

4.3. Hisztogram-kiegyenlítés

Az egyik legelterjedtem hisztogram-transzformáció a hisztogram kiegyenlítés.Ennek a módszernek az alkalmazásával a kontrasztszegény kép hisztogramja normalizálódik, a kevésbé domináló szürkeségi szintek több hangsúlyt kapnak [9].

Folytonos esetben:
Folytonos esetben a transzformációs függvény a következő alakban néz ki:

$$s = T(r) = \int_0^r p_r(w)dw, \;\; 0 \leq r \leq 1$$

Mivel az egyenlet jobb oldalán szereplő r valószínűségi változó kommutatív eloszlás függvényét simerjük fel, ezért ez az egyenlet kielégíti a két fenti feltételt.
Az s r-re vonatkozó deriváltja a következő alakban adható meg: $\frac{ds}{dr} = p_r(r)$, amiből könnyen adódik a következő kifejezés: $\frac{dr}{ds} = \frac{1}{p_r(r)}$
Visszahelyettesítve ezt a $p_s(s)$-re kapott egyenletbe a következő formulához jutunk:

$$p_s(s) = \left[p_r(r) \frac{1}{p_r(r)} \right]_{r=T^{-1}(s)} = [1]_{r=T^{-1}(s)} = 1, \;\; 0 \leq s \leq 1$$

Függetlenül attól, hogy milyen eloszlású valószínűségi változóból indultunk ki, az eredmény mindig egyenletes eloszlású lesz.

4.3.1. Diszkrét eset

A képet úgy transzformáljuk, hogy a világosságkódok hasonlítsanak egy egyenletes eloszlású valószínűségi változóra. Legyen egy kép, melyet hisztogram-kiegyenlíteni szeretnénk. Ez a kép tartalmazzon n db pixelt, és L szürke-árnyalatot [13].

Ezekre teljesül, hogy $0 \leq r_k \leq 1$, és $k = 0, 1, \ldots, L-1$. Jelölje r_k az n_k intenzitású képpontok számát a képen, a k-ik szürke-árnyalat gyakoriságát pedig a következőképpen határozzuk meg: $p_r(r_k) = \frac{n_k}{n}$, ahol $k = 0, 1, \ldots, L-1$. Diszkrét esetbe a transzformációs formula a következő képpen néz ki:

$$s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} \frac{n_j}{n}, \text{ ahol } 0 \leq r_k \leq 1, \;\; k = 0, 1, \ldots, L-1.$$

Az inverz transzformációs függvényt a következő alakban adhatjuk meg: $r_k = T^{-1}(s_k)$, $0 \leq s_k \leq 1$, ahol feltételezzük, hogy mind a $T(r_k$, mind pedig a $T^{-1}(s_k)$ teljesíti a korábban említett két transzformációs feltételt.
Miután kiszámoltuk az új s_k értéket, már csak egy lépés marad hátra. Előfordulhat, hogy az újonnan kiszámolt érték nem illeszkedik az eredeti r_k értékre. Ekkor az s_k értékeit a meglévő világosságkódokkal közelítjük, azaz minden új értékhez a hozzá legközelebbi régi értéket kell hozzárendelni.

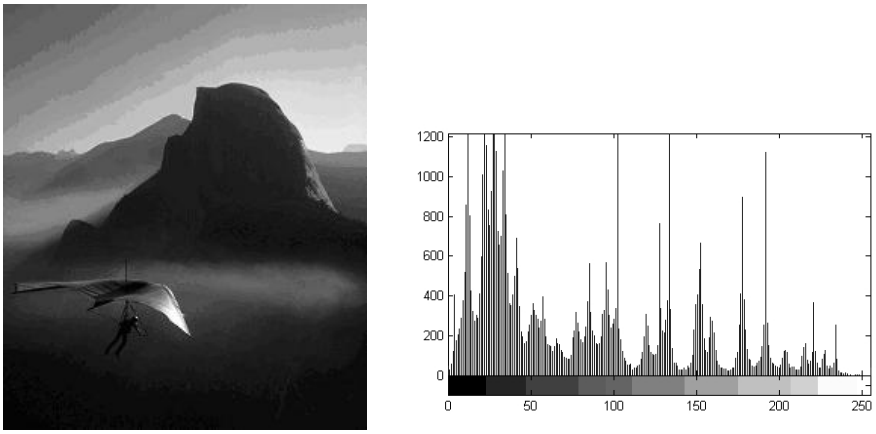
A hisztogram kiegyenlítés midnen olyan esetben jól alkalmazható, ahol a kép hisztogramja egy nagyobb, vékony csúcsot tartalmaz. Az eljárás ilyenkor széthúzza a csúcsot, ezzel elősegítve a kép kontrasztjának javulását.

Példa (Rafael C. Gongalez: Digital Image Processing) [13]:
Legyen egy 64x64-es, 8 szintű kép. A lenti táblázat oszlopai rendre tartalmazzák az intenzitás értékeket, az ilyen intenzitással rendelkező pixelek számát, az intenzitásértékek relatív gyakoriságát, a transzformációs formulával kapott s'_k új intenzitás értékeket, valamint ezen értékeket a régi intenzitásértékekkel való közelítését.

r_k	n_k	$p_r(r_k)$	s_k'	s_k
r_0=0	790	0.19	0.19	1/7
r_1=1/7	1032	0.25	0.44	3/7
r_2=2/7	850	0.21	0.65	5/7
r_3=3/7	656	0.16	0.81	6/7
r_4=4/7	329	0.08	0.89	6/7
r_5=5/7	245	0.06	0.95	1
r_6=6/7	122	0.03	0.98	1
R_7=1	81	0.02	1.00	1

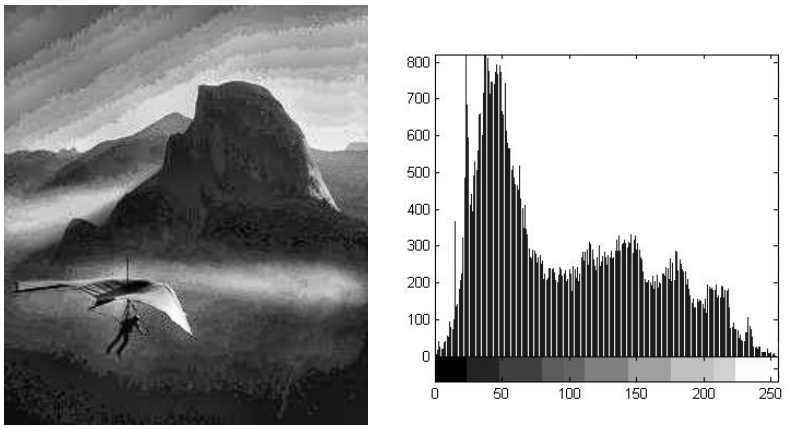
4.2. ábra. Egy 64x64-es kép hisztogramjának kiegyenlítése

Egy saját példán bemutatva mindezt: Legyen adott egy 241x282-es, 256 szintű kép. A 4.3. ábrán az eredeti kép látszik a hisztogramjával, majd a 4.4. ábrán a hisztogram-kiegyenlített képe látható. A hiszogram-kiegyenlítés a *Matlab* saját, beépített eljárásainak segítségével történt meg.



(a) (b)

4.3. ábra. Az eredeti kép, és annak hisztogramja. Az eredeti kép forrása: http://viewimages.com



(a) (b)

4.4. ábra. A már kiegyenlített kép, és annak hisztogramja.

4.3.2. Multispektrális eset

Az eddig bemutatott eljárás minden szürke skálás képekre egyszerűen alkalmazható, ahol a kép mindössze egy színcsatornát tartalmaz; de felmerülhet a kérdés, hogy a hisztogram transzformációt (pl. kiegyenlítést) mennyiben kell változtatni színes képbemenet esetén, ahol a forrás három (vagy több) színcsatornát tartalmaz. A manapság leginkább elterjedt digitális képtárolási forma a 24 bites (TrueColor) RGB reprezentáció, ahol a pixel végső árnyalatát a három színkomponens (vörös, zöld, kék) aránya határozza meg [9].

Az egyik legegyszerűbb megoldás az előbb tárgyalt algoritmus kis átalakításával könnyen végrehajtható: bontsuk fel a képet színkomponenseire, majd ezekre egyesével alkalmazzuk a hisztogram kiegyenlítő függvényt, végül a feldolgozott színcsatornákból újból „összerakjuk" a képet. Mivel az ezen a módszeren alapuló hisztogramkiegyenlítés a színkomponenseket külön-külön, egymástól függetlenül transzformálja, ezért ez a színek torzulásához vezethet (4.5.-4.6. ábra).

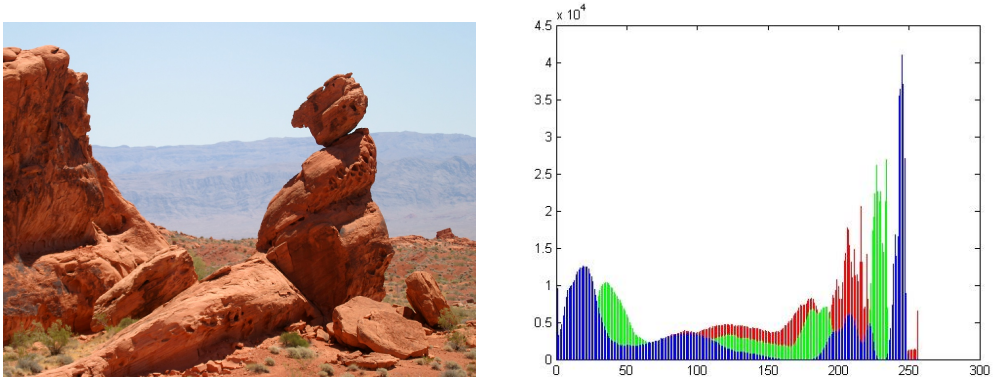
Alkalmazhatunk viszont egy másik módszert is, mely segítségével kapcsolatot teremthetünk a színkomponensek között. Csupán tekinteni kell a színcsatonák transzformációs függvényeinek olyan súlyozott átlagát, ahol a súlyok összege 1. Az így kapott transzformációs függvényt aztán az összes színkomponensre külön-külön alkalmazhatjuk.
Be kell látni, hogy ez az eljárás megőrzi a transzformációs függvények két tulajdonságát: a monotonirást, valamint hogy az r értéke 0, és 1 közé essen ($0 \leq r \leq 1$).

Az utolsó feltétel - mely r értékérevonatkozik - könnyen belátható: a függvények értékének maximuma 1 lehet, ezért ezek átlaga is 1 (a használt súlyozás miatt az intenzitások értékkészlete nem változik). A monotonitás belátásához előbb azt kell belátni, hogy két monoton függvény összege is monoton.
Legyen az f, és g két olyan függvény, mely kielégíti a monoton növekedés feltételét; azaz ha az $f{:}R \to R$ monoton növekvő, és $x < y$ ($x, y \in R$), akkor $f(x) < f(y)$. Két függvény összege alatt a következőt értjük: $(f+g)(x) = f(x)+g(x)$. Amennyiben mindkét függvény konstans (minden pontban ugyanazt az értéket veszik fel), akkor ezek összege is monoton (növekvő, és egyben csökkenő is). Ha az egyik függvény konstans, a másik viszont nem (de monoton, a feltételek miatt), akkor ez úgy tekinthető, mintha a nem konstans függvény értékeit minden pontban egy konstans értékkel növeltük volna meg. Amennyiben egyik függvény sem konstans, akkor az egyik függvény értékeit növeljük a másik függvény növekvő értékeivel. Amint látszik, az eredményül kapott függvény is monoton növekvő lesz. Könnyen belátható, hogy nem csak kettő, hanem több, monoton függvény összege is monoton marad.

Az alábbi kis MatLab program segítségével könnyedén megjeleníthetjük az RGB kép hiszogramját [19], [14]:

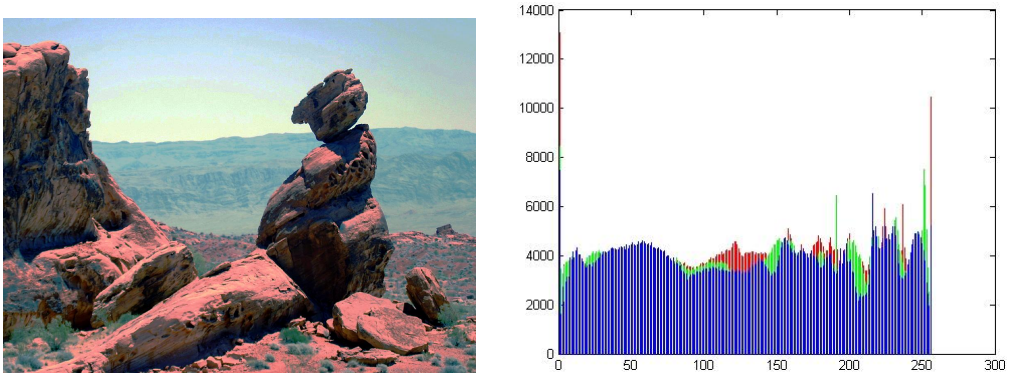
```
nBins = 256;
rHist = imhist(I(:,:,1), nBins);
gHist = imhist(I(:,:,2), nBins);
bHist = imhist(I(:,:,3), nBins);
```

```
hFig = figure;
h(1) = stem(1:256, rHist);
h(2) = stem(1:256 + 1/3, gHist);
h(3) = stem(1:256 + 2/3, bHist);
set(h, 'marker', 'none');
set(h(1), 'color', [1 0 0]);
set(h(2), 'color', [0 1 0]);
set(h(3), 'color', [0 0 1]);
```



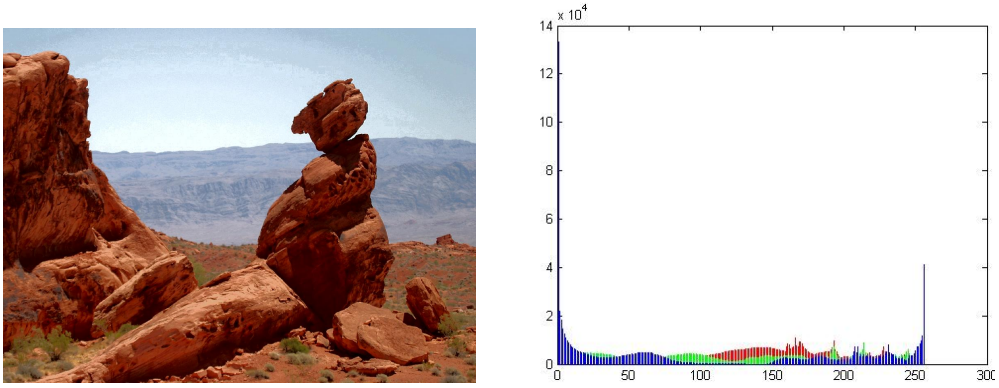
(a) (b)

4.5. ábra. Az eredeti kép, és annak hisztogramja. Az eredeti kép forrása: Richard Seaman, The Flying Kiwi weboldala (http://www.richard-seaman.com)



(a) (b)

4.6. ábra. A színcsatornák egymástól független kiegyenlítésével keletkezett ábra, valamint annak hisztogramja.



