

Szegedi Tudományegyetem
Informatikai Tanszékcsoporth
Képfeldolgozás és Számítógépes Grafika Tanszék

**Szimulált hűtésen alapuló eljárások hv-konvex
bináris képek két vetületből történő előállítására**

TDK dolgozat

Készítette:

Ozsvár Zoltán

programtervező informatikus Msc
szakos hallgató

Témavezető:

Dr. Balázs Péter

egyetemi adjunktus

Szeged
2014

Tartalomjegyzék

Tartalmi összefoglaló	3
Bevezetés	4
1. Alapfogalmak	6
2. Algoritmusok hv-konvex bináris képek rekonstruálására	9
2.1. A szimulált hűtésről általában	9
2.2. SWITCH_SA: kapcsoló komponenseken alapuló rekonstrukció szimulált hűtéssel	11
2.3. H-TILES_SA: h -konvex csempéken alapuló rekonstrukció szimulált hűtéssel	11
2.4. PIXEL_SA: pixel alapú rekonstrukció szimulált hűtéssel	16
3. Eredmények	18
3.1. A kapcsoló komponenseken alapuló szimulált hűtéses rekonstrukció vizsgálata	18
3.2. A h -konvex csempéken alapuló szimulált hűtéses rekonstrukció vizsgálata	20
3.3. A pixeleken alapuló szimulált hűtéses rekonstrukció vizsgálata	23
3.4. Az eredmények összehasonlítása	24
4. Összegzés és további tervek	25
Köszönetnyilvánítás	26
Irodalomjegyzék	27

Tartalmi összefoglaló

A dolgozatban célunk hv -konvex bináris képek rekonstruálása két vetületből. Általános hv -konvex bináris képek esetén ez a feladat ismertén NP-teljes. Az alábbi munkában kidolgozunk két szimulált hűtésen alapuló algoritmust, melyek a fenti rekonstrukciós feladat (közelítő) megoldásait keresik. Az egyik a rekonstrukciót h -konvex csempék segítségével oldja meg. A másik algoritmus pixel műveletek segítségével próbálja meg rekonstruálni a hv -konvex képeket. A hv -konvexitás mérését is két különböző módon el végezzük. A kidolgozott algoritmusok teljesítményét egy már közölt algoritmussal hasonlítjuk össze.

Kulcsszavak: bináris tomográfia; rekonstrukció; hv -konvexitás; diszjunkt régiók; 4-összefüggőség.

Bevezetés

A *tomográfiai* eljárások során célunk tárgyak belsejét leíró háromdimenziós modellek megalkotása a vizsgált objektum roncsolása nélkül. A megfigyelt tárgy modelljét annak keresztmetszeti képeiből építjük fel, melyeket vetületeik alapján határozunk meg. Egy keresztmetszet tekinthető egy kétváltozós függvénynek, míg a vetületek az adott függvény különböző irányokból vett vonalmenti integráljaiból állnak elő. A vizsgálat során sugárzással (például röntgen-, neutron- vagy elektronsugárzással) átvilágítjuk a testet. Ez a sugárzás különböző anyagi minőségű és/vagy eltérő vastagságú részekben eltérő mértékben hatol át. Az áthatolt sugárzást detektorokkal mérjük. Az így kapott adatok határozzák meg az objektum vetületeit, melyekből információhoz juthatunk a tárgy belsejével kapcsolatban. A tomográfiának számos alkalmazási területe ismert, mint például az orvosi képalkotás, a nem-roncsoló ipari minőségellenőrzés vagy az elektron mikroszkópos vizsgálatok. Az úgy nevezett szűrt visszavetítés (*FBP* - Filtered Backprojektion) a leggyakrabban alkalmazott rekonstrukciós eljárás, mely azonban a vizsgált objektumról több száz vetület készítését igényli. A vetületképzés során alkalmazott sugarak viszont károsíthatják a megfigyelt testet, emellett időigényes és költséges is lehet a vizsgálat, így nem mindig adódik lehetőség az *FBP* eljárás használatára.

A *diszkrét tomográfiát* akkor alkalmazhatjuk, ha az objektum ábrázolása során csak néhány előre ismert elnyelődési együttható jelenik meg, például ha a tárgy kevés különböző, de ismert anyagból épül fel. Ezzel az információval a szükséges vetületek száma drasztikusan csökkenthető, esetenként akár 10-20 vetület is elegendő lehet a kívánt képminőség eléréséhez [1, 2]. A *bináris tomográfia* során feltételezzük, hogy a vizsgált objektum homogén. A képpontok csak két értéket vehetnek fel, ahol 0 és 1 rendre a vizsgált anyag hiányát és jelenlétét jelöli az adott pozíciókban. Ezáltal a megkötés által sokszor már valóban kevés (2-10) vetület segítségével is rekonstruálható megfelelő minőségű eredmény. A bináris megszorításnak köszönhetően egyes esetekben akár 2 vetületből is előállítható a kép, viszont a vetületek alacsony száma miatt a probléma gyakran aluldefiniált, vagyis ugyanazon vetületeknek több különböző bináris kép is megfelelhet. A megoldáshalmaz méretének csökkentése érdekében kiegészítő feltételeket szokás megadni, amelyek a vizsgált kép további tulajdonságaira vonatkozhatnak. Ilyenkor a rekonstruált képnek ki kell

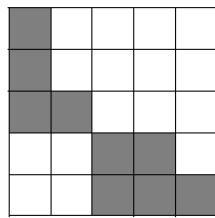
elégítenie az előre megadott vetületek mellett egyéb geometriai feltételeket is. Bizonyos geometriai tulajdonságok esetén létezik polinomiális idejű rekonstrukciós algoritmus, míg más esetekben a rekonstrukció NP-nehéz.

A dolgozat célja is egy ilyen NP-nehéz rekonstrukciós probléma vizsgálata. Két új *h_v-konvexitáson* alapuló, a szakirodalomban még nem közölt rekonstrukciós algoritmust mutatunk be és elemzünk. Az 1. fejezet a dolgozatban leírtak megértéséhez szükséges fogalmakat ismerteti. A 2. fejezet az algoritmusokat mutatja be. Végül a 3. fejezet a két algoritmusnak az elemzését és az abból levont következtetéseket tárgyalja.

1. fejezet

Alapfogalmak

Tetszőleges kétdimenziós kép számszerű leképezését *digitális képnek* nevezzük, aminek legkisebb eleme a *pixel* (vagy képpont). Egy digitális kép véges számú pixelből áll, és ezen képpontok csak egész értéket tartalmazhatnak. A *bináris kép* egy olyan digitális kép, mely a képpontokban csak kétféle értéket vehet fel. A bináris képeket kétdimenziós bináris mátrixokként is ábrázolhatjuk (1.1. ábra). A következőkben a félreértések elkerülése végett feltételezzük, hogy mindkét reprezentáció esetében az oszlopok számozását balról jobbra végezzük, illetve a sorokét fentről lefelé. Így a bal felső sarokban levő képpont az $(1,1)$ indexű elem.



(a)

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

(b)

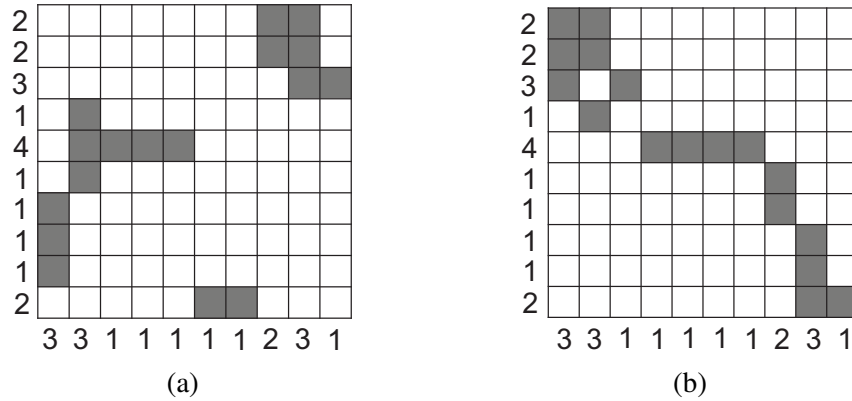
1.1. Ábra. Egy bináris kép két különböző ábrázolása: mint pixelek tömbje (a) és mint bináris mátrix (b)

Egy $m \times n$ méretű bináris kép tehát reprezentálható egy $A = (a_{ij})_{m \times n}$ bináris mátrixsal, ahol az egyesek azokat a pontokat jelzik, melyek a vizsgált terület egy objektum-pontját tartalmazzák (fekete pixelek), illetve nullák azokat amelyek háttérpontok (fehér pixelek). A továbbiakban az általánosság megszorítása nélkül feltételezzük, hogy az $m \times n$ méretű bináris mátrix minden sora és oszlopa legalább egy egyes értéket tartalmaz. Az A bináris mátrix *horizontális* és *vertikális* vetületén rendre az $R(A) = (h_1, h_2, \dots, h_m)$ és $S(A) = (v_1, v_2, \dots, v_n)$ vektorokat értjük, ahol

$$h_i = \sum_{j=1}^n a_{ij} \quad (i = 1, \dots, m), \quad v_j = \sum_{i=1}^m a_{ij} \quad (j = 1, \dots, n). \quad (1.1)$$

Jelölje $\mathcal{M}$ az $m \times n$ méretű mátrixok egy adott osztályát. A *bináris rekonstrukció* során adottak az $m, n \in \mathbb{N}$ egész számok és az $R \in \mathbb{N}^m$ és $S \in \mathbb{N}^n$ vektorok, feladatunk pedig egy olyan $A \in \mathcal{M}$ mátrixot konstruálni, melyre $R(A) = R$ és $S(A) = S$.

