Digitális terepmodell előállítása variációs spline interpolációval

Katona Endre

Szegedi Tudományegyetem, Képfeldolgozás és Számítógépes Grafika Tanszék
6720 Szeged, Árpád tér 2., +62/546 195, katona@inf.u-szeged.hu

1. Bevezetés

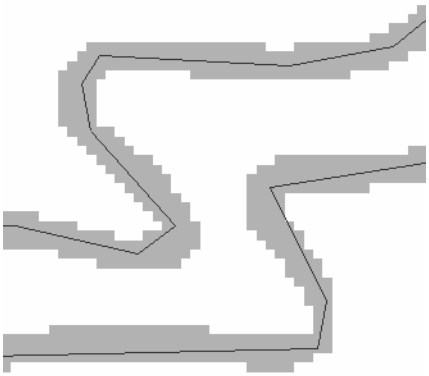
Digitális terepmodellek két fő változata, a háromszögrácsra épülő TIN (Triangulated Irregular Network) és a raszteres DEM (Digital Elevation Model) közül az utóbbival foglalkozunk. A DEM kétségtelenül nagyobb tárigénye a mai hardverlehetőségek mellett már alig jelent hátrányt, ugyanakkor a DEM könnyebb kezelhetősége és nagyobb pontossága miatt egyre szélesebb körben használatos a TIN-nel szemben.

DEM előállításához adatforrásként általában magassági ponthalmazt vagy szintvonalrajzot használnak. Tapasztalatunk szerint ez utóbbi kényesebb feladat, ezért dolgozatunkban elsősorban a *szintvonalas fedvény*ből történő DEM előállításának kérdéskörét tárgyaljuk.

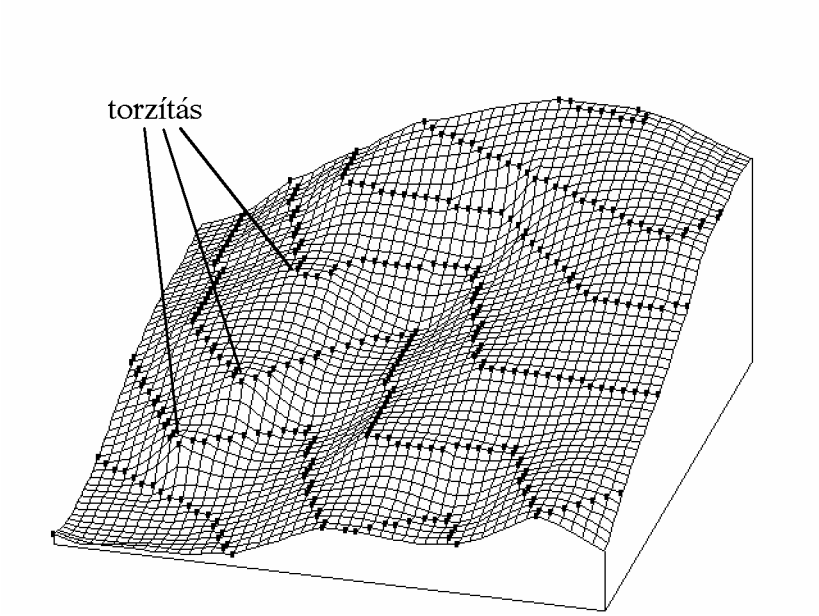
A 2. pontban a szintvonalrajz előfeldolgozására olyan, *tisztán raszteres* technológiát javasolunk, amelynek eredményeképpen egy magassági értékeket és definiálatlan pontokat tartalmazó *szintvonalmátrix* áll elő. A definiálatlan pontok értéke térbeli interpolációval nyerhető, ennek módját a 3. pont tárgyalja. Raszteres terepmodellek előállításának matematikailag leginkább megalapozott módját, a variációs spline interpolációt mutatjuk be, amelynek különféle változatait több szoftver is alkalmazza. Az eljárás lényege: olyan felületet keresünk, amely az ismert magasságú pontokra illeszkedik, és valamely energiafüggvény szerint minimális értéket vesz fel. Az optimum megtalálása lineáris egyenletrendszerre vezethető vissza, amelyet iteratív úton szoktak megoldani. A konvergencia gyorsítására az ún. multigrid módszer alkalmazható. Az alapmódszert kiegészítettünk egy *szintvonalritkító redukciós eljárás*sal, amelyet részletesen ismertetünk.