(a) (b)

4.7. ábra. A színkomponensek között kapcsolat lett teremtve a kiegyenlítés során, az ennek segítségével előálló ábra, valamint annak hisztogramja.

4.4. Lokális hisztogramkiegyenlítés

A hisztogramkiegyenlítés az eljárás globális volta miatt, eredménye nem minden esetben kielégítő, mivel vannak olyan esetek, amikor a kép egy kisebb részletét kell feldolgozni, mivel a kép megvlágítása nem egyenletes (ld. Gonzalez [13], [14]). Ilyenkor a globális eljárások nagy valószínűséggel nem vezetnek kielégítő eredményre. Ennek kiküszöbölésére fejlesztettek ki egy olyan eljárást, mely egy pixel lokális szomszédságán belül dolgozza fel a képrészletet. Ennek egy lehetséges megvalósítása a következőképpen néz ki :

$$g(x,y) = A(x,y)[f(x,y) - m(x,y)] + m(x,y),$$

ahol

$$A(x,y) = k\frac{R}{\sigma(x,y)} \qquad 0 \leq k \leq 1,$$

valamint $m(x,y)$, illetve $\sigma(x,y)$ jelöli az (x,y) középpontú maszk intenzitásértékeinek az átlagát, illetve a szórását; míg R a kép szürkeségi fokozatainak globális átlaga.

4.5. Küszöbölési technikák

A küszöbölési technikák alapját a hisztogram adja. A kép intenzitástartományok alapján történő szétbontását küszöbölésnek nevezzük. A cél a képen fellelhető intenzitás értékek nagy számú csökkentése, miközben a kép vizuális információ tartalma legkisebb mértékben változzon.A vágással gyors, és látványos eredményeket lehet elérni, azonban ha a kép zajokkal torított, a vágások valószínűleg nem fognak a kívánalmaknak megfelelő eredményeket hozni, pl. hogy kiválasszunk egy olyan lényeges régiót, melyet fel szeretnénk dolgozni. Ilyenkor ugyanis a zaj megváltoztatja a pixel világosságkódját. A küszöbölés során a határérték alapján az algoritmus a zajmentes képen azonos tartományba tartozó pixeleket a zaj miatt különböző osztályokba, illetve az eltérő osztályúakba tartozóakat azonosakba sorolhatja. Ezért a vágás előtt célszerű a zajokat valamilyen zajszűrő alkalmazással eliminálni [12], [3].

A küszöbölés a képszegmentálás egyik legegyszerűbb megvalósítása. Egy n szintű küszöbölés elvégézéséhez jelöljük el a küszöbértékeket: $k_1, k_2, \ldots, k_n$-el, az eredeti kép világosság kódjait pedig jelölje: $y_1 < y_2 < \cdots < y_l$. Legyen a k_1 küszöb értéke az eredeti kép legkisebb világósságkódjával, míg k_n az eredeti kép legnagyobb világosságkódjával egyenlő, azaz $k_0 = y_1$, és $k_n = y_l$. Ekkor az n-küszöbölés alatt a

$$f'(x, y) \leftarrow \{r_i | k_{i-1} \leq f(x, y) \leq k_i, i = 1, 2, \ldots, n\}$$

hozzárendelést értjük, ahol r_i az i-edik küszöbhöz előre megadott világosságkód. A küszöbölésnél a legfontosabb paraméter a határértékek (küszöbök) megadása, ami általában a hisztogram lokális szélsőértékeinek (leggyakrabban a minimum értékeinek) meghatározásával történik. A küszöbök kiválasztása történhet manuálisan (próba-hiba módszer, mely nagyon jó eredményt képes produkálni, de megköveteli az interaktivitást), vagy autómatikusan (algoritmus segítségével), bár ez utóbbi többszintű küszöbölés esetén bonyolultá, és pontatlanná válhat, mivel a legtöbb hisztogramnak nincsenek egyértelműen definiált „völgyei". Azt az eljárást, amikor a kép különböző régióin különböző küszöböket használunk *adaptív küszöbölésnek* nevezzük.

Szürke skálás kép küszöb(einek) kiválasztásának egyik legegyszerűbb megoldás az, ha kép képpontjainak világosságkód-átlagához viszonyítva határozzuk a küszöböt, hiszen a háttértől világosabb/sötétebb képpontok az előtér pontjai lesznek. Ez zajmentes, egyszínű háttérrel, és objektumokkal rendelkező kép esetében nagyon egyszerűen, jól megvalósítható, de a valóságban, az esetek többségében nincsenek meg ezek az ideális feltételek. Egy olyan küstöb-meghatározó algoritmus megadása a cél, mely elég egyszerű, nem igényel a képről plussz információt, és zajos képeken is jól működik! Egy ilyen iteratív, két szintre vágó algoritmus a következőképpen néz ki (Gongalez és Woods)[13], [12]:

1. Válasszunk egy K küszöböt, mely lehet akár egy véletlen érték, vagy mármely más módon meghatározott pixel intenzitás.

2. Szegmentáljuk a képet előtér, és háttér képpontokra, így előáll két halmaz:
   –$G_1 = \{f(x,y)|f(x,y) > K\}$ (előtér képpontok)
   –$G_1 = \{f(x,y)|f(x,y) \leq K\}$ (háttér pontok), ahol $f(x,y)$ az x, y koordinátájú pixel intenzitás értékét szolgáltatja

3. Számoljuk ki a halmazokba tartozó képpontok világosságkódjának átlagait:
   –m_1=G_1 halmaz átlagértéke
   –m_2=G_2 halmaz átlagértéke

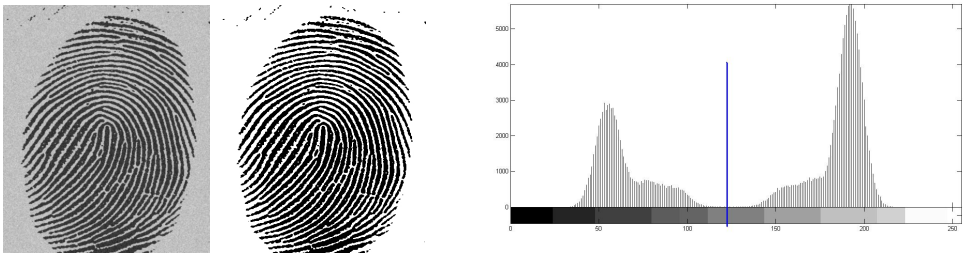
4. Határozzuk meg az új K' küszöböt az m_1 ésm_2 értékek segítségével:
   $K' = dfrac12(m_1 + m_2)$

5. Menjünk vissza a második lépésre, ám ezúttal az új küszöbértékkel hajtsuk végre a lépéseket. Egészen addig iteráltassunk, amíg két, egymást követő iteráció ugyanazt a küszöbértéket nem adja.

Ennek az algoritmusnak egy *MatLab* implementációja a kövtkezőképpen néz ki:
T=0.5(double(min(I(:))) + double(max(I(:))))*
done=false
while˜done
g= I>=T
Tnext=0.5(mean(I(g))+mean(I(˜g)))*
done=abs(T-Tnext)<=0.5
T=Tnext
end

ahol I változóban van eltárolva a beolvasott kép, T pedig a kiszámított küszöb.

Igen gyakori eset a kép *binarizálása*, más néven *két szintre vágása*, amikor a képpontokatat előtér, illetve háttérpontokká minősítjük. A szintrevágás művelete jelenti a klasszikus egy-, vagy többszintű küszöbölést. Az egyszintű küszöbölés esetén rögzítünk egy K küszöböt, majd annak megfelelően csoportosítjuk a képpontokat, hogy ehhez a küszöbhöz a pixel intenzitásértéke a hisztogramon hol helyezkedik el. Mint említettem, a küszöb meghatározása nem egyszerű, és egyértelmű feladat; amennyiben a küszöbölő eljárás nem nyújtja a megfelelő eredményt a képen, globálisan, akkor érdemes a képet szegmensekre osztani, és ott lokálisan elvégezni a szintrevágást.
A 4.8. ábrán egy könnyen binarizálható képet láthatunk. A hisztogrammon egyértelműen meg lehet határozni azt a pontot, ahol a vágást érdemes elvégezni.



(a) (b) (c)

4.8. ábra. Az eredeti, a két szintre vágott ujjlenyomat, és a felhasznált hisztogram. A kép forrása: http://nalts.wordpress.com

A *sávkivágás* a küszöbölés egy speciális esete, ahol az intenzitáshisztogram segítségével egyszerűen kiemeljük a fontosnak tartott intenzitás értékeket. Ehhez két küszöbre van szükség (K_1 és K_2), valamint egy intenzitás értékre (r). Ezek után a sávkiemelés a következőképpen néz ki [8]:

$$f'(x,y) = \begin{cases} f(x,y), \text{ ha } K_1 \leq f(x,y) \leq K_2 \\ r \qquad\quad \text{egyébként} \end{cases}$$

Azaz a két küszöb által behatárolt intenzitásérték tartományt az eljárás nem változtatja meg, a többi intenzitésértéket pedig r világosságkódúra módosítja.

(a) (b) (c)

4.9. ábra. A híres Lena Söderberg kép, valamint a sávkivágással kapott eremény, és a felhasznált hisztogram. Forrás: Playboy magazin, 1972. November

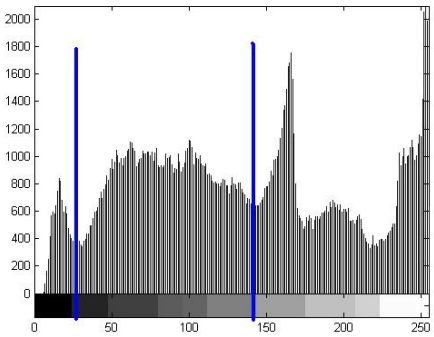
Az intenzitáshisztogram segítségével történő *sávkiemelés* azt jelenti, hogy a hisztogramon, a két előre definiált küszöb közé eső intenzitásértékekkel rendelkező képpontok világosságkódjait egy másik, előre definiált világosságkód-értékekre változtatjuk [8]:

$$f'(x,y) = \begin{cases} r, \text{ ha } K_1 \leq f(x,y) \leq K_2 \\ f(x,y) \qquad \text{egyébként} \end{cases}$$

A *sávkivágás kiemeléssel* pedig azt jelenti, hogy az eddig változatlanul hagyott pixelek világosságkódját egy előre definiált intenzitásúra változtatjuk át, azaz:

$$f'(x,y) = \begin{cases} q, \text{ ha } K_1 \leq f(x,y) \leq K_2 \\ r \qquad \text{egyébként, ahol } q \text{ egy előre definiált intenzitásérték} \end{cases}$$

(a) (b)



(c)

4.10. ábra. Egy kép, a sávkivágás kiemeléssel kapott eredménye, és a felhasznált hisztogram. Forrás: Massachusetts-i Egyetem, Számítógéptudományi Kar honlapja (http://www.cs.umass.edu)

4.5.1. Többsávú képek esetében

Előfordulhatnak olyan esetek, amikor egy jelenetről felvett több képen keresztül kell küszöbölni. Legáltalánosabb példa erre a színes képek esete, amelyen a fény különböz frekvenciái jelennek meg. Űrfelvételek esetében a képek több infrasávot tartalmazhatnak, mely különböző információkat hordoznak (különbőző anyagfajták/gázok, és azok mennyiségei, meteorológiai információk, stb.) [9]
Színes képek esetében az egyik legegyszerűbb megoldás a szegmentálás elvégzésére, ha a kép színcsatornáit szétbontjuk a három alapszínre (vörös, zöld, és kék), majd ezeken egyesével elvégezzük a megfelelő küszöbölő eljárást, mint azt az intenzitás hsztogram esetében is tettük. A kapott képeket aztán majd egy logaikai ÉS művelettel „összekapcsoljuk" (azaz egy pixel az előtérhez tartozik, ha mindhárom komponense [RGB] a „kiválasztott" tartományba esik). Ez logikailag megfelel egy olyan eljárásnak, mely során a képet színcsatornákra bontjuk, egymástól függetlenül bináris képeket állítunk elő belőlük, majd egy logikai ÉS müvelettel összakapcsoljuk. Ez az eljárás viszont közel sem biztos, hogy az elvárt, optimális eredményt fogja előállítani, ha egymástól függetlenül határozzuk meg a küszöbértékeket. Ennek több oka is van: a vörös, a zöld, és kék intenzitások azt reprezentálják, hogy a felvevő készülék hogyan működik, és az adatok milyen formában vannak eltárolva; de teljesen függeltenek attól, hogy az emberek hogyan „észlelik" a színeket.
Egy képet nem csak RGB, hanem HSI (Hue - árnyalat, Saturation - telítettség, Intensity - intenzitás) összetevőire is bonthatunk. Gyakran előfordul, hogy ha csak az egyik összetevőt küszöböljük, akkor is megfelelő eredményt kapunk; de a legtöbb esetben az összes rendelkezésre álló információt fel kell használni. A három, vagy több dimenzióban történő küszöbölés - legyen szó akár az RGB-ről, akár HSI-rő - nagyon nehéz feladat, hiszen az információ a csatornákban van együttesen kódolva.
Egy lehetséges rekurzív küszöbölési eljárás multispektrális képekre:

1. Inicializáljuk a képet: tekintsük az egészet, mint egyetlen összetartozó régiót.

2. Minden egyes (szín)csatornára ki kell számítani a hisztogarmot. Meg kell keresni a legkiemelkedőbb csúcsot minden egyes hisztogramon. Ezek után két küszöböt meg kell határozni, ami a maximum két oldalán elhelyezkedő egy-egy lokális minimum lesz. Szegmentáljuk a régiókat az összes csatornán alrégiókra az előbb meghatározott küszöbök segítségével.