A rekonstrukció során előfordulhat, hogy több megoldás is kielégíti ugyanazon vetületeket (1.2. ábra). Ennek oka a *kapcsoló komponensek* jelenléte, amik egy bináris mátrix $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ és $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ részmátrixai. Hangsúlyozzuk, hogy ezen részmátrixok nem feltétlenül két egymást követő sor/oszlop kiválasztásaként adódnak. A kapcsoló komponensek pozícióiban lévő értékeket invertálva egy olyan bináris mátrixot kapunk, melynek horizontális és vertikális vetületei ugyanazok, mint az eredeti mátrixé. Tudjuk, hogy az ilyen kapcsoló komponens jelenléte szükséges és elégséges feltétele annak, hogy a megoldás ne legyen egyértelmű. Ismert továbbá, hogy ha az A és B különböző mátrixok ugyanazon horizontális és vertikális vetületekkel rendelkeznek, akkor az A mátrixból el lehet jutni B mátrixba (és fordítva) kapcsoló komponensek véges számú átkapcsolásával [3].

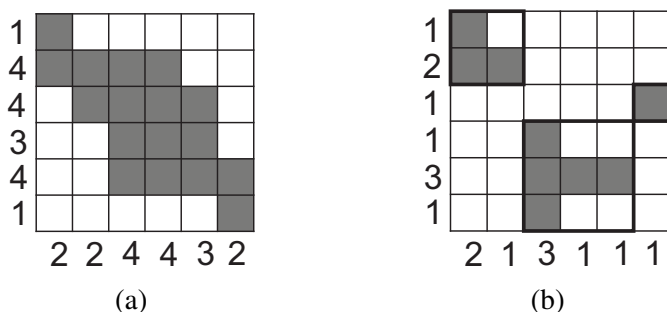


1.2. Ábra. Két különböző kép, melyek vetületei megegyeznek.

A rekonstrukcióhoz tehát szorosan kapcsolódik az *egyértelműség* eldöntésének kérdése is, azaz hogy egy $A \in \mathcal{M}$ mátrixhoz létezik-e olyan $A' \in \mathcal{M}$ bináris mátrix, melyre $R(A') = R(A)$ és $S(A') = S(A)$, de $A' \neq A$.

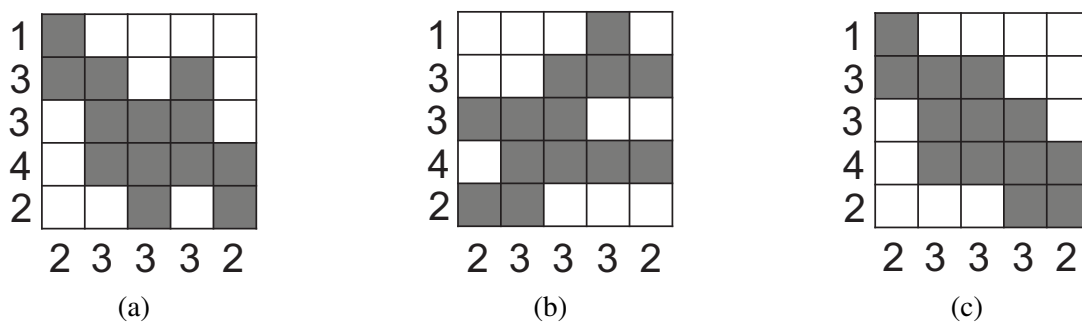
Különböző geometriai megszorításokkal csökkenthetjük a megoldáshalmaz méretét. Ilyen megszorítások lehetnek az *összefüggőség* és a *konvexitás*. Egy A bináris mátrix két pontja, $P = (p_1, p_2)$ és $Q = (q_1, q_2)$, *4-szomszéd*, ha $|p_1 - q_1| + |p_2 - q_2| = 1$. A $P_0, \dots, P_k$ különböző pontokból álló sorozat *4-út* a P_0 pontból P_k pontba, ha a sorozat minden pontjában A értéke 1, és P_l és P_{l-1} 4-szomszédok minden $l = 1, \dots, k$ -ra. Az A bináris mátrix *4-összefüggő*, ha bármely két 1-es értékű pontja között van A -beli 4-út. A 4-összefüggő bináris mátrixokat *poliominóknak* is nevezzük. Ha egy bináris mátrix nem 4-

összefüggő, akkor több poliominóból áll. Az ilyen maximális 4-összefüggő részmatrixok (részképek) az A bináris mátrix *komponenseit* adják (1.3. ábra).



1.3. Ábra. Egy 1 komponenst (a), és egy 3 komponenst (b) tartalmazó kép. Ez utóbbi képen a komponensek legkisebb befoglaló téglalapjait vastag vonalakkal ábrázoltuk

Egy bináris mátrix *horizontálisan konvex* (vagy *h-konvex*) ha minden sorában az egyesek egy összefüggő sorozatot alkotnak. Hasonlóan egy bináris mátrix *vertikálisan konvex* (vagy *v-konvex*), ha minden oszlopában az egyesek egy összefüggő sorozatot alkotnak. Ha egy bináris mátrix horizontálisan és vertikálisan is konvex, akkor *hv-konvexnek* nevezzük (1.4. ábra). Egy ilyen megszorítás már kizárja például az 1.2b ábrán látható képet, mint lehetséges megoldás.



1.4. Ábra. Egy 4-összefüggő *v*-konvex bináris kép (a), egy 4-összefüggő *h*-konvex bináris kép (b), egy 4-összefüggő *hv*-konvex bináris kép (c) ugyanazon vetületekkel.

2. fejezet

Algoritmusok hv -konvex bináris képek rekonstruálására

A 4-összefüggő hv -konvex bináris mátrixok rekonstrukciója két vetületből polinomiális időben megoldható. Erre vonatkozóan két megközelítést ismerünk. Az egyik egy úgynevezett mag-burok eljárás [4, 5], a másik pedig 2SAT problémára való visszavezetésen alapozik [6]. Az általános hv -konvex bináris mátrixok esetén azonban a feladat NP-teljessé válik, így nem várható, hogy polinomiális idejű algoritmust találunk rá [7]. Számos heurisztikus algoritmus létezik azonban, melyek igyekeznek hatékony választ adni az általános hv -konvex rekonstrukció kérdésére [8, 9, 10]. Egy előző munkában rámutattunk, hogy ezek teljesítménye nagyon különböző lehet, melyet befolyásolhat a rekonstruálni kívánt kép mérete, illetve az azt alkotó komponensek pozíciói, nagysága és darabszáma is [11]. Jelen dolgozat célja a szimulált hűtés, mint optimalizációs eljárás alkalmazhatóságának vizsgálata az adott problémára. Az eljárás általános volta miatt - ahogy azt majd látni fogjuk - lehetőség nyílik a bináris képek különböző strukturáltságának kezelésére (pixelek, kapcsoló komponensek, csempék).

Ebben a fejezetben először ismertetjük a szimulált hűtési keretrendszerünket, majd bemutatunk egy referenciának szánt algoritmust, mely kapcsoló komponensek segítségével próbálja megtalálni a feladat megoldását. Utána ismertetjük az általunk kidolgozott algoritmusokat. Az első h -konvex csempéket, a másik pedig pixeleket és azokkal való műveleteket alkalmaz a rekonstrukció során.

2.1. A szimulált hűtésről általában

Kombinatorikus optimalizáció során igyekszünk az adott feltételek szerinti legjobb megoldást kiválasztani egy adott lehetséges megoldásokat tartalmazó halmazból. A halmaz elemeit *állapotoknak* is nevezzük, melyeknek *jóság értékét*, egy f célfüggvény definiálja.