2. Szintvonalrajz feldolgozása

Mivel a terepmodellező rendszerek általában csak szintvonalakat, törésvonalakat és egyedi magasságpontokat fogadnak, így a szkennelt szintvonalrajzon valamennyi jelkulcsi elemet (alap-, fő- és felezőszintvonalak, szintvonalszámok, eséstüskék, magassági pontok, terepl épcsők, vízmosások stb.) ezekre az objektumokra kell leképezni. Az eljárás általában a szkennelt állomány manuális javításával történik, majd a szintvonalakat vektorizálják. Alább megmutatjuk, hogy vektorizálás nélkül, *tisztán raszteres technológiával* is lehet dolgozni, így nincs szükség „visszaraszterizálásra” a DEM generáláskor. Ez a megoldás maximális pontosságot biztosít, hiszen mindvégig a szkennelési felbontásban dolgozunk, és nem lépnek fel a vektorizálás okozta esetleges torzulások (1., 2. ábra). Az eljárás lépéseit alább részletezzük.



1. ábra. A vektorizáló algoritmusok ugyan garantálják, hogy a vektorok a rasztervonalon belül essenek, mégis számottevő eltérés lehetséges a középvonaltól



2. ábra. Vektorizált (és visszaraszterizált) szintvonalrajzból generált, kissé „szögletes” DEM

1. Szintvonalrajz szkennelése. Nyersanyagként az adott térképtípus szintvonalas fóliái a legalkalmasabbak, amelyek *csak* domborzati információt tartalmaznak.

2. Szintvonalrajz transzformációja a szelvénykeret négy sarokpontjának megfelelően. A szelvénykereten kívüli rész levágásra kerül.

3. Szintvonalrajz tisztítása valamely raszter-editor programmal. El kell távolítani a rajzról a zajokat és a szintvonalakon kívül minden jelkulcsi elemet.

4. Összeérések megszüntetése. Minden szintvonal-összeérés hibát okoz a továbbiakban, így ezeket kivétel nélkül meg kell szüntetni. Ha a szintvonalak annyira sűrűn helyezkednek el, hogy az adott szkennelési felbontás mellett nem választhatók szét, akkor minden másodikat meg kell szakítani.

5. Szintvonalrajz algoritmikus vékonyítása. A képfeldolgozásban használatos valamely vékonyító (vázképző) algoritmus segítségével minden szintvonalat egy pixel vastagságúra húzunk össze (Lam és tsai, 1992).

6. Szakadások megszüntetése. Szintvonal szakadások az alábbi okok miatt léphetnek fel:

– Szkennelési hiba.

– Nyomdatechnikai okok. Mivel általában a szintvonalas réteg nyomdai fóliáját dolgozzuk fel, így ezen megszakadnak a szintvonalak minden olyan helyen, ahol azokat valamely más rétegbeli elem – például út – kitakarja.

– Felező és negyedelő szintvonalak. Ezek a jelkulcs szerint szaggatott vonalak, amelyeket folytonos vonallá kell összekötni.

A szakadások összekötése szoftverrel támogatható: egy algoritmus megkeresi az egymáshoz közeli végpontokat, és felajánlja azok összekötését. (Vékonyítás után a végpontok már könnyen felismerhetők, mert 8-szomszédság mellett csak egy szintvonalpont-szomszédjuk van.) A szakadások csak csekély hibát okoznak a további feldolgozásban, ezért e műveletnél nem szükséges teljességre törekedni.

7. Összeérések ellenőrzése. A program rákeres minden megmaradt elágazási pontra (amelynek 8-szomszédság mellett 2-nél több szintvonalpont-szomszédja van). Az így kimutatott összeéréseket meg kell szüntetni. Az eredményül kapott raszteres állomány a 3. ábrán látható.

```
0  0  1  0  0  1  0  0  0  0
0  1  0  0  0  1  0  0  0  0
0  1  0  0  0  1  0  0  0  1
1  0  0  0  1  0  0  0  1  0
0  0  0  1  0  0  0  1  0  0
0  1  1  0  0  0  1  0  0  0
1  0  0  0  0  0  1  0  0  0
0  0  0  0  0  0  0  1  0  0
0  0  0  0  0  0  0  0  1  1
```