3. Addig ismételjük a második lépést, amíg az összes régió már csak 1 csúcsot tartalmaz.

Ennél jobb eredményt érhetünk el, ha a többdimenziós hisztogram elemezését halytjuk végre, a (2)-es lépésben leírt hisztogramok analízise helyett. Leggyakrabban egy pixel, vagy a kis szomszédságán értelmezett többcsatornás szegmentálás az n csatornában az n dimenziós szürke-skálás vektorokon alapszik. Ezt a fajta szegmentációt széles körben alkalmazzák a gépi látás területén. Általánosságban véve a régiókat olyan pixelekből képezik, amelyek minden színcsatornában hasonló tulajdonságokkal, hasonló n dimenziós vektorokkal rendelkeznek rendelkeznek.

5. fejezet

Környezeti átlagolás

A környezeti átlagolás az egyik legegszerűbb képsimító eljárás. A simító eljárásokat arra használjuk, hogy eliminálju a rossz mintavételezésből, vagy az átviteli csatornában keletkező hibákat. A környezeti átlagolás valójában egy képtartományban végrehajtható konvolúciós maszkolás. Ehhez definiálni kell egy maszkot (általában $n \times n$-es), amit minden pixelén „át kell mozgatni". A képen, az (x, y) koordinátájú képpont világosságkódját az előre definiált környezetének világosságkódjainak átlagaként kapjuk meg. Azaz ha $f(x, y)$ volt az pixel eredeti, $g(x, y)$ pedig az új világosságkód érték, akkor a kettő közötti összefüggest az alábbi képlet adja [9]:

$$g(x, y) = \frac{1}{N} \sum_{(m,n)\in S} f(m, n)$$

ahol S a környezeti pontok koordinátáinak halmaza, N pedig a környezeti pontok számossága (pl. egy 3×3-as maszk használata esetén $N = 9$). A szomszédság nem kizárólag négyzet alakú lehet, de a gyakorlatban ez a legkönnyebben megvalósítható, ezért ez a legelterjedtebb. A gyakorlatban leginkább elterjedt környezet a 3×3-as, vagy a 5×5-ös maszk, melynek a centrális eleme az (x, y) koordinátájú pixel [3]. A kép széleinél, és sarkainál fellépő számolási problémát a gyakorlatban többféleképpen szokták orvosolni: vagy egyszerűen háttér objektumnak tekintik a képhez nem tartozó képpontokat; vagy az éleket tükörként használva „megduplikálják" a pixeleket, vagy úgy tekintik a képet, mintha az folytonos lenne, azaz a kép jobb széle a bal oldalon folytatódna, és a fennti az lenn.

Az eljárás nagy hátránya, hogy erősen homályosítja a képet. Minél nagyobb maszkot használunk, az eredményül kapott kép annál elmosódottabb lesz; de a kis só-bors zajok is eltűnnek. Amennyiben az eljárás csak részben hozná létre az elvárt eredményt, az esetben az eljárást iteráltatni is lehet. Az iteráció hatására a kép egyre elmosódottabb lesz, és egy idő után mindenhol egy konstans árnyalatot kapnánk. A való életben ritkán használunk 5×5-ösnél nagyobb méretű szűrőt, mivel a szűrőméret növelésével egyenes arányban romlik a kép minősége is, egyre homályosabbá válik; mindamellett a számítási

idő is megnövekszik.
Az eljárásnak létezik egy javított változata, mely az erős homályosítást hívatott kijavítani: egy küszöböt kell bevezetni, és a helyettesítést csak akkor kell elvégezni, ha a eredményül kapott környezet átlaga, és a képpont színárnyalata közötti különbség meghaladja ezt a küszöb mértékét. Az erre szolháló matematikai képlet a következőképpen néz ki [9]:

$$g(x,y) = \begin{cases} \frac{1}{N} \sum\limits_{(n,m)\in S} f(n,m), \text{ ha } |f(x,y) - \frac{1}{N} \sum\limits_{(n,m)\in S} f(n,m)| > K, \\ f(x,y), \text{ egyébként} \end{cases}$$

ahol $g(x,y)$ a képpont új-, $f(n,m)$ a képpont eredeti intenzitása, K pedig az előre megadott küszöbérték.
Egy másik megoldás az elmosódottság csökkentésére, ha súlyozzuk a maszkban felhasznált szomszédokat. Súlyozott szomszédság esetén a felhasználandó képlet a következőképpen alakul [11]:

$$g(x,y) = \frac{\sum\limits_{(m,n)\in S} w(m,n) \cdot f(m,n)}{\sum\limits_{(m,n)\in S} w(m,n)}$$

ahol $w(m,n)$ jelöli a képpontok súlyozását. Egy lehetséges súlyozás, mely kevésbé torzítja el a képet:

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

A középső 4-es súly gondoskodik arról, hogy a centrális elem dominájon, és így ne homályosodjon annyira el a kép. Az átlós szomszédok azért „csak" 1 értékűek, mert távolabb helyezkednek el a centrális elemtől, mint a többi szomszédos pixel (valójában $\sqrt{2}$ egységnyire). Végezetül a súlyok összege 16, ami 2 hatávnya, így nagyon könnyen lehet vele osztani a számítógépes alkalmazásokban.
Analízisekből kiderült, hogy a „leghatékonyabb" módszer, ha a Gauss-i súlyozást használunk. Ez olyan eégsz számok halmaza, mely nagyjából a Gauss-i függvényt „ábrázolja" minden sorban, oszlopban, vagy átlóban, mely a centrális elemen megy keresztül. A maszk méretét akkorának szokták választani, hogyha új sorokat/oszlopokat fűznénk hozzá, az elég kis súlyokkal bővítené azt - ideális esetben nulákkal -; természetesen a sarkokban mindenképp nullák fognak megjelenni, mivel ezek elég távol vannak a centrális elemtől.
Az egész számok (súlyok) kiválasztása elég nehéz feladat, mivel a fő feladat, hogy megközelítsék a Gauss-i görbe jellegzetességét, de a súlyok összege maradjon elég alacsony ahhoz, hogy számítógéppel könnyen, és gyorsan lehessen velük számolni (Russ, 1995d). Egy kisebb maszk többszöri-, vagy két, esetleg több maszk egymás utáni alkalmazása egy képen ekvivalens lenne azzal, mintha egy nagyobb maszkot használtunk volna, amit a két

maszk konvolúciójaként kapnánk. Lássunk néhány példat a leginkább elterjedt súlyozásokra [11]:

(3×3) pixel:

$$\begin{pmatrix} 1 & 4 & 1 \\ 4 & 12 & 4 \\ 1 & 4 & 1 \end{pmatrix}$$

(5×5) pixel:

$$\begin{pmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 7 & 11 & 7 & 2 \\ 3 & 11 & 17 & 11 & 3 \\ 2 & 7 & 11 & 7 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{pmatrix}$$

(9×9) pixel:

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 & 3 & 3 & 2 & 1 & 0 \\ 1 & 2 & 3 & 6 & 7 & 6 & 3 & 2 & 1 \\ 1 & 3 & 6 & 9 & 11 & 9 & 6 & 3 & 1 \\ 1 & 3 & 7 & 11 & 12 & 11 & 7 & 3 & 1 \\ 1 & 3 & 6 & 9 & 11 & 9 & 6 & 3 & 1 \\ 1 & 2 & 3 & 6 & 7 & 6 & 3 & 2 & 1 \\ 0 & 1 & 2 & 3 & 3 & 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Érdemes megjegyezni, hogy a simítás műveletét fel lehet gyorsítani, ha egy nagyobb szűrő helyett két kisebbet alkalmazunk. Például egy 15×15 méretű szűrő helyett - amihez 255 szorzást, és összeadást kell elvégezni - alkalmazhatnánk egy vízszintes Gauss-i simítást a súlyok lineáris tömbjével (mely 15 szorzást, és összeadást tartalmaz), majd utána alkalmazhatnánk egy vízszintes Gauss-i simítást (mely szintén 15 szorzás, és összeadás; ami összesen 30 műveletet tartalmaz).
Olyan alkalmazás is létezik, ahol a súlyok nem szimmetrikus elrendezésüek, sőt egyes súlyok nem pozitív értékűek. Ilyenkor a kernel (maszk) implementációja ugyanolyan, mint az előző esetekben, annyi különbséggel, hogy a negatív súlyok a normalizálást képviselik; melyeket kizárólag a pozitív értékek összegével osztunk el (mivel ilyenkor a súlyok összege zérus).
Ez a fajta alkalmazás eltávolítja a zajokat, de sajnos ezzel együtt elmossa az éleket, és csökkenti a kontrasztot. A negatív súlyok pont az élek elmosódottságát hívatottak szolgálni. Ugyanezt a módszert már nagyon régóta használják az egy dimenziós jelprofilok simításánál, mint például röntgen diffrakciós mintáknál, vagy időben változó elektromos jeleknél.
A gyakorlatban a kisebb-, vagy a szeparálható - mint a Gauss-i - kernelek segítségével elég

gyors alkalkalmazások implementálhatóak. Viszont azok a kernelek, melyeknek a mérete eléri a 15 × 15-ös méretet (ez a csak egy nagyságrend, a pontos érték számítógépenként változik, pl. a képet képes-e egészében a memóriában tárolni, vagy a háttértárról kell beolvasni; több processzort képes-e a művelet végrehajtásánál használni, vagy csak egyet; stb.), akkor a frekvenciatartományban végzett szűrés gyorsabb lesz. Az alábbi *MatLab* kód segítségével könnyedén megvalósítható a Gauss-i szűrés (az eljárás egy 10 × 10-es maszkot használ [19], [14]:

```
function y=gaussian_filter(x, sigma);

% A Gauss-i szuro parameterei:
n1=10;sigma1=sigma;n2=10;sigma2=sigma;

filter_mask=d2gauss(n1,sigma1,n2,sigma2);
y=conv2(x,filter_mask,'same');

% Ez az eljaras a 2D-s Gauss-i szurovel ter vissza, aminek a merete n1*n2;
function h = d2gauss(n1,std1,n2,std2)
for i = 1 : n2
for j = 1 : n1
u = [j-(n1+1)/2 i-(n2+1)/2]';
h(i,j) = gauss(u(1),std1)*gauss(u(2),std2);
end
end
h = h / sqrt(sum(sum(h.*h)));

function y = gauss(x,std)
y = exp(-x^ 2/(2*std^2)) / (std*sqrt(2*pi));
```

A függvények segítségével kapott képek eredményei:

(a) (b) (c)

5.1. ábra. Az eredeti, zajos kép, valamint a sigma=1, és sigma=2 paraméterek segítségével kapott eredményük (a képre a zaj az imnoise(I,'salt & pepper',0.03)) függvény segítségével került). A kép forrása: http://www.wander-woman.com

6. fejezet

Medián szűrés

Mint azt már említettem, a környezeti átlagolásnak több hátránya is van. Az egyik az élek elmosása, amit küszöbölési technikával javítani lehet, de ehhez meg kel találni a megfelelő küszöbértéket. A másik igen fontos dolog, hogy eddig nem szereplő, új színárnyalatok kerülhetnek bele a képbe, ami esetleg megnehezítheti a kép további feldolgozását [8], [9].

A zajok eltávolításának egy másik, alternatív módja a medián szűrés. Ez az eljárás nagyon hatékony, ha a zaj alakja erős, tüskés komponensekből áll, és az algoritmus használata után meg akarjuk tartani az élek „élességét". Ennél a módszenél is meg kell adni egy környezetet (általában négyzetes alakú). Tehát adva van az (x, y) koordinátájú centrális elem, és annak egy $n \times n$-es környezete (ahol n általában páratlan). A környezetben található pixelek világosságkódját nagyság szerint növekvő (vagy csökkenő) sorrendbe rendezzük (világosságkód szerint), majd az így kapott elemsorozatból kiválasztjuk a középső elemet, és ezzel az értékkel helyettesítjük az (x, y) koordinátájú pixel világosságkódját.