Az állapotok között mozogni is tudunk, így változtatva a célfüggvény értékét. Feladatunk megtalálni ennek a célfüggvénynek a maximumát vagy minimumát. Az optimum keresése során problémát jelent, hogy az egyszerűbb optimalizációs (pl. hegymászáson alapuló) eljárások beragadhatnak a lokális optimumokba. A *szimulált hűtés* során megengedünk az aktuálisnál rosszabb új állapotokat is, ezáltal elkerülhetjük a lokális optimumba ragadást. Az eljárás során egy új állapotot mindenképpen elfogadunk, ha jobb mint az előző állapot. Lehetőséget adunk azonban az új állapot elfogadására akkor is, ha az rosszabb (kevésbé optimális), mint a jelenlegi megoldás. Az eljárás előrehaladtával folyamatosan csökken annak a valószínűsége, hogy elfogadjuk a rosszabb állapotokat a $p < \exp(\frac{-|d|}{Temp})$ feltétel segítségével, ahol $0 < p < 1$ egy minden iterációban újra generált véletlen szám, $d = f(elozo) - f(aktualis)$, *Temp* pedig az aktuális hőmérséklet. Amennyiben elfogadunk egy állapotot, akkor elmentjük azt, csökkentjük a hőmérsékletet, és lenullázzuk a számlálót, ami számon tartja, hogy hány iteráció óta nincs jobb állapot. Ellenkező esetben növeljük a számlálót. Ezt az eljárást addig folytatjuk míg a hőmérséklet el nem éri az általunk kívánt értéket, vagy ha adott számú iteráció óta nem találtunk jobb megoldást. A dolgozatban szereplő algoritmusoknál a szimulált hűtés keretrendszerét fogjuk alkalmazni (1. algoritmus).

Algoritmus 1: Szimulált hűtés keretrendszer

Input: $R \in \mathbb{N}^m$ és $S \in \mathbb{N}^n$ két vektor; $Temp = t$; $Nbr = 0$;

Output: Bináris mátrix R sorösszeggel és S oszlopösszeggel.

```
1 Kezdőállapot kiválasztása;
2 while ((Temp < minTemp) és (Nbr < MaxNbr)) do
3   Lepes;
4    $d := f(elozo) - f(aktualis)$ ;
5   if (( $d < 0$ ) vagy ( $p < \exp(\frac{-|d|}{Temp})$ )) then
6     Lépés elfogadása;
7     előző = aktuális;
8     Temp csökkentése;
9     Nbr = 0;
10  else
11    | Nbr ++;
12  end
13 end
14 end
```

2.2. SWITCH_SA : kapcsoló komponenseken alapuló rekonstrukció szimulált hűtéssel

A tervezett algoritmusok teljesítményének méréséhez szükségünk van egy referenciára, ami egy a szakirodalomban már megjelent algoritmus lesz. Ez az eljárás szintén szimulált hűtésen alapul, ami egy jól választott kezdőállapotból indul az optimum felé [10]. A kiindulási állapotot a Ryser algoritmus segítségével állítjuk elő (Algoritmus 1, 1. lépés). A Ryser algoritmus eredménye kielégíti a horizontális és vertikális vetületet, viszont általában nem *hv*-konvex [3]. Célunk, hogy minél *hv*-konvexebb legyen a végső állapot. Egy állapot "jóságát" a konvexitás mértéke határozza meg, amit a szomszédos egyesek száma ad meg:

$$f(A) = \sum_{i=1}^{m-1} \sum_{j=1}^n a_{ij} a_{(i+1)j} + \sum_{i=1}^m \sum_{j=1}^{n-1} a_{ij} a_{i(j+1)} \quad (i = 1, \dots, m; j = 1, \dots, n), \quad (2.1)$$

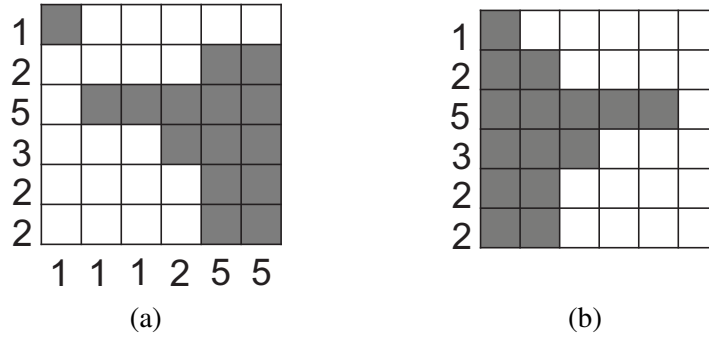
ahol A az eljárás adott iterációjában szereplő bináris kép. Ennél a feladatnál tehát a cél-függvényt maximalizálnunk kell. A kapcsoló komponensek átkapcsolásával jutunk az egyik állapotból a másikba (Algoritmus 1, lépés 3.). Ez biztosítja, hogy a rekonstrukció során csak a kép fog változni, annak vetületei nem. Az eljárás terminál, ha hosszabb ideig nem sikerül jobb eredményt elérni, vagy elérjük a kilépési feltételben megadott hőmérsékletet. Hosszas kísérletezgetés után az alábbi paraméterezést alkalmaztuk: $Temp = 50$, $Temp$ csökkentés: $Temp = Temp \cdot 0.999$, $minTemp = 0.001$, $MaxNbr = 1000$.

2.3. H-TILES_SA : *h*-konvex csempéken alapuló rekonstrukció szimulált hűtéssel

A szakirodalomban található bináris képeket rekonstruáló algoritmusok gyakran pixel alapúak, ami azt jelenti, hogy a rekonstrukció során alkalmazott lépéseknél egy-egy képpont változtatásával közelítenek a megoldás felé. A most ismertetésre kerülő algoritmus pixelek helyett nagyobb alakzatok, úgy nevezett *csempék*, diszjunkt uniója segítségével próbálja meghatározni a kapott vetületekből a képet.

Az algoritmus alapötlete a következő. Mindegyik sorban elhelyezünk egy összefüggő egyesekből álló „csíkot” (csempét), úgy hogy az adott sor vetületi értéke határozza meg a csempe hosszát. Természetesen az így elhelyezett elemek *h*-konvexek. Ezen elemeket a függőleges vetületek alapján is létre lehetne hozni, mi azonban az egyszerűség kedvéért csak a vízszintes vetületek alapján határozzuk meg a kiindulási képet. Például ha a képünk

a 2.1a. ábrán látható, melynek vetületei $R = (1, 2, 5, 3, 2, 2)$ és $S = (1, 1, 1, 2, 5, 5)$, akkor 6 darab h -konvex csempével fogunk foglalkozni (2.1b. ábra), melyek mérete az első sortól az utolsóig: 1, 2, 5, 3, 2, 2.



2.1. Ábra. Bináris kép, melynek vetületei: $R = (1, 2, 5, 3, 2, 2)$ és $S = (1, 1, 1, 2, 5, 5)$ (a), az adott képből kapható h -konvex elemek, ahol minden elem az első oszloptól kezdődik (b)

A rekonstrukció során a csempéket horizontálisan mozgathatjuk. Célunk, hogy megtaláljuk a csempék pozícióinak azon kombinációját, mely (megközelítőleg) kielégíti a vertikális vetületet is. További feltétel, hogy az eredmény (szintén megközelítőleg) hv -konvex legyen. Mivel az említett elemek kivétel nélkül vízszintesen konvexek, ezért a h -konvexitás garantált az algoritmus egész futása alatt, tehát csak a v -konvexitást kell ellenőriznünk a rekonstrukció során. Nagy méretű kép rekonstruálásánál a csempék elhelyezésének rengeteg kombinációját kell kipróbálni, így nem célszerű a megoldást a keresési tér teljes bejárása segítségével előállítani. Ezért optimalizálási feladattal közelítjük meg a problémát. A h -konvex elemek megfelelő elhelyezéséhez szimulált hűtést alkalmazunk, melyhez az 1. algoritmusban ismertetett keretrendszert használjuk. A kezdeti állapot kialakításához a h -konvex elemeket véletlenszerűen helyezzük el, minden csempe egy véletlenszerűen választott oszlopban fog kezdődni, úgy hogy ne lógjon ki egyik vége sem az $m \times n$ -es területről (Algoritmus 1, 1. lépés). Az optimalizálás során a kezdő hőmérséklet 50. Az eljárás terminál, ha a hőmérséklet eléri a kilépési feltételben megszabott értéket ($\min Temp = 0.005$) vagy ha 500 iteráció során nem talált jobb megoldást ($\max Nbr = 500$). Ezeket a paramétereket szintén hosszas kísérletek során határoztuk meg. Minden iterációban véletlenszerűen egyenletes eloszlás alapján választunk egy csempét, amit egy szintén véletlenszerűen választott pozícióba fogunk áthelyezni úgy, hogy az $m \times n$ -es területről ne lógjon ki egyik oldala sem (Algoritmus 1, 3. lépés). Az f célfüggvény értékét az alábbi módon határozzuk meg. Először számítsuk ki a vetületi hibát a vertikális vetületre. Ehhez az eredeti vetületek és az aktuális vetület abszolút különbségét állapítjuk meg:

$$P = \sum_{j=1}^n |v_j^o - v_j^r| \quad (v_i^o \in R^o; v_i^r \in R^r; j = 1, \dots, n), \quad (2.2)$$

ahol R^o az eredeti, és R^r az aktuális vertikális vetület. Ezek után megállapítjuk a v -konvexitás mértékét. Ezt többféleképpen is számíthatjuk. Legegyszerűbben úgy, hogy összeszámoljuk az oszlopokban levő vertikális szomszédokat:

$$K = \sum_{i=1}^{m-1} \sum_{j=1}^n a_{ij} a_{(i+1)j} \quad (i = 1, \dots, m; j = 1, \dots, n). \quad (2.3)$$