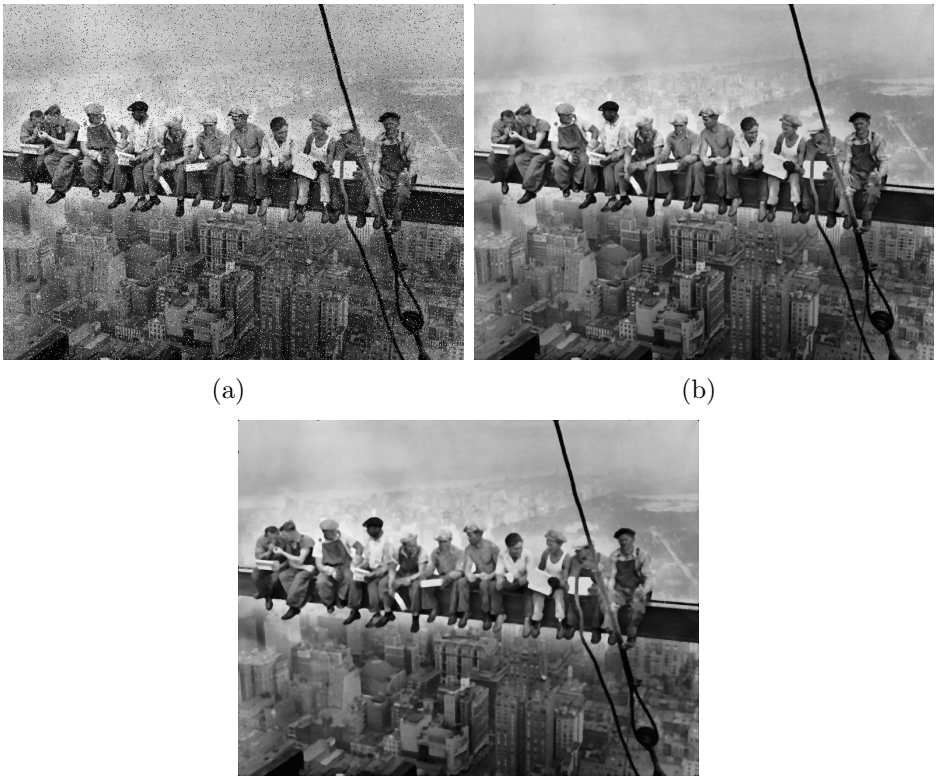
A módszer azon alapszik, hogy a kiugró intenzitásértékű zajok a rendezés befejeztével a sorozat legelejére, vagy a legvégére kerülnek, a középső (medián) elem pedig az adott képtartomány átlagos intenzitását mutatja. Ennél a módszernél is, akárcsak a környezei átlagolásánál, akár nagyobb környezetet is lehet definiálni (5×5, vagy 7×7, stb.), és ezt a módszert is lehet ugyanarra a képre iteráltatni. Viszont ellentétben a környezeti átlagolásnál, nem biztos, hogy az iteráció után egy konstatns intenzitású képet kapnánk. Példa medián szűrésre: Legyen adott a következő 3×3-as mátrix, melynek a centrális eleme kiugró értéket tartalmaz, ezért ki kell cserélni [3].

$$\begin{pmatrix} 3 & 5 & 2 \\ 4 & 19 & 6 \\ 8 & 1 & 4 \end{pmatrix}$$

Rendezve az elemeket kiderül, hogy a 19-es érték helyére a 4-es kerül, tehát a medián szűrés eredménye:

$$\begin{pmatrix} 3 & 5 & 2 \\ 4 & 4 & 6 \\ 8 & 1 & 4 \end{pmatrix}$$

Ha a kapott centrális elemre újra alkalmaznánk a medián szűrést, és a környezet nem változott volna, akkor a centrális elem is maradt volna 4 értékű.



(a) (b)

(c)

6.1. ábra. A kép só-bors zajjal lett terhelve, majd egy 3×3-as, és egy 5×5-ös medián maszk lett alkalmazva. Az eredeti képet készítette: Charles Ebbett, 1932., New York, Forrás: Bettmann Archívum

7. fejezet

Képek átlagolása

A módszer használatának előfeltétele az hogy az adott jelenetről több felvétel (képsorozat) készüljön (pl. nagysebességű kamera segítségével). Az ugyanis előfordulhat, hogy a kép felvétele esetén valami additív zaj rárakódik a felvételre, azaz ha az eredeti (ideális, zajmentes) képet f-el jelölöm , az adott képre rárakódó zajt pedig μ-vel, akkor a kettő összegéből kapott kép a [8], [3]:

$$g(x,y) = f(x,y) + \mu(x,y)$$

Ekkor, ezen módszer alkalmazása esetén a több, zajos képeket átlagoljuk, az eredmény pedig egy zajoktól mentes képek kapunk. Ennek a hátterében az áll, hogy feltételezzük, a $\mu(x,y)$ zaj korrelálatlan, és nulla várható értékű. Tekintsünk N megfigyelést (felvételt a jelenetről):

$$\begin{aligned} g_1(x,y) &= f(x,y) + \mu_1(x,y), \\ g_2(x,y) &= f(x,y) + \mu_2(x,y), \\ &\vdots \\ g_N(x,y) &= f(x,y) + \mu_N(x,y) \end{aligned}$$

Ezek után készítsük el a zajos képek átlagát:

$$\bar{g}(x,y) = \frac{1}{N}\sum_{j=1}^{N} g_i(x,y)$$

Az átlag várható értéke az eredeti kép lesz, azaz $N \to \infty$ esetén

$$E(\bar{(}g)(x,y)) = f(x,y)$$

valamint a szórásnégyzete

$$\sigma^2\bar{g}(x,y) = \frac{1}{N}\sigma^2\mu(x,y)$$

8. fejezet

Box-módszer

Hasonlóan a környezeti átlagoláshoz, itt is egy környezetet (maszkot) kel definiálni, mely leggyakrabban négyzet alakú szokott lenni, azaz $n \times n$ méretű. A módszer lényege, hogy a centrális elem világosságkódját összehasonlítjuk a megadott környezet világosságkódjai által meghatározott intervallummal, majd ha a centrális elem intenzitásértéke beleesik ebbe a tartományba, akkor azt változatlanul hagyjuk, ellenkező esetben más intenzitásértékkel helyettesítjük azt. [8]

Az összehasonlítás előtt érdemes egy rendező eljárást végrehajtani a környezet intenzitásértékein. Amennyiben a centrális elem világosságkódja nem a minimum és a maximum érték által meghatározot intervallumba esik, akkor helyettesítsük azt a tartomány legkisebb, vagy legnagyobb értékével, attól függően, melyikhez van közelebb.

A módszer nagyon hasonlít a medián szűrésre, mivel például nem kapunk eddig nem szereplő intenzitásértékeket. A különbség csupán annyi, hogy nem a „középső" elemet választjuk ki, hanem valamelyik szélső értékkel helyettesítjük a centrális elemet, amennyiben az nincs benne az intervallumban, azaz elüt a környezet árnyalataitól.

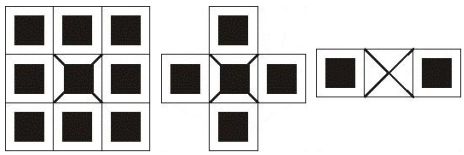
9. fejezet

Matematikai morfológia

Ezt a fajta eljárást a '60-as években fejlesztették ki, mely a geometrián, és alakon alapul; leegyszerűsíti a képet, de az objektum alakját változatlanul hagyja. A morfológiai operátorokat elsősorban a következő területeken szokták használni [12]:

- Képek előfeldolgozására (zajszűrés, alak egyszerűsítés)
- Objektum struktúrájának kiemelésére (skeletonizálás, vékonyítás, vastagítás, objektum megjelölése)
- Objektum kvantitatív leírására (terület, projekciók, Euler-Poincaré karakterisztika)

A matematika morfológia a halmazelméleten alapszik, azaz a képi objektumokat halmazokkal reprezentáljuk. Mi a bináris képet úgy fogjuk tekinteni, mint egészekből álló 2D-s részhalmazokat. Egy pontot két egésszel reprezentálunk, melyek a pixel koordinátáit képviselik; a rács hosszát a koordinátarendszerben pedig a mintavételezés periódusa határozza meg. Azok a pontok, melyek az előtér objektumhoz tartoznak, az X halmazba tartoznak - ezen pontok értékei egy. Azok a pontok, melyek háttérpontok, azaz egy objektumhoz se tartoznak, azok az X komplementereként lesznek nyilvántartva - értékük nulla - : X^C.
Egy morfológiai ϕ transzformációt a kép relációjának (X ponthalmaz), és egy másik, B ponthalmaz megadásával definiálunk, amit struktúra elemnek hívunk. A B ponthalmaz egy kitüntetett ponttal rendelkezik, amit reprezantív pontnak is neveznek, jele: O. A 9.1. ábrán a gyakorlatban leginkább elterjedt struktúra-elemeket látjuk. Amint azt a 9.1(c). ábra is illusztrálja, a reprezentatív pontnak nem feltétlen kell, hogy eleme legyen a B struktúra elemnek.



(a) (b) (c)

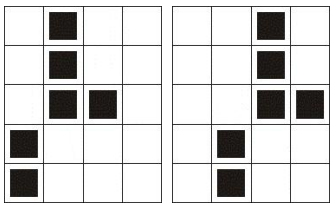
9.1. ábra. A leginkább alkalmazott struktóra-elemek

A morfológiai $\phi(X)$ transzformáció alkalmazása az X képre annyit jelent, hogy a B struktúra-elemet szisztematikusan „átmozgatjuk" az egész képen. A képnek azt a képpontját, ahol a B struktúra O reprezentatív pontja épp tartózkodik aktuális pixelnek nevezzük. Az eredményül kapott reláció lehet akár üres is. A morfológiai művelet dualitása a halmaz komplementerének létezéséből adódik; minden egyes $\phi(X)$ morfológiai transzformációnak létezik a $\phi^*(X)$ duálisa:

$$\phi(X) = (\phi^*(X^C))^C$$

Egy X ponthalmaz h vektorral való eltoltját X_h-val jelöljük, és a következőképpen határozzuk meg [9]:

$$X_h = \{d \in E^2 | d = x + h, \text{ néhány } x \in X \text{ esetén}\}$$



(a) (b)

9.2. ábra. Egy vektorral való eltolás

Érdemes korlátozni a lehetséges morfológiai transzformációk halmazát megszorítások bevezetésével; ezen dolgozatban négy ilyen megszorítást mutatok be. Ezen koncepciók megértése nem egyszerű, de nem szükséges megérteni őket a használatukhoz. A részletesebb leírás, és bizonyítás a Serra által készített könyvben [10] található.
Egy morfológiai transzformációt akkor, és csak akkor hívunk kvantitatívnak, ha teljesíti a következő négy morfológiai alapelvet:

- A fordítással kompatibilis: A ϕ transzformáció függjön az O pozíciójától, és jelöljük ezt a transzformációt ϕ_O-val. Ha az összes pontot eltoljuk $-h$ vektorral, akkor azt a következőképpen jelöljük: ϕ_{-h}. Ekkor a fordítással való kompatibilitás „törvénye" a következőt mondja ki:

$$\phi_O(X_h) = (\phi_{-h}(X))_h$$

Amennyiben a ϕ nem függne az O reprezentatív pont pozíciójától, akkor a „törvény" a következő alakra redukálódna:

$$\phi(X_h) = (\phi(X))_h$$

- Arányváltással való kompatibilitás: Jelöljuk λX-el az X ponthalmaz skalárszorosát (azaz minden pont koordinátáit szorozzuk meg egy pozitív λ kostanssal). Ez egy arányváltásnak felel meg. Jelöljük ϕ_λ-val azt a transzformációt, mely függ a pozitív λ paramétertől. Az arányváltással való kompatibilitást a következőképpen adjuk meg:

$$\phi_\lambda(X) = \lambda\phi(\frac{1}{\lambda}X)$$

Amennyiben a ϕ transzformáció nem függne a λ-tól, az arányváltás a következőre redukálódna:

$$\phi(\lambda X) = \lambda\phi(X)$$

- Lokális tudás: A lokális tudás „törvénye" azt a szituációt veszi figyelembe, hogy egy nagyobb struktúrának mindössze egy kisebb részét tudjuk megvizsgálni - gyakorlatban pedig szinte mindig ez foldul elő a digitális háló méretének korlátozása miatt. Azt mondjuk, hogy a ϕ morfológiai transzformáció rendelkezik a lokális tudásal alapjaival, ha a $\phi(X)$ transzformációban bármely Z' korlátos ponthalmazhoz létezik olyan Z korlátos ponthalmaz, hogy ezekből elő lehessen állítani a ϕ-t. A lokális tudás alapjai a következőképpen írható le szimbólumokkal:

$$(\phi(X \cap Z)) \cap Z' = \phi(X) \cap Z'$$

- Felülről félig folytonos: A felülről félig folytonosság alapja azt mondja ki, hogy a morfológiai transzformáció nem érzékeny a hírtelen változásra. Ennek precíz elmagyarázása sok topológiai fogalmon alapszik, melynek részletes leírása megtalálható a Serra féle irodalomban [10].

A legegszerűbb kvantitatív morfológiai transzformációk a dilatáció, az erózió, a nyitás, valamint a zárás.

9.1. Dilatáció

A dilatáció $\oplus$ egy olyan morfológiai transzformáció, mely két halmazt a vektor összeadás (vagy Minkowski halmaz összeadás) segítségével kombinál. Az $X \oplus B$ dilatáció eredménye egy olyan halmaz, ahol az elemeket az összes lehetséges módon vektoriálisan összeadjuk; egyik elem mindig az X, másik a B halmazból származik.