Ezután az f függvény értéket a v -konvexitásnak és a vetületi hiba értékének a különbsége fogja meghatározni

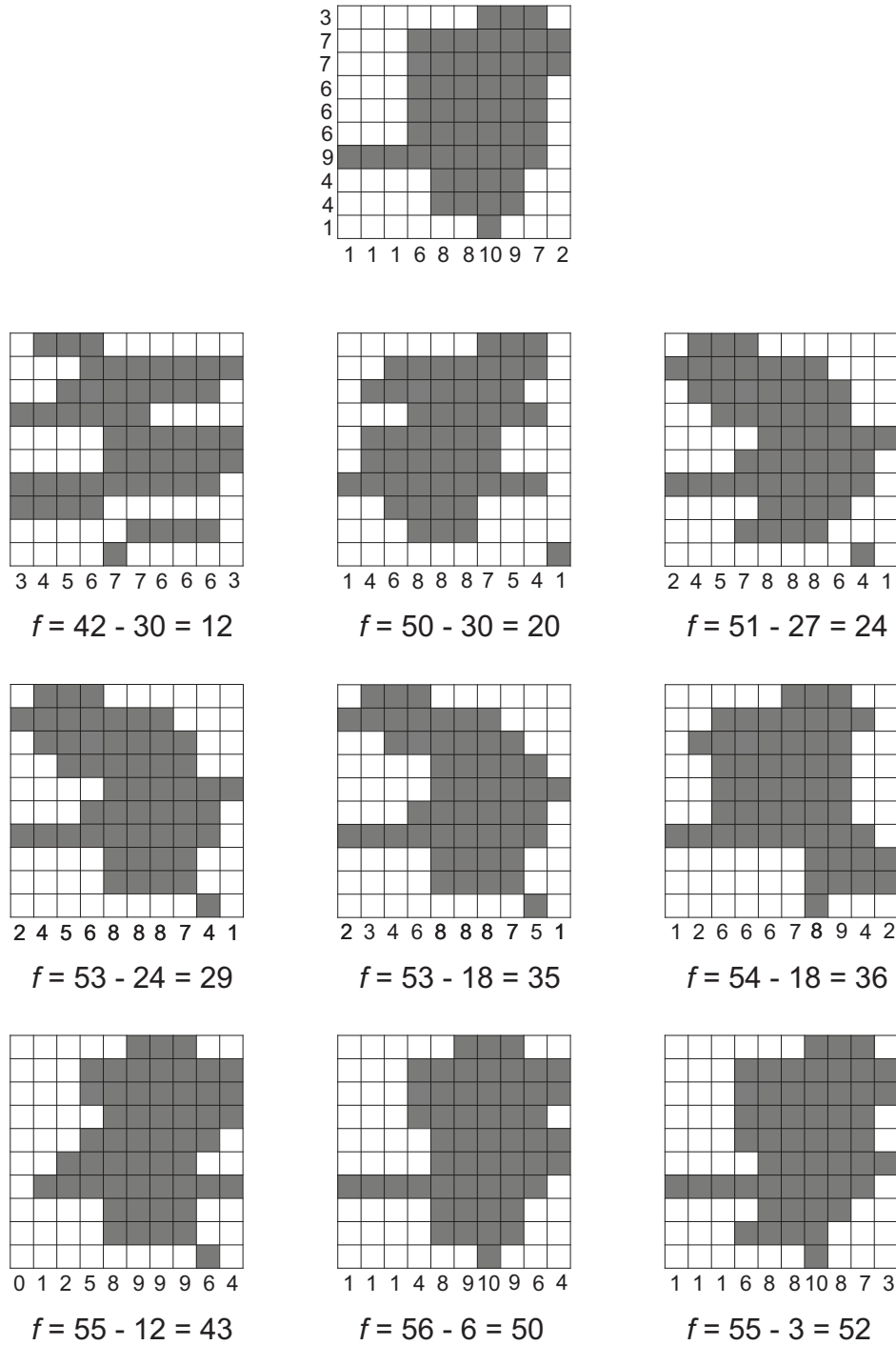
$$f(A) = K - \gamma \cdot P, \quad (2.4)$$

ahol a γ paraméter elősegíti, hogy azonos nagyságrendű legyen a célfüggvény két tagja. Ebben az esetben az f célfüggvény maximalizálandó. Amennyiben a (2.3) képlet alapján határozzuk meg a konvexitást, akkor a γ paraméter értékére egy becslés adható, a maximális v -konvexitás és a maximális vetületi hiba hányadosaként. Ez az érték a tapasztalataink alapján megközelítőleg 1.5, ezért a rekonstrukciók során a $\gamma = 1.5$ értéket használjuk. Az eljárás néhány iterációja a 2.2. ábrán figyelhető meg.

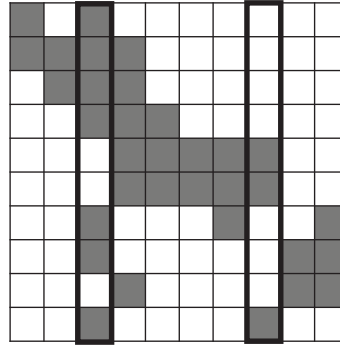
Másik módszer a v -konvexitás megállapítására az úgy nevezett *iránymenti* konvexitás mérése [12]. Az oszlopok szerint vizsgáljuk a konvexitást, a következőképpen. Bontsuk fel az oszlopot 1-ek és 0-ák futamaira. A *futam* egy maximális hosszágú egymást követő elemekből álló sorozat, amelyekben az elemek értéke azonos. A *futam hosszán* a futamban szereplő azonos elemek számát értjük. Az *1-futam* egyeseket tartalmazó futam, míg a *0-futam* nullákat tartalmaz. Példaként a 0011100010 sorozatot az alábbi formában is ábrázolhatjuk: $0^2 1^3 0^3 1^1 0^1$, ahol az indexszám a futam hosszát ábrázolja. Mint nemsokára látni fogjuk a kezdő és záró 0-futamok nincsenek hatással a konvexitásra, így feltételezhető, hogy a kép minden oszlopjának leírása 1-futammal kezdődik és végződik. Ezek alapján egy vizsgált oszlop $1^{k_1} 0^{l_1} 1^{k_2} 0^{l_2} \dots 1^{k_n}$ alakban írható le, ahol n az 1-futamok száma és $k_1, l_1, k_2, l_2, \dots, k_n > 0$. Két különböző (i -edik és j -edik) 1-futam között a *nem-konvexitás* értéke az alábbiak szerint számítható

$$k_i k_j \sum_{t=i}^{j-1} l_t. \quad (2.5)$$

Egy teljes oszlop vizsgálata esetén a benne levő 1-futamok összes lehetséges párosítására (melyből $\binom{n}{2}$ darab van) kiszámoljuk a (2.6) képletet és az eredményeket összeadjuk



2.2. Ábra. A H-TILES_SA algoritmus pár lépése a végeredménnyel, ahol a konvexitás meghatározásához a 2.3 képletet alkalmaztuk. Az első kép az eredeti képet ábrázolja. Alatta a rekonstrukció előrehaladása során jelentkező néhány állapot látható, ahol az ábrák alatt az $f(A) = K - (1.5 \cdot P)$ jószágértékek találhatók.