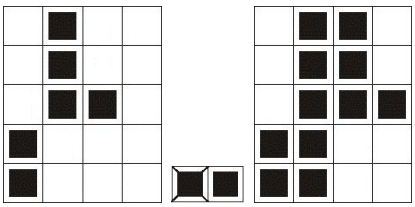
$$X \oplus B = \{d \in E^2 : d = x + b, \forall x \in X, \text{ és } b \in B\}$$

Példák dilatációra [20]:

$$X = \{(0,1), (1,1), (2,1), (2,2), (3,0), (4,0)\}$$

$$B = \{(0,0), (0,1)\}$$

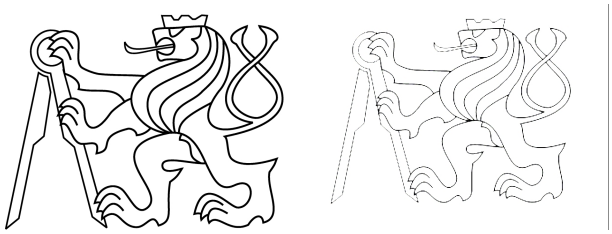
$$X \oplus B = \{(0,1), (1,1), (2,1), (2,2), (3,0), (4,0), (0,2), (1,2), (2,2), (2,3), (3,1), (4,1)\}$$



(a) (b) (c)

9.3. ábra. Dilatáció

Ezt a műveletet sokszor (helytelenül) a két halmaz konvolúciójának szokták nevezni. Cseh egyetem 800×619-es felbontású emblémájának dilatáltja egy 4×4-es struktúrális elem segítségével:



(a) (b)

9.4. ábra. Dilatáció egy képen. Eredeti kép forrása: A Prágai Cseh Technikai egyetem honlapja (http://www.cvut.cz)

MatLab-ban elkészített kódja a következőképpen néz ki [14]:

```
I=imread(...);
B=[ 1 1 1 1; 1 1 1 1; 1 1 1 1; 1 1 1 1 ];
I2=imdilate(I,B);
```

A dilatációt más néven növelésnek, vagy kitöltésnek is szokták nevezni.
A dilatáció rendelkezik néhány érdekes tulajdonsággal, melyek elősegítik az alkalmazás hardveres, és szoftveres implementációját; ezeket bizonyítás nélkül mutatom most be. Ennél részletesebb információ a Serra [10], és a Haralick által készített irodalomban [15] található.
A dilatáció kommutatív

$$X \oplus B = B \oplus X$$

valamint asszociatív

$$X \oplus (B \oplus D) = (X \oplus B) \oplus D$$

A dilatációt úgy is meghatározhatjuk, mint eltolt ponthalmazok unióját

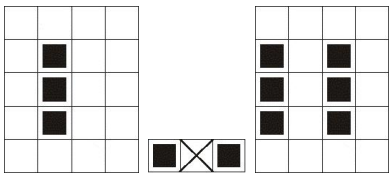
$$X \oplus B = \cup_{b \in B} X_b$$

valamint fordítás invariáns

$$X_h \oplus B = (X \oplus B)_h$$

Az utolsó két egyenlőség azt mutatja, hogy a dilatáció műveletét eltolással fel lehet gyorsítani. Egy *szóban* több pixelt lehet eltárolni (pl. 32-t, egy 32 bites processzorban), valamint ezeket egy utasításban el lehet tolni, vagy összeadni. Valamint az eltolást a pipeline párhuzamos processzorokon könnyen lehet implementálni, mint egyfajta késleltetést. A dilatáció egy *növekményes* transzformáció:

$$\text{Ha } X \subseteq Y, \text{ akkor } X \oplus B \subseteq Y \oplus B$$



9.5. ábra. Dilatáció, ahol a reprezentatív pont nem eleme a struktúra elemnek

Amint az a képen is látható, ha reprezentatív pont nem eleme a B struktúraelemnek, akkor a végeredmény eléggé különbözhet a bemeneti halmaztól. A dilatációt kis lyukak, és "öblök" eltüntetésére használják az objektumon. A dilatáció megnöveli az objektum méretét - amennyiben az eredeti méretet kell megtartani, a dilatációt erózióval kell kombinálni, amit a következő részben írok le részletesen.

9.2. Erózió

Az erózió $\ominus$ a két halmazt úgy kombinálja, hogy elemeit vektoriálisan kivonja egymásból; és a dilatáció duálisa. Sem az erózió, sem a dilatáció nem invertálható transzformáció

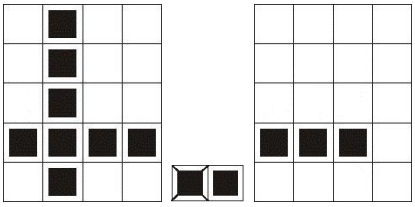
$$X \ominus B = \{d \in E^2 : d + b \in X, \forall b \in B\text{-re}\}$$

Ez a formula azt mondja ki, hogy az X halmaz (kép) minden d pontját megvizsgáljuk, és az eredmény azon d pontok lesznek, melyekre igaz, hogy a $d + b$ is bennevan az X-ben. Példák erózióra:

$$X = \{(0,1),(1,0),(2,1),(3,0),(3,1),(3,2),(3,3),(4,1)\}$$

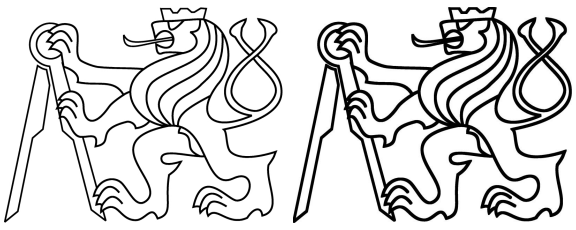
$$B = \{(0,0),(0,1)\}$$

$$X \ominus B = \{(3,0),(3,1),(3,2)\}$$



(a) (b) (c)

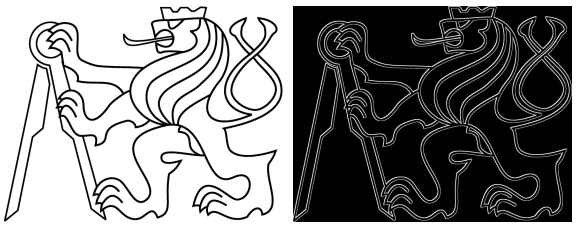
9.6. ábra. Erózió egy két elemű struktúra elemmel



(a) (b)

9.7. ábra. Az erózió, mint izotrópikus zsugorító eljárás

Érdemes megfigyelni, hogy a pár pixel széles vonalak eltűnnek. Az izotrópikis struktúraelemmel történő eróziót redukálásnak, vagy zsugorításnak nevezzük.
Az alap morfológiai transzformációkat az objektumok kontúrjának nagyon gyors megtalálására is szokták használni. Ezt úgy érik el, hogy ez eredeti képből kivonják annak erodáltját.



(a) (b)

9.8. ábra. Az objektum kontúrját úgy kapjuk meg, hogy ez eredeti képből kivonjuk az erodáltját (jobbra)

Az eróziót a kép egyzserűsítésére is szokták alkalmazni, hiszen ha az objektum, vagy bármely része egy pixel vastag, azt a részt eltünteti. Így a bonyolultabb objemtumot több kisebb, egyszerűbb darabra lehet felbontani. Létezik az eredeti erózió definicióval egy ekvivalens megfogalmazás, melynél talán könnyebben megérthetők a tulajdonságai:

$$X \ominus B = \{d \in E^2 : B_d \subseteq X\}$$

Ilyenkor a B struktúraelem úgy van interpretálva, mint egy mozgó ablak, mely az X képen végigmozog; ha a B elem d vektorral való eltoltja benne van az X képben, akkor a B reprezentatív pontja az $X \ominus B$ erózió eredményéhez fog tartozni.
Az erózió implementációját kissé egyzerűsíteni lehet, azaz az X kép B struktúraelemmel való erodáltját úgy kapjuk meg, hogy az X képet eltoljuk a $-b \in B$ vektorral [1], majd vesszük ezek metszetét.

$$X \ominus B = \cap_{b \in B} X_{-b}$$

Ha a reprezentatív pont nem eleme a struktúra elemnek, akkor az erózió egy kiterjedés mentes transzformáció; azaz ha $(0,0) \in B$, akkor $X \ominus B \subseteq X$. Az erózió szintén eltolás invariáns

$$X_h \ominus B = (X \ominus B)_h$$

$$X \ominus B_h = (X \ominus B)_{-h}$$

és mint a dilatáció, ez is növekményes transzformáció

$$\text{Ha } X \subseteq Y, \text{ akkor } X \ominus B \subseteq Y \ominus B$$

[1] Ez a fajta erózió különbözik attól, amit Serra megfogalmazott [10]. Ott a $\ominus$ a Minkowski kivonást jelöli, ami egy eltolást jelöl minden irányban $b \in B$ vektorral, és ezeknek veszik a metszetét. A mi esetünkben egy mínusz jel került a vektor elé.

Hogyha mind B, mind pedig D struktúra elemek, és a B tartalmazza a D-t, akkor azt mondjuk, hogy a B agresszívabb, mint a D; azaz ha $D \subseteq B$, akkor $X \ominus B \subseteq X \ominus D$. Ez a tulajdonság egyfajta rendezést eredményez az azonos „formájú", de különböző méretű eróziók között.
Jelölje $\hat{B}$ a B szimmetrikus halmazát (más irodalomban a szimmetrikus halmaz helyett felcserélő, vagy racionális halmaz kifejezést használhatnak), legyen ennek egy O reprezentatív pontja. Ekkor

$$\hat{B} = \{-b : b \in B\}$$

Például ha

$$B = \{(1,2),(2,3)\}, \text{ akkor}$$
$$\hat{B} = \{(-1,-2),(-2,-3)\}$$

Mint már említettem, az erózió, és a dilatáció egymás duálisai. Ez formálisan azt jelenti, hogy

$$(X \ominus Y)^C = X^C \oplus \hat{Y}$$

Az erózió, és a dilatáció közötti különbségekre a következő tulajdonságok világítanak rá. Az erózió (ellentétben a dilatációval) nem kommutatív

$$X \ominus B \neq B \ominus X$$

Ha kombináljuk az eróziót, és a képek metszetét, a következőkhöz jutunk

$$(X \cap Y) \ominus B = (X \ominus B) \cap (Y \ominus B)$$
$$B \ominus (X \cap Y) \supseteq (B \ominus X) \cup (B \ominus Y)$$

Másrészt a dilatációt, és a képek metszetét nem lehet felcserélni, ugyanis két kép metszetének a dilatáltja valódi részhalmaza lesz a dilatációk metszetének

$$(X \cap Y) \oplus B \subseteq (X \oplus B) \cap (Y \oplus B)$$
$$B \oplus (X \cap Y) \subseteq (X \oplus B) \cap (Y \oplus B)$$

Az erózió rendezése (illetve a dilatációé is) kicserélhető a halmazelméleti unióval. Ennek eredményeként a struktúra elemet szét lehet bontani több, egyszerűbb struktúra elem uniójára

$$B \oplus (X \cup Y) = (X \cup Y) \oplus B = (X \oplus B) \cup (Y \oplus B)$$
$$(X \cup Y) \ominus B \supseteq (X \ominus B) \cup (Y \ominus B)$$
$$B \ominus (X \cup Y) = (X \ominus B) \cup (Y \ominus B)$$

Az X képen a B, majd D struktúrális elemekkel való egymást után végrehajtott dilatáció (illetve erózió is) ekvivalens azzal, mintha az X képet a $B \oplus D$-vel dilatálnánk (erodálnánk)

$$(X \oplus B) \oplus D = X \oplus (B \oplus D)$$
$$(X \ominus B) \ominus D = X \ominus (B \ominus D)$$

MatLab-ban az eróziót az *imerode* függvény használatával a legegyszerűbb elérni.

9.3. Nyitás, és zárás

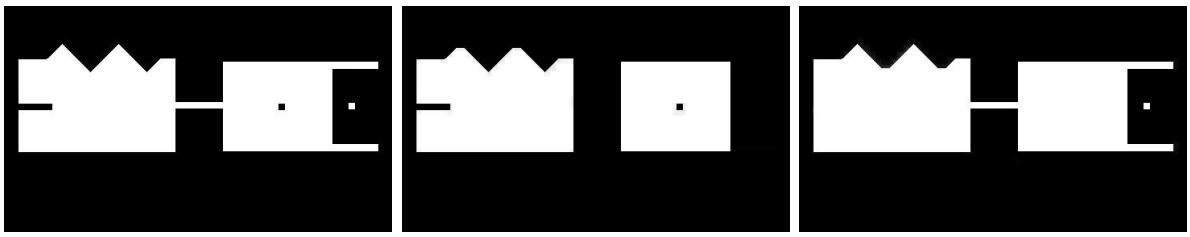
Az erózió, és dilatáció nem invertálható transzformációk, azaz ha egy képet erodálunk, majd dilatálunk, akkor nem a kiindulási képet fogjuk visszakapni. Az eredményül kapott kép „egyszerűbb", és kevésbé részletes lesz, mint az eredeti.
Ha egy képen az eróziót követően végrehajtunk egy dilatációt, akkor egy nagyon fontos morfológiai transzformációt kapunk, amit *nyitásnak* nevezünk. Az X képen a B struktúra elemmel végrehajtott nyitást $X \circ B$-vel jelöljük, és a következőképpen definiáljuk:

$$X \circ B = (X \ominus B) \oplus B$$

A morfológiai nyitás olyan egész régiókat tüntet el az objektumból, melyekbe „nem fér bele" a struktúra elem. Az objektum határán lévő kinövéseket letörli, megszakítja a keskeny szálakat.
Amennyiben viszont a dilatációt erózió követi, azt a módszert *zárásnak* nevezzük. Az X képen a B struktúra elemmel végrehajtott zárás $X \bullet B$-vel jelöljük, és a következőképpen definiáljuk:

$$X \bullet B = (X \oplus B) \ominus B$$

Ha az X kép nem változik a B struktúra elemmel való nyitás után, akkor azt úgy nevezik, hogy nyitás B-re vonatkozóan. Hasolóképpen, ha az X kép nem változik a B struktúraelemmel való zárás esetén, azt úgy nevezzük, hogy zárás B-re vonatkozóan.
Az izotrópikus struktúraelemmel végrahajtott nyitás, és zárás arra jó, hogy eltüntessük a struktúreelemnél kisebb képrészleteket, miközben az objektum globális alakját nem torzítjuk el. A zárás összeköti azokat az obejktumokat, amelyek elég közel vannak egymáshoz, eltünteti, kitölti a kis lyukakat, és kisimítja az objektum körvonalát úgy, hogy eltünteti a kis „öblöket". A „közeli", kicsi", és „kis lyukak" fogalmak a struktúraelem méretétől, és alakjától függnek.



(a) (b) (c)

9.9. ábra. Az eredeti kép, valamint az azon végrehajtott nyitás, és zárás

A 9.9. ábrán látható 482×200 felbontású képet egy 10×10-es struktúrális elemmel lett erodálva, valamint dilatálva. Ennek MatLab kódja a következőképpen néz ki [14]:

```
i=imread(...);
j=[10 10];
struct_elem=strel('rectangle',j);
eredmeny=imopen(i,struct_elem);
```

A stuktúrális elemet a *strel* függvény segítségével alakítottam ki, melynek alakja egy négyszög. A zárás forráskódja hasonlóan néz ki ehhez, csak *imclose* függvényt kell benne használni.