2.3. Ábra. A harmadik oszlop az 1111001101 sorozatot tartalmazza, melynek értéke $\varphi = 4 \cdot 2 \cdot 2 + 4 \cdot 1 \cdot 3 + 2 \cdot 1 \cdot 1 = 30$. A nyolcadik oszlopban a 000110001 sorozat nem-konvexitása $\varphi = 2 \cdot 1 \cdot 3 = 6$.

$$\varphi = \sum_{1 \leq i < j \leq n} k_i k_j \sum_{t=i}^{j-1} l_t. \quad (2.6)$$

A nagyobb φ kisebb konvexitást jelent, ezért arra törekszünk, hogy minimalizáljuk ezt az értéket. A 2.3 ábra egy példát mutat egy bináris képre, ahol a kiemelt oszlopokban kiszámoltuk a φ értékét. Értelemszerűen a φ értékét az összes oszlopra ki kell számolni, legyen tehát

$$K = \sum_{r=1}^n \varphi_r, \quad (2.7)$$

ahol φ_r az r -edik oszlopban mért iránymenti nem-konvexitás mértéke. Ezek után a következőképpen állapíthatjuk meg a célfüggvény értékét

$$f(A) = K + \gamma \cdot P, \quad (2.8)$$

amit ez esetben minimalizálni fogunk.

2.4. PIXEL_SA: pixel alapú rekonstrukció szimulált hűtéssel

A következő algoritmus pixel műveleteket fog alkalmazni a szimulált hűtés során, hogy előállítsa a rekonstruált képet. Először kezdeti állapotként készítünk egy képet, mely véletlenszerűen egyeseket és nullákat tartalmaz egyenletes eloszlás alapján.

A célfüggvény meghatározásához szükségünk lesz az iránymenti konvexitás értékére. Ehhez először megállapítjuk a normalizált nem-konvexitást az egyes sorokra a (2.6) képlet alapján

$$\hat{\varphi} = \frac{\varphi}{(n/3)^3}, \quad (2.9)$$

ahol n a vizsgált kép oszlopainak a száma [12]. Ezután az sorok nem-konvexitásának átlagát vesszük az m darab sorban.

$$\Phi_h = \frac{\sum_{r=1}^m \hat{\varphi}_r}{m}, \quad (2.10)$$

ahol a $\hat{\varphi}_r$ normalizált nem-konvexitás értéke az r -edik sorban. Hasonlóan számítjuk a Φ_v vertikális nem-konvexitás normalizált értékét is. Tapasztalatok azt mutatják, hogy a $\Phi_h + \Phi_v$ összeg maximuma 1.5 körül van. Tehát ahhoz, hogy a horizontális és vertikális iránymenti nem-konvexitást (megközelítőleg) normalizáljuk, a $\Phi_h + \Phi_v$ összegét el kell osztanunk 1.5-tel.

Szükségünk van továbbá a vetületi hiba számítására is, amit az alábbi képlet segítségével határozzunk meg

$$P = \frac{\sum_{i=1}^m (h_i^o - h_i^r) + \sum_{j=1}^n (v_j^o - v_j^r)}{m \cdot n} \quad (2.11)$$

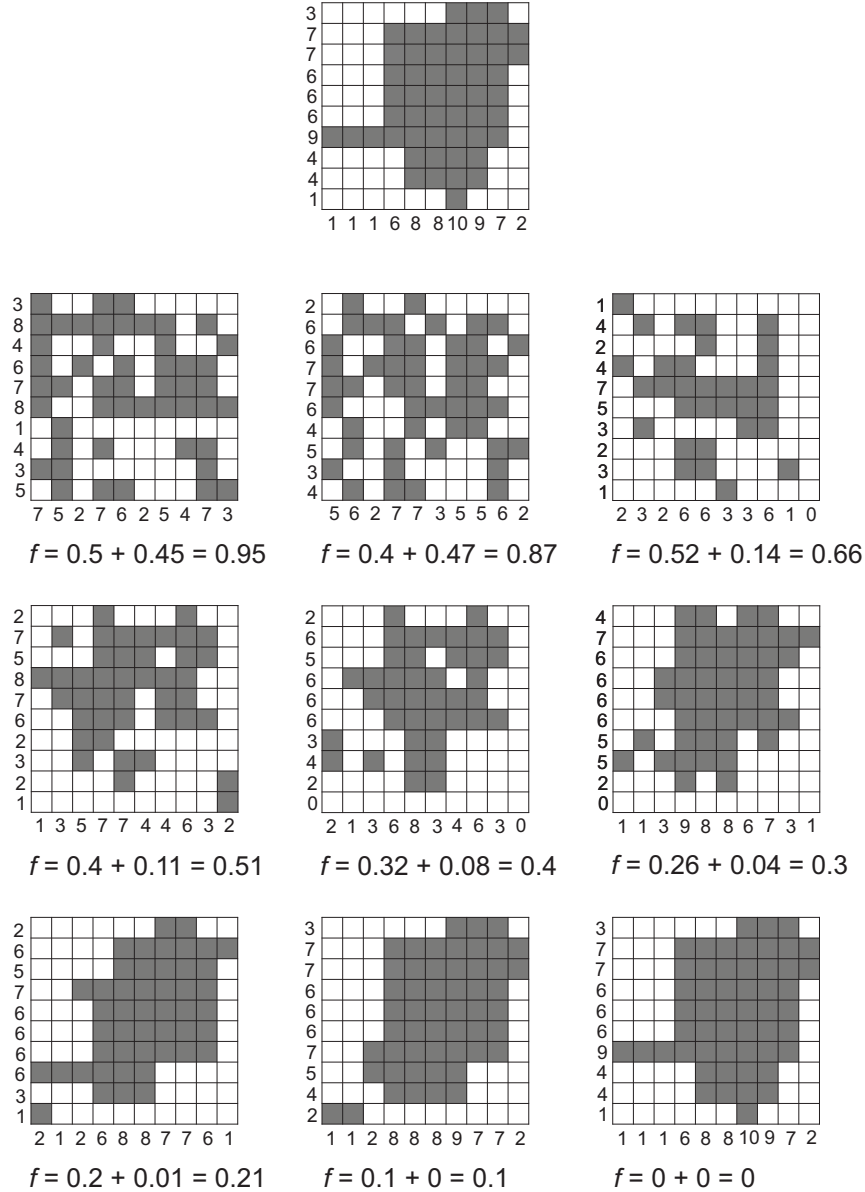
$$(v_i^o \in R^o; h_i^o \in S^o; v_i^r \in R^r; h_i^r \in S^r; i = 1, \dots, m; j = 1, \dots, n),$$

ahol a képletben szereplő v_j^o és h_i^o az eredeti kép vetületének j -edik oszlopát és i -edik sorát, továbbá v_j^r és h_i^r az aktuális iterációban szereplő kép vetületének j -edik oszlopát és i -edik sorát jelölik. Ezek alapján a célfüggvény a normalizált projekciós hiba és a normalizált horizontális és vertikális iránymenti nem-konvexitás összege lesz

$$f(A) = P + \frac{\Phi_h + \Phi_v}{1.5}. \quad (2.12)$$

Az eljárás során célunk ezen célfüggvény minimalizálása. Ahhoz, hogy egyik állapotból a másikba eljussunk választunk egy pixelt véletlenszerűen egyenletes eloszlás alap-

ján és invertáljuk az értékét. A szimulált hűtés paraméterei az alábbiak: $Temp = 50$, $Temp = Temp \cdot 0.99$, Temp csökkentés: $minTemp = 0.009$, $MaxNbr = 1000$, melyeket próbálgatás során állapítottuk meg. Az eljárás néhány iterációja a 2.4. ábrán figyelhető meg.



2.4. Ábra. A PIXEL_SA algoritmus pár lépése. Az eljárás a 2.2. ábrán található képet rekonstruálja. A képen látható a rekonstrukció előrehaladása során jelentkező pár állapot, ahol az ábrák alatt az $f(A) = P + \frac{\Phi_h + \Phi_v}{1.5}$ jószágértékek találhatók.

3. fejezet

Eredmények

Az algoritmusok teljesítményének vizsgálatát több szempont alapján végeztük. A teszt-környezet az alábbi paraméterekkel rendelkezett: Processzor Intel[®] Core[™] i7-2600 CPU @ 3.40 GHz $\times$ 8; Memória 7,8 GiB; operációs rendszer: Ubuntu 13.04 (64 bit). Az algoritmusokat C++-ban implementáltuk. A vizsgálat 2000 $h\nu$ -konvex képen folyt, melyeknek méretei 10×10 -től 50×50 -ig terjedtek 10 pixel lépésközzel, illetve 70×70 -es és 100×100 -as képekre is végeztünk vizsgálatokat. A képek egyenletes eloszlásból származtak és különböző számú komponenseket tartalmaztak (1-től 4-ig) [13, 14]. A szimulált hűtés sztochasztikus volta miatt a teszteket minden kép esetében 10-szer végeztük el, és az eredmények átlagát vettük.