Ellentétben az erózióval, és a dilatációval, a nyitás és zárás invariáns a struktura elem eltolására. Az erózió, és dilatáció tulajdonságaiból következik, hogy a nyitás, és a zárás egyaránt növekményes transzformációk. A nyitás kiterjedés mentes ($X \circ B \subseteq X$), míg a zárás kiterjedéses transzformáció ($X \subseteq X \bullet B$).
Akárcsak a dilatáció, és erózió, a nyitás és zárás is duális transzformációk

$$(X \bullet B)^C = X^C \circ \hat{B}$$

Másik lényeges különbség, hogy az egymás után, iteratív módon végrehajtott nyitás, és zárás idempotens, azaz a transzformáció újbóli alkalmazása nem változtatja meg az előzőekben megkapott eredményt. Formálisan:

$$X \circ B = (X \circ B) \circ B$$

$$X \bullet B = (X \bullet B) \bullet B$$

(a) (b) (c)

9.10. ábra. Egy zajos újlenyomat, az eredeti képen végrehajtott nyitás, majd az ezen végrehajtott zárás eredménye. Az eredeti kép forrása: http://nalts.wordpress.com

10. fejezet

Hasonlóság alapú impulzív zaj eltüntetése színes képeken

Az algoritmus a paraméter nélküli fajsúly becslésen alapszik. Ennek az új szűrőtechnikának az az elméleti alapja, hogy maximalizálja a pixelek közötti hasonlóságot egy előre definiált szűrőablakban. Ez az új eljárás gyorsabb, mint a standard vektor median szűrés (VMF), valamint annál élesebb, finomabb képet produkál.

10.1. A zaj kiszűrése színes képekről

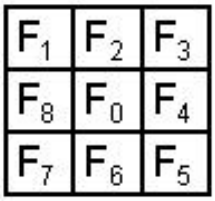
A legnépszerűbb nemlineáris, multicsatornás szűrők azon alapszanak, hogy egy előre megadott mozgó ablakban valamilyen sorrendbe rendezik a vektorokat. [5]
Jelölje $F(x)$ a multicsatornás képet, legyen W az ablak, melynke mérete: $n+1$ (szűrő hossza). A W ablakon belül a zajos képvektorokat jelöljük F_j-vel, ahol $j = 0, 1, \ldots, n$. Ha az F_i és F_j vektorok közötti különbséget $\rho(F_i, F_j)$-vel jelöljük, akkor az $R_i = \sum_{j=0}^{n} \rho(F_i, F_j)$ távolság az F_i-t jellemzi. Ha rendezzük az R_i értékeket: $R_{(0)} \leq R_{(1)} \leq \cdots \leq R_{(n)}$ az maga után vonja a hozzájuk tartozó F_i vektorok rendezését: $F_{(0)} \leq F_{(1)} \leq \cdots \leq F_{(n)}$. Bár mindez egyszerűnek tűnik, mégsem annyira könnyű megvalósítani, hiszen többcsatornás adatokon kell rendezést végrehajtani, és nem létezik univerzális rendező algoritmus a vektortérben. Hogy ezt a problémát áthidalják, a vektorok hosszát veszik a vektorrendezés alapjául.

A standard szűrőknek az az alapja, hogy megkeressék, majd kicseréljék a zajos pixeleket. De azon fontos tulajdonságot, hogy a nem zajos pixeleket ne módosítsák; sajnos közel sem tudják ideálisan megvalósítani. Ám a most ismertetésre kerülő algoritmus elég gyors, valamint a nem zajos pixelek értékeit nem változtja meg.

10.2. Az algoritmus

10.2.1. Szürke skálás képeken

Legyen W egy szűrőablak, $n+1$ pixel méretű, $\{F_0, F_1, \ldots, F_n\}$, ahol F_0 jelöli a centrális elemet, és n a centrális elem szomszédjáinak a száma.



10.1. ábra. Az F_0 centrális elem, és annak nyolc-szomszédsága

Valamint definiáljunk egy hasonlósági függvényt: $\mu[0,\infty] \to \mathbf{R}$, mely nem növekvő $[0,\infty]$-ban, konvex a $[0,\infty]$-ban, valamint kielégíti a $\mu(0) = 1$, és $\mu(\infty) = 0$-t. Két azonos intenzitású pixel hasonlóságának 1-nek kell lennie; két, a szürke skálán egymástól távol elhelyezkedő pixel hasonlóságának 0-hoz kell közelítenie. A $\mu(F_i, F_j)$ függvényt definiáljuk a következőképpen: $\mu(F_i, F_j) = \mu(|F_i - F_j|)$, és kielégíti a fennt említett három tulajdonságot.
Definiáljuk M-t úgy, hogy M-be tároljuk az F_k, és szomszédai hasonlóságának összegét. Azaz a centrális elemre, valamint a szomszédaira vonatkozó M_0, illetve M_k :

$$M_0 = \sum_{j=1}^{n} \mu(F_0, F_j), \qquad M_k = \sum_{j=1, j\neq k}^{n} \mu(F_k, F_j)$$

Megfigyelhetjük, hogy F_k-t csak az F_0 szomszédaival hasonlítjuk össze, de F_0-al nem; ez az alap elgondolása az algoritmusnak. Az M_0-ban n darab összehasonlítást kell végezni: $\mu(F_0, F_k)$, $K = 1, 2, \ldots, n$; M_k-ban viszont $n-1$ összehasonlítást kell elvégezni ($k > 0$), mivel az M_k-ból kihagyjuk az F_0 összehasonlítását.

Ebben a konstrukcióban, a W ablakban az F_0 referencia pixelre ha fennáll az $M_0 < M_k$ eset valamely $k = 1, 2, \ldots, n$-ra, akkor cseréljük ki a centrális elemet az egyik szomszédjára. Méghozzá arra az F_k-ra cseréljük az F_0-t, ahol $k = \arg\max M_i$, $i = 1, \ldots, n$. Más szóval, ha fennáll az $M_0 < M_k$ egyenlőtlenség ($k = 1, \ldots, n$), akkor az F_0-t hibás pixelnek érzékeljük, amit ki kell cserélni az egyik F_i-re, ahol az M-ekben tárolt hasonlóság összegek maximuma van. Ezen hasonlóságok összegek a szomszédok között vannak vizsgálva, és ebbe nem számoljuk a centrális elemre vonatkozó hasonlóságot.

Összefoglalva az új pixelt a W ablakból kell kiválasztani (ellentétben például a VMF szűrővel). Ezért kell a μ-nek konvexnek lennie, azaz hogyha az M hasonlósági funkciók összegei közül a maximumot akarom megkeresni, akkor elegendő az M értékeit az $F_0, \ldots, F_n$ pontokban meghatározni.

10.2.2. Színes képeken

A bemutatott eljárás kisebb módosítások után átültethető színes képekre. Itt definiáljuk a hasonlósági funkciót a következő képpen: $\mu\{F_i, F_j\} = \mu(||F_i - F_j||)$, ahol $||.||$ a vektor normáját jelöli. Ezek után ugyanúgy maximalizálni kell az összes M hasonlósági függvényt, ahogyan azt az előbb csináltuk. Más, konvex szűrőkkel összehasonlítva ezt az eljárást, bíztató eredményeket kapunk, ha a következő hasonlósági funkciók valamelyikét használjuk, amit a paraméter nélküli fajsúly becslés magjának nevezünk:

$$\mu_0 = exp\left\{-\left(\frac{x}{h}\right)^2\right\}, \quad \mu_1 = exp\left\{-\frac{x}{h}\right\}, \quad \mu_2 = \frac{1}{1+x/h}, \quad \mu_3 = \frac{1}{(1+x)^h},$$

$$\mu_4 = 1 - \frac{2}{\pi} arctan\left(\frac{x}{h}\right), \quad \mu_5 = \frac{2}{1+exp\left\{\frac{x}{h}\right\}}, \quad \mu_6 = \frac{1}{1+x^h}, \quad \mu_7 = \begin{cases} 1 - x/h, \text{ ha } x \le h \\ 0, \text{ ha } x > h \end{cases}$$

Érdemes megemlíteni, hogy a legjobb eredményt ezek közül az egyik legegyszerűbb, a $\mu_7(x)$ hasonlósági függvény produkálta, ami így nagyon gyorsan képes zajmentes képet produkálni. Tehát többcsatornás esetben:

$$M_0 = \sum_{j=1}^{n} \mu(F_0, F_j), \quad M_k = \sum_{j=1, j\neq k}^{n} \mu(F_k, F_j), \text{ (2)}$$

ahol $\rho\{F_i, F_k\} = (||F_k - F_l||)$, és $||.||$ jelöli a vektor L_2 normáját.
Alkalmazva a kapottakat a μ_7 hasonlósági függvénybe a következőt kapjuk:

$$\mu_7(F_i, F_k) = \begin{cases} 1 - \rho(F_i, F_k)/h, \text{ ha } \rho(F_i, F_k) < h, \\ 0 \text{ egyébként} \end{cases}$$

Ezek után a (2)-es egyenlőségekbe behelyettesítve a következő egyenlelőségekhez jutunk:
$M_0 = n - \frac{1}{h}\sum_{j=1}^{n} \rho(F_0, F_j)$, és $M_k = \sum_{j=1, j\neq k}^{n} \left(1 - \frac{\rho(F_k, F_j)}{h}\right) = n - 1 - \frac{1}{h}\sum_{j=1}^{n} \rho(F_k, F_j)$.
Ezt felhasználva megkaphatjuk az M_0 és M_k különbségét:

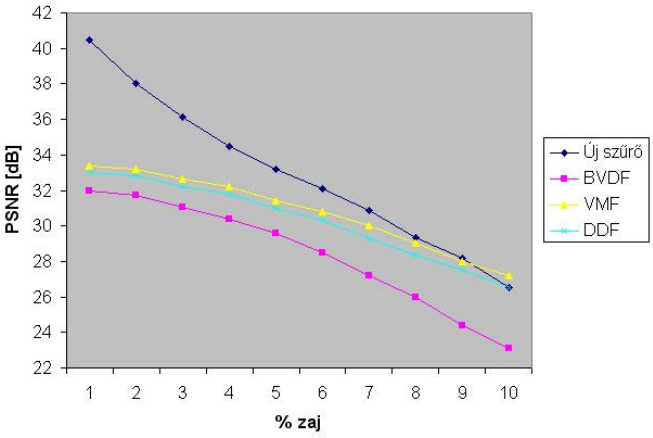
$$M_0 - M_k = 1 - \frac{1}{h}\sum_{j=1}^{n} [\rho(F_0, F_j) - \rho(F_k, F_j)], \text{ (3)}$$

$$M_0 - M_k > 0, \text{ ha } h > \sum_{j=1}^{n} [\rho(F_0, F_j) - \rho(F_K, F_j)]. \text{ (4)}$$

Ha ez utóbbi egyenlőtlenség (4) teljesül, akkor a centrális elemet zajmentes pixelnek tekintjük; ellenkező esetben viszont arra az F_i pixelre cseréljük ki, ahol az összegzett hasonlósági értékek maximuma van. Ebben az esetben viszont az eljárás csak akkor cseréli ki a centrális elemet, ha az valóban zajos, különben megőrzi az eredeti kép struktúráját.
A h paraméter értékét vagy tapasztalatok, kísérletezgetések alapján, vagy adaptív módon határozzuk meg, aminek részletes leírását az *A. Smolka - On the reduction of impulsive noise in multichannel image processing* [6] könyvében olvashatjuk.

10.3. Eredmények, és konklúziók

A most bemutatott eljárás a Vektor Medián Filter egy továbbfejlesztett változatána tekinthető. Összehasonlítva más szűrő eljárásokkal kiderül, hogy ez az új eljárás felülmúlja a hagyományos szűrőket teljesítményben, amikor az impulzív zaj eltüntetése a cél. Ez az új szűrőtechnika, ami a hasonlósági függvényen, illetve a fajsúly becslésen alapszik sokkal gyorsabb, mint a VMF szűrő, ami olyan alkalmazásokba való beépítését teszi lehetővé, ahol igenis sokat számít a sebesség.



(a)

Eljárás	NMSE[10^{-4}]	RMSE	PSNR[dB]
AMF	82,86	12,9	25,91
VMF	23,3	6,84	31,42
ANNF	31,27	7,92	30,14
BVDF	29,07	7,64	30,46
HDF	22,84	6,77	31,51
AHDF	22,6	6,73	31,55
DDF	24	6,94	31,28
FVDF	26,75	7,33	30,82
Szűrők magjai			
$\mu_0(x)$	5,05	3,16	38,17
$\mu_1(x)$	4,95	3,15	38,14
$\mu_2(x)$	5,39	3,29	37,77
$\mu_3(x)$	9,57	4,38	35,28
$\mu_4(x)$	5,06	3,19	38,05
$\mu_5(x)$	4,77	3,09	38,3
$\mu_6(x)$	11,02	4,7	34,67
$\mu_7(x)$	4,69	3,07	38,38

(b)

10.2. ábra. Az (a) ábrán az új algoritmus hatékonyságát látjuk összehasonlítva a többi szűrő hatékonyságával, a PSNR függvényében, ha a LENA képet 1% - 10% zajjal terheljük minden csatornán. A (b) ábrán a különböző kernel függvényeket láthatjuk összahasonlítva a standard szűrő algoritmusokkal, ha a szűrést a LENA képen végezzük 5% zajjal terhelve.

11. fejezet

Szűrés frekvencia tartományban

A frekvenciatartománybeli szűrés alapját a konvolúciós tétel adja. Legyen $g(x, y)$ az eredmény kép, $f(x, y)$ az eredeti kép és a $h(x, y)$ pedig a szűrő függvény. Ekkor

$$g(x, y) = h(x, y) * f(x, y)$$

A frekvenciatérbeli szűrés alapját a konvolúciós tétel adja [9]:

$$G(u, v) = H(u, v) \cdot F(u, v)$$

ahol a G, H és F a g, h, f függvények Fourier transzformáltját jelölik. A H függvégyt szűrő függvénynek nevezzük. A képtartománybeli konvolúcióval szemben a frekvenciatartományban mindössze egy szorzást kell elvégezni, ami jóval kevésbé műveletigényes, ezért gyorsabb is. A gyors-Fourier transzformáció használatával pedig sokkal hatékonyabb, mint a képtartománybeli szűrés.