3.1. A kapcsoló komponenseken alapuló szimulált hűtési rekonstrukció vizsgálata

A SWITCH_SA algoritmus sebességét tekintve megállapíthatjuk, hogy a nagyobb méretű képek esetén a futásidő csökken a komponensek számának növekedésével (3.1. táblázat). Ennek oka, hogy egy komponens esetén relatíve kevesebb kapcsoló komponens van, így kisebb a valószínűsége, hogy olyan képet találunk, amely kielégíti a vetületeket, úgy, hogy a kép eközben $h\nu$ -konvex is.

A következő vizsgálati szempont a rekonstrukció pontossága. Mivel a SWITCH_SA algoritmus adhat olyan megoldást is eredményül, mely nem $h\nu$ -konvex, ezért érdemes megvizsgálni, hogy a rekonstruált kép mekkora mértékben tér el az eredeti képtől. Legyen $A^o = (a_{ij}^o)$ az eredeti és $A^r = (a_{ij}^r)$ a rekonstruált kép bináris mátrixa. A rekonstrukció pontosságának a mértékét (relative mean error - RME) a (3.1) képlet segítségével határozzuk meg, ahol az eredeti és a rekonstruált kép közötti eltérő pixelek számát normalizáljuk az eredeti képen szereplő objektumpontok számával.

Futás idők - SWITCH_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	10.81	44.17	40.20	186.76	56.97	315.09	102.29	584.30	120.84	633.93
2	10.24	43.50	20.94	114.74	34.28	158.23	57.23	193.04	110.25	497.38
3	13.91	53.06	32.25	154.95	32.77	119.11	58.01	182.63	98.65	311.03
4	11.96	44.19	42.48	187.26	37.81	155.05	52.65	167.18	97.08	301.74

3.1. táblázat. A SWITCH_SA algoritmus futásidejei milliszekundumban megadva. A sorok a komponensek számát ábrázolják, az oszlopok a tesztképek méreteit.

RME - SWITCH_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	0.05	0.17	0.05	0.19	0.07	0.23	0.07	0.24	0.06	0.22
2	0.05	0.22	0.06	0.24	0.08	0.29	0.10	0.31	0.10	0.33
3	0.05	0.22	0.06	0.24	0.10	0.34	0.12	0.37	0.11	0.36
4	0.07	0.28	0.09	0.32	0.11	0.36	0.11	0.37	0.12	0.36

3.2. táblázat. A SWITCH_SA algoritmus RME értékei. A sorok a komponensek számát ábrázolják, az oszlopok a tesztképek méreteit.

$$RME = \frac{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}^o - a_{ij}^r|}{\sum_{i=1}^m \sum_{j=1}^n a_{ij}^o} \quad (i = 1, \dots, m; j = 1, \dots, n). \quad (3.1)$$

Ahogy a 3.2 táblázatból látható a rekonstrukció pontossága csökken a komponensek számának növekedésével. Ennek oka vélhetően az, hogy több komponenses képnél a teljes képmérethez viszonyítva arányaiban kevesebb objektumpont van, és kevesebb objektumpont esetén több olyan megoldás létezhet, ami nem hasonlít az eredeti képre. (3.3. táblázat). A képek méretének növelése során szintén csökken a rekonstrukció pontosságára.

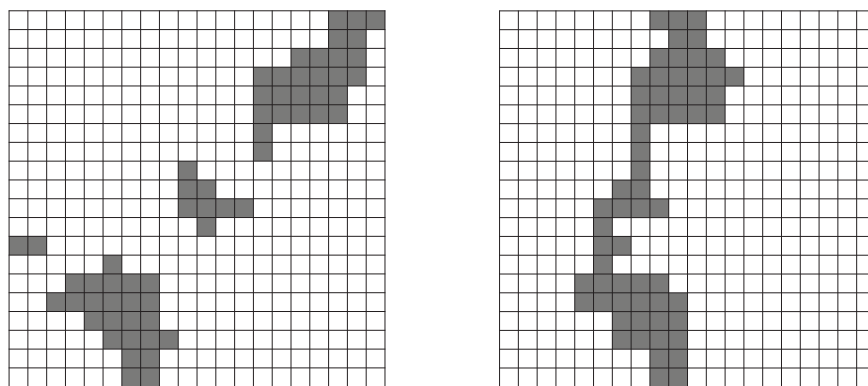
Objektumpontok száma										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	48.98	10.62	176.57	41.16	370.69	85.00	638.67	183.29	1032.38	260.58
2	31.76	7.75	116.34	31.48	236.26	64.47	392.38	101.98	607.21	198.08
3	26.69	6.43	92.36	26.79	186.48	55.17	306.45	97.79	443.27	118.73
4	21.20	4.52	69.59	16.54	150.10	39.68	250.15	74.18	392.97	108.89

3.3. táblázat. A tesztelt képek objektumpontjainak átlaga és szórása. A sorok a komponensek számát ábrázolják, az oszlopok a tesztképek méreteit.

3.2. A *h*-konvex csempéken alapuló szimulált hűtéses rekonstrukció vizsgálata

Először a H-TILES_SA algoritmus szomszédos pixeleken alapuló mértéket alkalmazó változatát vizsgáltuk. Az eredményeket vizsgálva (3.4. táblázat) látható, hogy a rekonstrukció idejét nagymértékben befolyásolja a kép mérete és komponenseinek száma. A kép méretének a növelése során több *h*-konvex csempét kell elhelyezni, ami lassítja a rekonstrukciót. Több komponens esetén, ahogy ezt már a 3.3 táblázatban is láttuk, általánosságban kevesebb objektumpontot tartalmaz a kép (3.3. táblázat), melynek eredményeképpen kisebbek a csempék és így több pozícióba lehet őket elhelyezni, ami szintén növeli a futásidőt.

A 3.5. táblázatból leolvasható, hogy a rekonstrukció pontossága nem változik szignifikáns mértékben a kép méretének a változtatása esetén. Viszont a komponensek számának növelésével növekszik az RME értéke. Ennek vélhetően megint az az oka, hogy több komponens esetén kevesebb az objektumpont, így könnyebb olyan megoldást találni, ami jobban eltér az eredeti képtől. További probléma, hogy a konvexitás mértékét a vertikális szomszédok száma határozza meg. Ez az érték képméretenként és komponensszámtól függően eltérő lehet, ezért nehéz megadni a pontos arányt a vetületi hiba és a konvexitás között az állapot jóságának becslése során (emlékezzünk vissza, hogy a (2.4) képletben $\gamma = 1.5$ faktor csak egy tapasztalatokon alapuló becslés volt). Így előfordulnak olyan esetek, amikor az algoritmus a szomszédos egyesek számát maximalizálva egy oszlopba próbálja rendezni a csempéket, kevésbé ügyelve a projekciós hibára (3.1. ábra).



3.1. Ábra. Eredeti kép (bal oldal). Az eredeti kép vetületeiből rekonstruált kép H-TILES_SA eljárás segítségével (jobb oldal).

Az irány szerinti konvexitás alkalmazásával ezek a szituációk elkerülhetők, jobb eredményeket lehet elérni. Sajnos ezzel a módszerrel műveletigényes az összes oszlopot megvizsgálni, ezért különböző stratégiákat dolgoztunk ki a futásidő csökkentése érdekében. Referenciaként az összes oszlop vizsgálatát *ALL* stratégiának fogjuk nevezni. Első meg-

Futás idők - H-TILES_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	1.05	3.33	2.52	7.99	4.50	14.07	7.61	24.03	9.61	30.05
2	1.34	4.13	3.49	10.67	5.89	17.92	10.31	31.31	11.98	36.43
3	1.48	4.54	3.87	11.75	6.53	19.83	11.02	33.44	13.17	39.87
4	1.84	5.65	4.38	13.26	6.89	20.81	11.72	35.48	13.49	41.02

3.4. táblázat. A szomszédos pixeleken alapuló mértéket alkalmazó H-TILES_SA algoritmus futásidejei milliszekundumban megadva. A sorok a komponens számokat ábrázolják, az oszlopok a tesztképek méreteit.

RME - H-TILES_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	0.06	0.19	0.06	0.22	0.07	0.22	0.07	0.24	0.06	0.22
2	0.11	0.34	0.11	0.34	0.11	0.36	0.12	0.38	0.13	0.40
3	0.12	0.38	0.12	0.38	0.13	0.41	0.14	0.43	0.14	0.44
4	0.13	0.41	0.15	0.45	0.14	0.45	0.14	0.45	0.15	0.45

3.5. táblázat. A szomszédos pixeleken alapuló mértéket alkalmazó H-TILES_SA algoritmus RME értékei. A sorok a komponens számokat ábrázolják, az oszlopok a tesztképek méreteit.

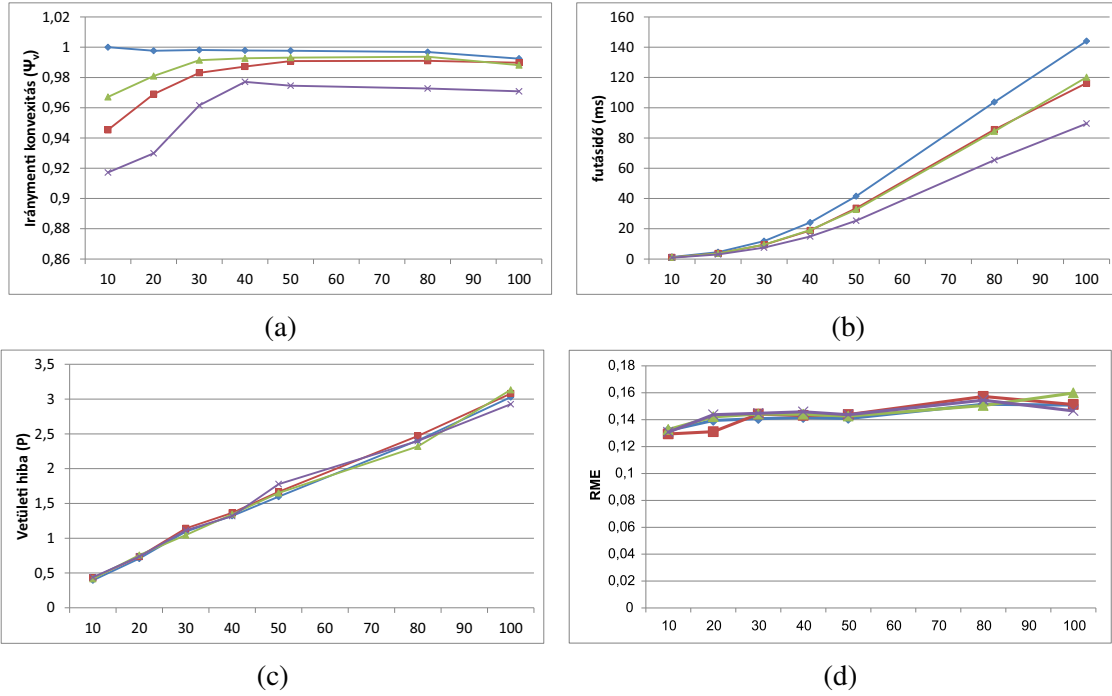
közelítésben csak a páros oszlopokat vizsgáljuk (a módszer neve a továbbiakban *EVEN* lesz). A második stratégia szerint ugyanúgy az oszlopok felét vizsgáljuk, viszont egyenletes eloszlás szerint választjuk ki véletlenszerűen a vizsgált oszlopokat minden iterációban (*RANDOM*). Az utolsó módszer esetén először csak az oszlopok 10%-át vizsgáljuk, majd az algoritmus futása során a hőmérséklet csökkenése alapján lineárisan növeljük a vizsgált oszlopok számát (*INCREASE*). Ezeket az oszlopokat szintén egyenletes eloszlás alapján véletlenszerűen választjuk. Az iránymenti konvexitás mérésére az alábbi képletet alkalmaztuk

$$\Psi_v = 1 - \Phi_v , \quad (3.2)$$

ahol Φ_v a (2.9) képletben adott. Ez az érték a $[0, 1]$ intervallumba esik (1 a teljes konvexitást jelenti). Az iránymenti konvexitás mértékét csak az oszlopokra vizsgáltuk meg, mivel a sorok mindig konvexek. Az eredmények (3.2. ábra) alapján megállapítható, hogy nyilvánvalóan az összes oszlopot felhasználva lehet a legkonvexebb eredményt kapni. Viszont, ha kevesebb oszlopot használunk fel a rekonstrukció során, akkor érdemes minden iterációban véletlenszerűen vizsgálni az oszlopokat, azaz a *RANDOM* stratégiát követni (3.2a. ábra).

A 3.2b. ábrán látható, hogy az *INCREASE* stratégia teljesített legjobban a futásidők szemszögéből, míg a *RANDOM* és az *EVEN* közel ugyanúgy. Ennek oka, hogy a vizsgált

oszlopok száma és hossza határozza meg a rekonstrukció időtartalmát. A vetületi hibák és az RME esetén a stratégiák között nincs számottevő különbség (3.2c. és 3.2d. ábra).



3.2. Ábra. H-TILES_SA algoritmus teljesítményének szemléltetése különböző szempontok alapján a konvexitás mérés stratégiájától függően (ALL (◆), EVEN (■), RANDOM (▲), INCREASE (×)). Szempontok: konvexitás (a), futásidő (b), vetületi hiba (c), RME (d).

3.3. A pixeleken alapuló szimulált hűtéses rekonstrukció vizsgálata

A PIXEL_SA algoritmusnál a komponensek száma nem befolyásolja jelentős mértékben a futásidő átlagát és szórását (3.6. táblázat). A rekonstrukció pontosságának átlaga és szórása is növekszik azonban a komponensek számának növelésével (3.7. táblázat). Kevesebb képpont esetén ugyanis ritkább a mátrix, ami elősegíti a kapcsoló komponensek jelenlétét, így több megoldás is előfordulhat, mely kielégíti ugyanazokat a vetületeket. Ezért nagy a valószínűsége, hogy olyan megoldást találunk, ami nem hasonlít az eredeti képre.

Futás idők - PIXEL_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	12.75	46.88	43.48	144.46	55.06	173.68	112.89	346.80	166.18	508.01
2	15.48	61.72	50.70	169.12	66.10	209.76	115.91	358.70	166.79	511.72
3	14.88	55.03	51.66	170.54	64.16	200.31	121.02	370.91	166.63	510.87
4	14.71	57.44	51.86	173.53	69.88	218.22	124.84	382.18	169.81	519.34

3.6. táblázat. PIXEL_SA algoritmus futásidejei milliszekundumban megadva. A sorok a komponensek számát ábrázolják, az oszlopok a tesztképek méreteit.

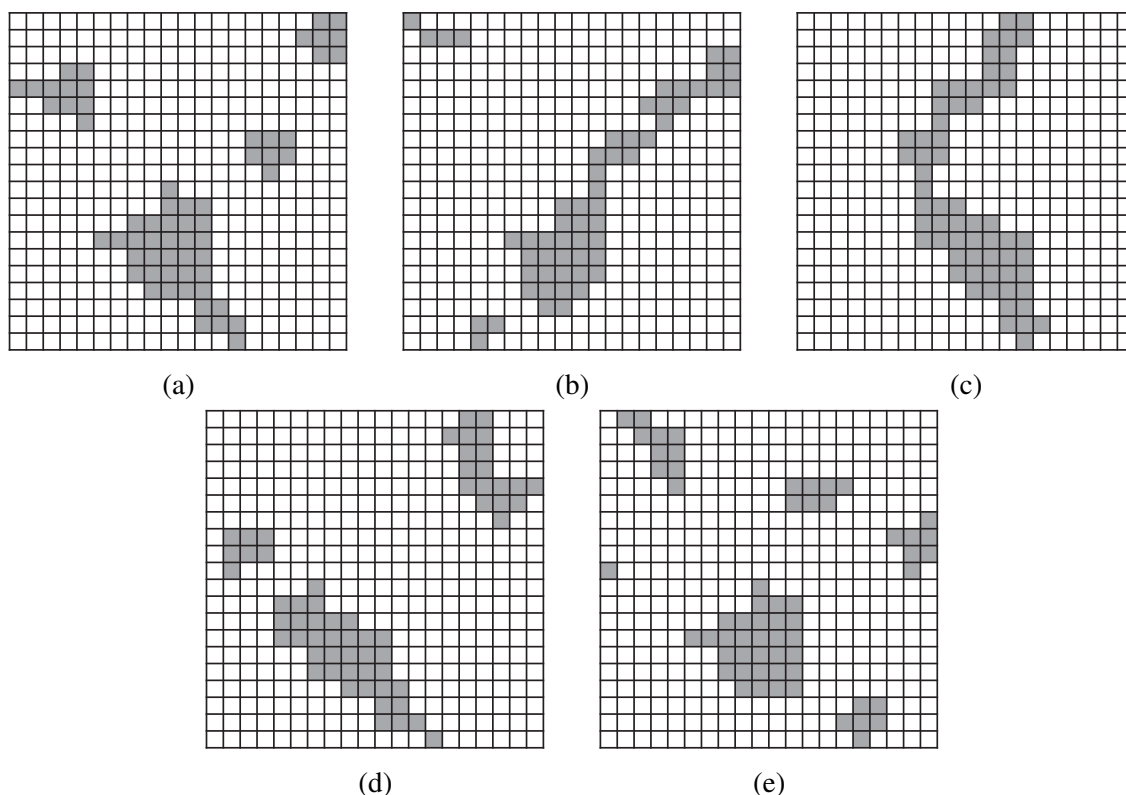
RME - PIXEL_SA										
	10 × 10		20 × 20		30 × 30		40 × 40		50 × 50	
	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás	átlag	szórás
1	0.03	0.18	0.05	0.21	0.06	0.25	0.08	0.30	0.10	0.37
2	0.05	0.26	0.08	0.32	0.09	0.34	0.13	0.43	0.15	0.49
3	0.07	0.28	0.09	0.34	0.12	0.42	0.14	0.45	0.17	0.54
4	0.09	0.34	0.11	0.40	0.13	0.44	0.15	0.47	0.17	0.54