11.1. Aluláteresztő szűrő

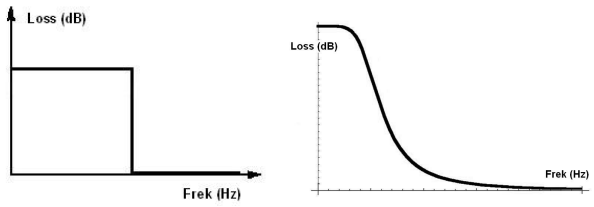
A zajok, és a világosságkódban mutatkozó éles eltérések a kép Fourier-transzformáltjának magas frekvenciás összetevőiben jelentkezik. Ebből következik, hogy ha a frekvenciatartománybeli magas frekvenciákat gyengítjük (vagy teljesen kiszűrjük), akkor „megszabadulunk" a képen az éles intenzitás-átmenetektől (zajoktól); viszont így a képen egy erős elmosó hatást érünk el, az élek is elmosódnak, elhomályosodnak. [8]
A $G(u, v) = F(u, v) \cdot H(u, v)$ összefüggésből következik, hogy a H függvényt úgy kell megválasztani, hogy az f kép F Fourier trnaszformáljának magas frekvenciás részeit szűrja ki, és az alacsonyabb frekvenciás részeit hagyja változatlanul. Az ideális aluláteresztő függvény (ILPF) átviteli függvénye a következőképpen néz ki:

$$H(u, v) = \begin{cases} 1, \text{ ha } \delta(u, v) \leq \delta_0, \\ 0, \text{ egyébként} \end{cases}$$

ahol δ a távolságfüggvény, δ_0 pedig egy küszöb. A definícióból jól látszik, hogy a szűrő változatlanul hagyja a δ_0 küszöbnél alacsonyabb frekvenciás összetevőket, míg a magasabb frekvenciájúkat teljesen kiszűri. A δ_0-t vágási frekvenciának nevezzük. Mivel ez a szűrőmodell nem csak a zajokat, hanem - mint említettem - a teljes magas frekvenciás részt kivágja, a kép veszíteni fog élességéből. Ezen felül egy olyan hatása is lesz az eljárásnak (ring effect - gyűrű hatás), hogy periódikusan ismétlődő foltok jelennek meg a képen. Ennek kiküszöbölésére találták ki pl. a Butterworth-szűrőt (BLPF), mely a következőképpen adható meg:

$$H(u,v) = \frac{1}{1 + (\frac{\delta(u,v)}{\delta_0})^{2n}}$$

ahol δ_0 a vágási frekvencia, $2n$ pedig az átmenet sebességét szabályozó pozitív paraméter.



(a) (b)

11.1. ábra. Az ideális -, valamint a Butterworth aluláteresztő szűrő karakterisztikája

11.2. Felüláteresztő szűrő

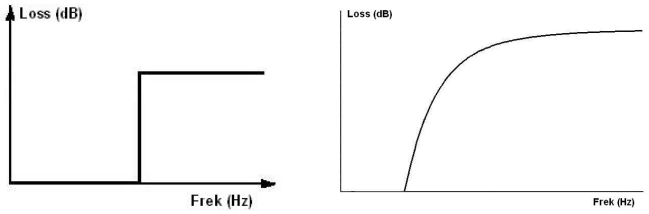
A torzítások jelenkezhetnek úgy is a képen, hogy az élek elmosódnak, a határátmenetek kiszélesednek. A képélesítéshez tehát egy olyan H függvényre van szükség, ami gyengíti, vagy teljesen eltünteti az alacsony frekvenciás részeket a kép Fourier transzformáltjában. A legyegyszerűbb ilyen szűrő az idális felüláteresztő szűrő (IHPF), mely a következő matematikai modellel adható meg: [8]

$$H(u,v) = \begin{cases} 0, \text{ ha } \delta(u,v) \leq \delta_0, \\ 1, \text{ egyébként} \end{cases}$$

Akárcsak az aluláteresztő szűrönek, ennek a módszernak is van alternatív változata, mely nem annyira élesen/hírtelen szünteti meg a frekvenciasávokat. A Butterworth felüláteresztő szűrő (BHPF) a következőképpen definiálható:

$$H(u,v) = \frac{1}{1 + (\frac{\delta_0}{\delta(u,v)})^{2n}}$$

ahol δ_0 a vágási küszön, $2n$ pedig az átmenet sebességét szabályozó pozitív paraméter.



(a) (b)

11.2. ábra. Az ideális -, valamint a Butterworth felüláteresztő szűrő karakterisztikája

11.3. Sávszűrő

Az alul-, és felüláteresztő szűrőt kombinálva könnyen készíthetünk speciális célú sávszűrőket, melyek csupán egy adott frekvenciasávot engednek át, vagy erősítenek. Így lehetőségünk adódik a kép éleinek feldolgozására azok erősségük alapján. Egy ilyen ideális sávkiemelő függvény matematikai modellje: [8], [3]

$$H(u,v) = \begin{cases} 1, \text{ ha } \delta_0 \leq \delta(u,v) \leq \delta_1, \\ 0 \text{egyébként} \end{cases}$$

ahol δ_0, és δ_1 a vágáshoz szükséges alsó-, illetve felső nemnegatív küszöbérték. Mivel ideális esetben a karakterisztika igen meredek, ezért hasonló mellékhatás keletkezhet, mint az ideális alul-, illetve felüláteresztő szűrőknél. Érdemes ezért mondjuk két Butterworth szűrőt alkalmazni, melyeknek nem annyira meredek a karakterisztikájuk.

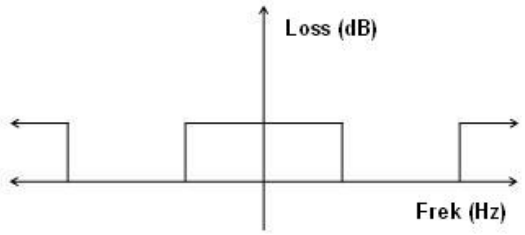
11.4. Notch filter

A jelfeldolgozás területén a sávzáró, vagy sávkivágó függvényt olyan alklalmazásokban szokták használni, amikor egy bizonyos frekvenciatartományt egy nagyon alacsony szintre kell gyengíteni, esetleg teljesen meg is kell szüntetni; a többi frekvenciatartományt pedig változatlanul kell hagyni. A notch filter (ideális esetben) egy meredek vágási karakterisztikával (és magas Q faktorral) rendelkező sávzáró függvény. [13]

Elsősorban az élő hangzások megteremtésében (PA rendszereknél), különböző hangerősítőknél (elsősorban a különböző az akusztikus hangeszközök pl. gitár, mandolin, stb. elő-, és közép - erősítőiben), valamint a jelfeldolgozás speciális területein szokták alkalmazni, hogy lecsökkentsék, vagy meggátolják a visszacsatolásokat, miközben a többi frekvenciaspektumra legfeljebbnagyon kicsi észrevehető hatást gyakorolnak.

A következőképpen szokták még nevezni ezt a fajta szűrőt: 'sávlimitáló szűrő', 'T-noch szűrő', 'sáv elimináló szűrő', 'sávkivágó szűrő'.

Mivel a karakterisztikája meredek, elég kis frekvenciatartományt is lehet vele szűrni. A vágás standard referenciapontjának azokat a pontokat szoktuk megnevezni a görbén, ahol a jel 3dB-el, azaz az eredeti jel 70.7%-ára csökken. [20]



(a)

11.3. ábra. A notch szűrő karakterisztikája

12. fejezet

Képrekonstrukciók

A képrekonstrukciók figyelembe veszik a kép torzulásának módját, melynek függvényében megbecsüli az eredeti képet. A becslés nagymértékben függ a torzulás becslésének ismeretétől. A legtöbb rekonstruáló alkalmazás a konvolúción alapszik. [8]

A képek torzulását sok minden előidézheti: tökéletlen optikai lencsék, nem lineáris elektro-optikai szenzorok, szemcsés film alapanyag, rossz fókusz, légáramlat, stb.

A képrekonstrukciós eljárások két fő csoportba sorolhatóak: determinisztukusak, és sztochasztikusak. [12] A determinisztikus eljárások akkor használhatóak, ha nagyon kevés a képen a zaj, és ismert a „torzító függvény". Ekkor az eredeti képet úgy kapjuk meg, hogy a torzult képre alkalmazzuk a torzító inverz függvény. A sztochasztikus technikák valamilyen sztochasztikus kritérium alapján próbálják megtalálni a legjobb visszaállító technikát. A legtöbb esetben az első feladat a torzító transzformáció megtalálása.

Az explicit módon meghatározható torzító függvény mindig előnyös. Minél pontosabb tudunk a torzulásról, annál jobb eredményt érhetünk el. A legygyakoribb, és legegyszerűbb torzító függvények a következőek: a kamerához képest a tárgy relatív állandó sebességgel mozog, rossz lencsefókus, atmoszférikus turbulencia.

A legtöbb esetben sajnos elégtelen tudással rendelkezunk a torzulásról, ezért becsülnünk, és modelleznünk kell. A rendelkezésre álló információk alapján a becslések két csoportba sorolhatóak: a-priori, és posteriori.

Az a-priori tudásnál a torzulásról előzetesen tudunk, vagy a rekonstrukció előtt szerzünk róla tudmást. Ha tudjuk, hogy az objektum a szenzorhoz képest mozgást végzett, akkor a modellezés csak a mozgás irányát, és sebességét tudja meghatározni. Egy másik példa a TV kamerák, és különböző digitalizálók paramétereinek megbecslése, torzulásukat pedig úgy modellezhetjük, hogy ismerünk egy mintaképet, és az általa felvett torzított mását.

Posteriori tudáshoz a torzított kép analízise során jutunk. Erre egy tipikuspélda, ha bizonyos pontokra vagyunk kíváncsiak (pl sarkok, egyenes vonalak), hogyan is nézhettek ki a torzulás előtt. Egy másik módszer lehet a kép régióinak spektrális karakterisztikájának használata, amik viszonylag homogének.

A g torzított képet az f eredeti képből a következőképpen kapjuk meg

$$g(i,j) = s\left(\int\int_{(a,b)\in O} f(a,b)h(a,b,i,j)\ da\ db\right) + \nu(i,j)$$

ahol ν a zajt, s pedig egy nem lineáris függvényt ír le. Sokszor szokták ezt a képletet oly módon egyszerűsíteni, hogy elhagyják a képletből a nemlíneáris tulajdonságot, és feltételezik, hogy a h invariáns a kép pozíciójára. A torzítást a konvolúció segítségével is ki lehet fejezni

$$g(i,j) = (f * h)(i,j) + \nu(i,j)$$

Amennyiben a zaj elhanyagolhatóan kicsi, akkor a képrekonstrukció csupán az inverz konvolúcióból áll. Ebben az esetben az inverz konvolúciót lineáris egyenletrendszerként kell megoldani.

12.1. Torzulások, melyeket egyszerű helyreállítani

Mint azt korábban melítettem, három, matematikailag egyszerűen leírható, könnyen visszaállítható torzulás létezik. Ezeket konvolúcóval, egyenlettel lehet meghatározni.

12.1.1. A kamerához képest való elmozdulás

Tételezzük fel, hogy a kép egy olyan készülékkel lett elkészítve, mely mechanikus zárral rendelkezik. Ha az objektum a T záridő alatt elmozdul a kamerához képest a felvétel során, akkor az objektum a képen el fog mosódni. Ha az objektum állandó V sebességgel halaz az x tengely irányában, akkor a $H(u,v)$ torzulás Fourier transzformációját a T idő alatt a következő képlet adja meg [12]

$$H(u,v) = \frac{sin(\pi VTu)}{\pi Vu}$$

12.1.2. Rossz lencse-fókusz

Ha a vékony lencse tökletlen fókuszából származik a kép elmosódása, akkor azt a következő matematikai egyenlettel lehet modellezni [12]

$$H(u,v) = \frac{J_1(ar)}{ar}$$

ahol J_1 az első rend Vessel függvénye, $r^2 = u^2 + v^2$, és a az elmozdulás.

12.1.3. Atmoszférikus turbulencia

Az atmoszférikus turbulencia által keltett torzulást az asztronómiánál, és távérzékelésnél szokták javítani. Ezt az atmoszférában található nem homogén hőmérséklet okozza, mely elhajlítja a fénysugarakat. Ennek matematikai modelljét a következő egyenlet írja le [4]

$$H(u,v) = e^{-c\left(u^2+v^2\right)^{\frac{5}{6}}}$$

ahol c egy konstans, értéke függ a turbulencia típusától, és általában tapasztalati úton állítunk be. Az $\frac{5}{6}$-os kitevőt néha 1-re kell cserélni. [12]

12.2. Inverz szűrő

Az inverz szűrő azon alapszik, hogy a torzulást a lineáris $h(i,j)$ függvény okozta, valamint a additív ν zaj is szerepet játszik a torzulásban. Továbbá tételezzük fel, hogy a ν zaj független a jeltől. A Fourier transzformáció alkalmazása után a következő egyenlőséghez jutunk

$$G(u,v) = F(u,v)H(u,v) + N(u,v)$$

A torzulás eliminálható, ha a rekonstruáló szűrőnek létezik olyan áthelyező függvénye, mely inverze a h torzulásnak. Akkor az inverz szűrő Fourier transzformáltját a $H^{-1}(u,v)$-ként határozzuk meg. A torzítatlan F képet (Fourier transzformáció után) a torzított G képből, a következő egyenlet segítségével határozzuk meg

$$F(u,v) = G(u,v)H^{-1}(u,v) - N(u,v)H^{-1}(u,v)$$

Ez az egyenlet arra mutat rá, hogy az inverz szűrő nagyon jól működik azokon a képeken, amelyek zajmentesek. Amennyiben zaj is van a képen, és additív hiba keletkezik, akkor az befolyásolja a frekvenciákat, ahol a $H(u,v)$ alacsony nagyságrendel rendelkezik. Általában a frekvencia térbeli magas frekvenciában jelentkezik, így a képen a finom részleteket elmossa. A zaj szintjének megváltoztatása szintén problémákat okozhat, mivel a kis nagyságrendű $H(x,y)$ megváltozása nagyon más végeredményt eredményezhet. Az inverz szűrőben nem szabad, hogy nulla érték legyen, hogy az egyenlet nevezőjében ne szerepeljen nulla.