3.7. táblázat. PIXEL_SA algoritmus RME értékei. A sorok a komponensek számát ábrázolják, az oszlopok a tesztképek méreteit.

3.4. Az eredmények összehasonlítása

Az algoritmusok eredményeit összehasonlítva megállapíthatjuk, hogy a futásidők szempontjából a H-TILES_SA algoritmus a legjobb, ha a konvexitást a szomszédos egyesek alapján határozzuk meg. A többi algoritmus sem teljesített azonban sokkal rosszabbul a futásidők alapján. A rekonstrukció pontossága közel azonos mindegyik vizsgált algoritmusnál. Viszont a H-TILES_SA algoritmusnál nem érdemes használni a szomszédos egyesek számán alapuló konvexitás mértéket, mivel gyakran előfordul, hogy a módszer egy oszlopba próbálja tömöríteni az egyeseket (3.3c. ábra). A SWITCH_SA algoritmusnál is észrevehető, hogy ezen mérték alapján az eljárás a képet különálló komponensekre bontja, viszont ezen komponensek összessége gyakran nem *hv*-konvex képet eredményez (3.3e. ábra). Mindemellett az RME értékek közel azonosak mindhárom algoritmus esetén.

További fontos szempont, hogy a SWITCH_SA algoritmus érzékeny a vetületi zajra. Amennyiben zajos lenne a vetület, akkor esetleg már a kezdeti megoldást sem tudná előállítani. Viszont a H-TILES_SA és a PIXEL_SA algoritmus elvégzi a rekonstrukciót vetületi zaj esetén is. Összegzésként megállapítható, hogy a H-TILES_SA algoritmus iránymenti konvexitást felhasználva teljesít a legjobban a vizsgált szempontok alapján.



3.3. Ábra. Eredeti kép (a). Rekonstrukciós algoritmusok eredményei: PIXEL_SA (b), H-TILES_SA (c), ahol a konvexitás mértékét a szomszédos egyesek száma határozza meg, H-TILES_SA az ALL stratégiával (d), SWITCH_SA (e).

4. fejezet

Összegzés és további tervek

A dolgozat célja két új rekonstrukciós eljárás ismertetése és elemzése volt. Megmutattuk, hogy a kapcsoló komponensek mellett kisebb (pixelek) és nagyobb (csempék) egységek segítségével is lehet alkalmazni a szimulált hűtést a rekonstrukció során. Nagy adathalmazokon végzett kísérletek eredményeit elemezve megállapítottuk, hogy az optimalizációs eljárás során, ha szükségünk van a konvexitás mértékének a megállapítására, akkor iránymenti konvexitást érdemes használnunk. A szomszédos egyesek számának mérésével ugyanis gyakran előfordulnak olyan esetek, hogy két komponens egy sorban vagy oszlopban helyezkedik el. Ezt a helyzetet az iránymenti konvexitással könnyű kezelni.

Az iránymenti konvexitás megállapítása egy viszonylag költséges művelet. Ezért különböző stratégiákat adtunk meg, melyekkel csökkentettünk a vizsgált oszlopok vagy sorok számát. Lehetséges lenne olyan stratégiát is megadni, ami az eddig leírt stratégiákhoz képest még kevesebb oszlopot vagy sort vizsgál meg, úgy, hogy az iterációk során bizonyos szempontok alapján rangsorolja, hogy melyik oszlopokat lenne a legérdekesebb megvizsgálni. Ezek alapján különböző valószínűségi értékeket rendelne ezekhez az oszlopokhoz, és a preferált oszlopok nagyobb valószínűséggel kerülnének kiválasztásra a mérés során. Szükséges lesz annak vizsgálata is, hogy zajos vetületek esetén pontosan milyen eredményeket produkál a H-TILES_SA és a PIXEL_SA algoritmus. Mindezen vizsgálatokat a közel jövőben tervezzük elvégezni.

Jelen dolgozat eredményeiből elkészítettünk egy folyóirat publikációt is, mely jelenleg bírálattal van [15].

Köszönetnyilvánítás

Jelen kutatási munkát a hallgató esetében támogatta: a Közigazgatási és Igazságügyi Hivatal által a TÁMOP 4.2.4.A/1-11-1-2012-0001 Nemzeti Kiválóság Program - Hazai hallgatói, illetve kutatói személyi támogatást biztosító rendszer kidolgozása és működtetése konvergencia program című kiemelt projekt keretében meghirdetett kutatói ösztöndíj program. A kutatási munkát a témavezető esetében támogatta: a Közigazgatási és Igazságügyi Hivatal által a TÁMOP 4.2.4.A/2-11-1-2012-0001 Nemzeti Kiválóság Program - Hazai hallgatói, illetve kutatói személyi támogatást biztosító rendszer kidolgozása és működtetése konvergencia program című kiemelt projekt keretében meghirdetett kutatói ösztöndíj program.

Irodalomjegyzék

- [1] G.T. Herman, A. Kuba (Eds.), *Discrete Tomography: Foundations, Algorithms and Applications*, Birkhäuser, Boston, 1999.
- [2] G.T. Herman, A. Kuba (Eds.), *Advances in Discrete Tomography and Its Applications*, Birkhäuser, Boston, 2007.
- [3] H. J. Ryser, Combinatorial properties of matrices of zeros and ones, *Canad. J. Math.* **9** (1957) 371–377.
- [4] E. Barcucci, A. Del Lungo, M. Nivat, R. Pinzani, Reconstructing convex polyominoes from horizontal and vertical projections, *Theor. Comput. Sci.* **155** (1996) 321–347.
- [5] S. Brunetti, A. Del Lungo, F. Del Ristoro, A. Kuba, M. Nivat, Reconstruction of 4- and 8-connected convex discrete sets from row and column projections, *Lin. Algebra Appl.* **339** (2001) 37–57.
- [6] M. Chrobak, C. Dürr, Reconstructing hv-convex polyominoes from orthogonal projections, *Inform. Process. Lett.* **69** (1999) 283–289.
- [7] G.J. Woeginger, The reconstruction of polyominoes from their orthogonal projections, *Inform. Process. Lett.* **77** (2001) 225–229.
- [8] A. Kuba, Reconstruction of two-directionally connected binary patterns from their two orthogonal projections, *Computer Vision, Graphics, and Image Proc.* **27** (1984) 249–265.
- [9] P. Balázs, Discrete tomographic reconstruction of binary images with disjoint components using shape information, *International Journal of Shape Modeling.* **14** (2008) 189–207.
- [10] F. Jarray, G. Tlig, A simulated annealing for reconstructing hv-convex binary matrices, *Discrete Appl. Math.* **36** (2010) 447–454.

- [11] Z. Ozsvár, P. Balázs, An empirical study of reconstructing *hv*-convex binary matrices from horizontal and vertical projections. *Acta Cybern.* **21(1)** (2013) 149–163.
- [12] T.S. Tasi, L.G. Nyúl, P. Balázs, Directional convexity measure for binary tomography. *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Lecture Notes in Computer Science* **8259** (2013) 9–16.
- [13] P. Balázs, A benchmark set for the reconstruction of *hv*-convex discrete sets, *Discrete Appl. Math.* **157** (2009) 3447–3456.
- [14] P. Balázs, A framework for generating some discrete sets with disjoint components by using uniform distributions, *Theoret. Comput. Sci.* **406** (2008) 15–23.
- [15] P. Balázs, Z. Ozsvár, L.G. Nyúl, A discrete convexity measure inspired by binary tomography, *Intelligent Data Analysis*, submitted.