12.3. Wiener szűrő

Nem meglepő, hogy az inverz szűrő nagyon gyenge eredményt produkál azokon a képeken, amelyeken zaj is szerepel, mivel a képleteket úgy határozták meg, hogy nem vették figyelembe a zajt. A Wiener szűrő a zajról szerzett a-priori ismereteket vizsgálja meg. A Wiener szűrő az eredeti, torzítatlan $\hat{f}$ képet a minimális középérték négyzetet e^2 hibával becsüli (E jelöli a középérték operátort) [12]

$$e^2 = E\left\{(f(i,j) - \hat{f}(i,j))^2\right\}$$

Ha nem alkalmazunk egyéb feltételeket a fenti egyenlőségre vonatkozóan, akkor az optimális $\hat{f}$ kép becslése az ideális f kép középértéke a g feltétel mellett lesz. Ezen felül az optimális f, és a torzított g kép közötti feltételes valószínűség sűrűsége a legtöbb esetben nem ismert. Általában az optimális becslést a g kép nemlineáris függvényeként lehet megadni.

A fenti egyenlőséget könnyű minimalizálni, ha az $\hat{f}$ becslése a g képben lévő értékek lineáris kombinációja. Ez a becslés csak akkor lesz az elméleti optimum, ha a szochasztikus eljárások által leírt f, és g képek, valamint a ν zaj is homogén, és a sűrűségfüggvényük Gaussi. Ezeket a feltételeket a tipikus képek nem teljesítik.

Jelöljük el a Wiener szűrő Fourier transzformáltját H_W-vel. Ekkor az eredeti kép F Fourier transzformáltját a következő képlet segítségével tudjuk megbecsülni

$$\hat{F}(u,v) = H_W(u,v)G(u,v)$$

A Wiener szűrő H_W függvényét a következő egyenlet szolgáltatja

$$H_W(u,v) = \frac{H^{\#}(u,v)}{|H(u,v)|^2 + \frac{S_{\nu\nu}(u,v)}{S_{gg}(u,v)}}$$

ahol a H jelöli a torzulás transzformációs függvényét, $\#$ a komplex konjugáltat, $S_{\nu\nu}$ a zaj spektrális sűrűségét, S_{gg} pedig a torzított kép spektrális sűrűségét. [12]

Amennyiben a Wiener szűrőt szeretnénk használni, ismerni kell a H torzítás természetét, és a zaj statisztikai paramétereit. A Wiener szűrés elmélete megoldja a posteriori linearáis négyzetátlag probléma becslését - de ehhez minden fontosabb statisztikai adat rendelkezésre kell álni (például a hatványspektrum). Az inverz szűrő valójában a Wiener szűrő egy speciális esete, amiből hiányzik a zaj, azaz $S_{\nu\nu} = 0$.

(a) (b)

12.1. ábra. Egy elmosódott kocsikép, és a Wiener szűrővel helyreállított mása

A 12.1. ábrán egy 10×10-es Gaussi zajjal elmosott képet lathatunk, és annak a Wienner szűrővel javított mását. A MatLab beépített Wiener szűrő függvénnyel rendelkezik, mely többféleképpen is felparaméterezhető annak függvényében, mennyire ismerjük a képre rakódó additív zajokat.

13. fejezet

Gábor szűrő

A Gábor szűrő egy olyan lineáris szűrő, melynek impulzus válaszát egy olyan szorzat adja, melynek egyik tagja egy harmónikus -, másik tagja pedig egy Gaussi függvény. A Gábor szűrő impulzus válaszának a Fourier transzformáltja megegyezik a harmónikus függvény Fourier transzformáltjának, és a Gaussi függvény Fourier transzformáltjának konvolúciójával, a konvolúciós szorzás miatt (konvolúciós tétel). [20]

$$g(x, y, \lambda, \theta, \psi, \sigma, \gamma) = exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) cos\left(2\pi\frac{x'}{\lambda} + \psi\right)$$

ahol

$$x' = xcos\theta + ysin\theta$$

valamint

$$y' = -xsin\theta + ycos\theta$$

A fenti egyenletben a λ a cosinus együttható, és a szűrő hullámhosszát, a θ a Gábor függvény párhuzamos csíkok normáljának irányát (orientáció), a ψ a fáziseltolást, a γ pedig a térbeli méretarányt jelenti, mely a Gábor függvény elliptikusságáért felelős.

A hullámhossz a képpontokban van specifikálva. Ennek az értéknek nagyobbnak kell lennie 2-nél. Ha $\lambda = 2$ akkor azt a $\psi = -90$, és $\psi = 90$ értékekkel együtt nem szabad használni, mert akkor a Gábor függvény a saját zéró kereszteződéséből lesz mintavételezve. Hogy a kép széleinél ne jelenjen meg semmi nem kívánt hatás, a hullámhosszot válasszuk a kép méretének ötödétől kisebbre.

Az orientáció fokokban van megadva. Értéke 0, és 360 között lehet.

A Gábor függvény koszinusz együtthatójának fáziseltolása is fokokban van megadva. Értékeit -180, és 180 közötti intervallumból veheti fel.

A méretarány a Gábor függvény elliptikusságáért felelős. Ha $\gamma = 1$, akkor kört, $\gamma < 1$ esetén pedig orientáció irányába megnyújtott képet szolgáltat.
Ez egy lehetséges *Matlab* implementáció:

```
function gb=gabor_fn(sigma_x,sigma_y,theta,lambda,psi,gamma)

sz_x=fix(6*sigma_x);
if mod(sz_x,2)==0, sz_x=sz_x+1;end

sz_y=fix(6*sigma_y);
if mod(sz_y,2)==0, sz_y=sz_y+1;end

[x y]=meshgrid(-fix(sz_x/2):fix(sz_x/2),fix(-sz_y/2):fix(sz_y/2));

x_theta=x*cos(theta)+y*sin(theta);
y_theta=-x*sin(theta)+y*cos(theta);

gb=exp(-.5*(x_theta.^ 2/sigma_x^ 2+gamma^ 2*y_theta.^ 2/sigma_y^ 2)).*
*cos(2*pi/lambda*x_theta+psi);
```

A Gábor szűrő szoros összefüggésben van a Gábor hullámokkal, mivel meghatározott számú dilatációt, és forgatást lehet velük csinálni. Általában véve nem szokták a Gábor hullámokat növelni, mert biortogonális hullámokkal való számolást igényelne, ami eléggé időigényes. Ha mégis ilyesmit kellene csinálni, akkor általában egy szűrő bankot szoktak létrehozni, melyek különböző arányú, és forgatású Gábor szűrőket tartalmaznak. A jelet konvolválják a Gábor szűrőkkel, melyek úgymond Gábor teret hoznak létre. A Gábor teret nagyszerűen fel lehet használni a képfeldolgozás különféle területein, úgy mint írisz -, és ujjlenyomat felismerés. A Gábor térből olyan fontos aktivációk nyerhetők ki, mely segítségével ritka objektum-reprezentációt készíthetünk. [1]

(a) (b)

13.1. ábra. Egy kép, és a Gábor szűrő kimenete, amikor $\lambda = 8$, $\theta = 0$, $\psi = 0.9$, $\gamma = 0.5$

14. fejezet

Összefoglalás

A számítástechnika immár több, mint 50 éve van jelen az emberiség életében. Azért alkottuk meg, hogy megkönnyítsék a mindennapjainkat, gyorsabakká, pontosabbaká tegyék számításainkat, bizonyos funkciókat autómatizáljanak, olyan dolgokat valósítsanak meg, amit eddig nem lehetett. A hagyományos képfeldolgozás egy picit öregebb tudományterület (pl. röntgenképek elemzése), de arra már a kezdeti időszakban rájöttek a kutatók, hogy számítógépek megalkotása után a képfeldolgozás területe nagyszerűen autómatizálható. Egyrészt az adatok sokassága, másrészt a pontosság megkövetelése miatt.

A digitális képfeldolgozás elsősorban az orvostudományban (pl. PET, röntgenkép elemzés, stb.), a hadiiparban (pl. pilóta nélküli felderítőgépek, ...), biztonságtechnikában (különböző riasztó-, és megfigyelőberendezések), és az élet egyéb, különböző területein használják. Azért (is) alkották meg ezt a technológiát, mert a felhasználási területeken emberéletek függhetnek/függenek. A precizitás, fontos adatok előállítása alapkövetelmény, a lényegtelen információkat pedig ki kell szűrni.

Amikor egy digitális felvételt feldolgozás céljából megkapunk, rengeteg információt, zajt, és torzítást tartalmaz(hat). Mivel ezek (esetleg rossz irányba) módosítják a számításokat, a feldolgozás előtt el kell távolítani. Első lépésként tehát egy olyan előfeldolgozást kell végrehajtani, mely eltávolítja a zajokat, helyreállítja a torzulásokat, és eltünteti a felesleges információ jó részét. A cél nem egy tökéletes kép előállítása, hanem egy olyan átmeneti adathalmazé, melyet könnyebben a következő lépésben könnyebben lehet értelmezni, feldolgozni.

A képek előfeldolgozása során elsősorban a következő három problémahalmazt szokták eliminálni: zajok, nem megfelelő hisztogramértékek, és elmosódott élek. Mindezek megvalósítása történhet vagy képtérben, vagy frekvenciatérben. Ami általánosságban kijelenthető, hogyha összehasonlítunk egy frekvenciatérben elvégzett eljárást egy képtartományban végzettével, megállapíthatjuk, hogy a képtérben végzettek azonos, vagy jobb eredményt produkálnak, és kevesebb erőforrást igényelnek.[3]

Az igazi célom a képek torzításainak, a képre rakódó zaj természetének megismerése volt, és az eddig felfedezett, mindennapi életben is elterjedt algoritmusok bemutatása. A

gyakorlatban a zajok eltüntetésére rengeteg algoritmus létezik. Az additív zajok eltüntetésére leginkább a valószínűség alapú zajeltüntetést emelném ki. Amint az a 10.2. ábrán is látszik, hogy messze ez a leghatékonyabb módszer a zajok eliminálására. A nem odaillő képpontokat eltávolítja, majd olyan intenzitású pixellel helyettesíti, mely a legnagyobb valószínűséggel oda illik, így nem torzítja a képet, nem vezet be új intenzitás értékeket. Az eljárás egyetlen negatívumát egy paraméter megadása szolgáltatja, mely képenként változhat. Funkciója ennek a paraméternek az, hogy meghatározza, mely képpont van zajjal torzítva. Így nem megfelelő érték megadása esetén nem a megfelelő eredményre számíthatunk. Értéke leginkább tapasztalati után állítandó be.

Hisztogrammok kiegyenlítésre a legtöbb képfeldolgozó rendszer, köztük a *MatLab* is rendelkezik beépített eljárásokkal. Ezek az eljárások képtartományban dolgoznak, nagyon gyorsak, és hatékonyak. Amennyiben kizárólag a képnek egy része szorul hisztogram kiegyenlítésre, erre is van lehetőség, ugyanis lehetőség van lokális hisztogramkiegyenlításre is.

Élek kiemelésére elsősorban frekvenciatartománybeli eljárásokkal lehetségesek. Ezek a konvolúciós elméleten alapszanak, de még gyors Fourier transzformáció használata esetén is kicsit lassabbak, mint a képtartománybeli másuk.
Mint azt már korábban írtam, a tárgyalt eljárások többségéhet készítetem demonstrációs programot. A többség MatLab-ban van megírva, de néhány a bonyolultság, és a MatLab rendszerben való járatlanságom miatt Delphi rendszerben.

Remélem, szakdolgozatom elnyerte az olvasó tetszését, és sikerült felkeltenem érdeklődését a képfeldolgozás iránt.

Irodalomjegyzék

[1] A. Teuner, B. J. Hosticka : Adaptive Gabor Transformation for Image Processing, 1993.

[2] Álló G., Föglein J., Hegedűs Gy. Cs., Szabó J. : Bevezetés a számítógépes képfeldolgozásba, Budapest, 1985.

[3] Álló Géza, Hegedűs Gy. Csaba, Kelemen Dezső, Szabó József : A digitális képfeldolgozás alapproblémái, Akadémiai Kiadó, Budapest, 1989

[4] A. Huertas, N. R. Stanley : Modulation transfer function associated with image transmission trough turbulent media, 1964.

[5] B. Smolka, K. N. Plataniotis, R. Lukac, A. N. Venetsanopoulus: Similarity based impulsive noise removal in color images, ICIP 2003.

[6] B. Smolka, A. Chydzinski, K. Wojciechowski, K. Plataniotis, A. N. Venetsanopoulus, On the reduction of impulsive noise in multichannel image processing, Opticl engineering, vol 40, no. 6, pp. 902 - 908, 2001

[7] D. Nicolson : Digital hardware

[8] Fazekas Attila : Bevezetés a digitális képfeldolgozásba, kézirat, KLTE 1998.

[9] Fazekas Gábor, Hajdu András : Képfeldolgozási módszerek, egyetemi jegyzet, Debreceni Egyetem, Informatikai kar, Debrecen, 2004

[10] J. Serra : Image Analysis and Mathematical Morphologym Academic Press, London, 1982.

[11] John C. Russ : The Image Processing Handbook, CRC Press, 1992.

[12] Milan Sonka, Vaclav Hlavac, Roger Boyle: Image Processing, Analysis and Machine Vision, International Thomson,computer press

[13] Rafael C. Gonzalez, Richard E. Woods : Digital Image Processing, Second Edition, Prentice Hall, 2002.

[14] Rafael C. Gonzalez, Richard E. Woods : Digital Image Processing using Matlab, Prentice Hall, 2004.

[15] R. M. Harlick, S. R. Stendberg, X. Zhuang : Image analysis using mathematical morphology, IEEE Transaction on Pattern Analysis and Machine Intelligence, 1987.

[16] Tamás Péter, Tóth Bertalan, Benkő Tiborné, Kuzmina Jekatyerina : Programozzunk Delphi Rendszerben

[17] W. K. Pratt : Digital Image Processing, Second Edition, John Wiley and Sons, New York, 1978.

[18] *http://matlabserver.cs.rug.nl*

[19] *http://www.mathworks.com/*

[20] *http://wikipedia.org*