3. ábra. Szkennelt,vékonyított szintvonalrajz részlete.

```
 X   X 200  X   X 240  X   X   X   X
 X 200  X   X   X 240  X   X   X   X
 X 200  X   X   X 240  X   X   X 280
200  X   X   X 240  X   X   X 280  X
 X   X   X 240  X   X   X 280  X   X
 X 240 240  X   X   X 280  X   X   X
240  X   X   X   X   X 280  X   X   X
 X   X   X 253  X   X   X 280  X   X
 X   X   X   X   X   X   X   X 280 280
```

4. ábra. Szintvonalmátrix részlete. X definiálatlan pontot, a 253 érték egyedi magasságpontot jelöl.

8. Szintvonalmátrix előállítása. Minden szintvonalhoz magasságértéket rendelünk, amelyet a szintvonal minden pontja felvesz. A magasságérték végigfuttatása igen gyorsan végezhető valamely „connected component labelling” algoritmussal (lásd például Rosenfeld és Pfaltz 1966, Stefano és Bulgarelli 1999, Katona 2001). Ezután felvisszük (vagy egy állományból betöltjük) az egyedi magasságpontokat is (4. ábra). Tekintettel a gyakran rendhagyó domborzati viszonyokra, a szintvonalakhoz történő magasságérték-rendelés automatikusan nem oldható meg, viszont gépi támogatással jelentősen gyorsítható, például

alapszintközzel automatikusan növelt ill. csökkentett magasságértékek generálásával és csoportos értékadással.

9. Szintvonalmátrixból DEM generálása. Multigrid relaxációs módszerrel határozzuk meg a definiálatlan pontok értékét, ezzel áll elő a végső DEM, amelynek felbontása már kisebb is lehet a szintvonalmátrixénál. Az eljárást a következő fejezet részletezi.

3. Térbeli interpoláció

A legjobb minőségű terepmodellt a *vékonylemez modellre épülő multigrid eljárások* szolgáltatják, ilyen elven működik például az ANUDEM program (Hutchinson és Gallant, 2000) és az Arc/Info rendszer TOPOGRID modulja (ESRI 1994).

3.1. Variációs spline illesztés

Olyan $f(x, y)$ függvényt keresünk, amely az adott (x_1, y_1), ..., (x_m, y_m) pontokban a megfelelő d_1, ..., d_m értékeket veszi fel, és „megfelelően sima" felületet alkot (spline illesztés). A megoldás egy „energia mérőszám" bevezetése lehet: a felszín energiája annál nagyobb, minél kevésbé sima a felület. Tehát olyan függvényt keresünk, amely illeszkedik az adott pontokra, és energiája minimális.

Terzopoulos (1986) az alábbi általános r-edfokú spline energiaformulát adja:

$$E_f^r = \iint \left[\sum_{i=0}^{r} \binom{r}{i} \left(\frac{\partial^r f}{\partial x^i \partial y^{r-i}} \right)^2 \right] dx\, dy$$

Ez egy funkcionál, amely minden $f(x, y)$ függvényhez egy skalár E_f^r energiaértéket rendel. Ezen funkcionál szélsőértékét keressük, tehát variációszámítási feladattal állunk szemben. A formula két legfontosabb speciális esete:

r=1: membrán modell. Ekkor az integrál jó közelítéssel egy membrán energiáját adja:

$$E_f^1 = \iint \left(f_x^2 + f_y^2 \right) dx\, dy \qquad (1)$$

ahol f_x az x szerinti parciális deriváltfüggvényt jelöli, hasonlóan f_y. Terepmodell esetén a membrán modell megengedi, hogy a szintvonalaknál törések lépjenek fel, a zárt szintvonallal határolt kúpok pedig laposak lesznek, amennyiben magassági pont nincs hozzájuk megadva.

r=2: vékonylemez (thin plate) modell. Ekkor az integrál jó közelítéssel egy idealizált vékony acéllemez görbületi energiáját adja:

$$E_f^2 = \iint \left(f_{xx}^2 + 2 f_{xy}^2 + f_{yy}^2 \right) dx\, dy \qquad (2)$$

ahol f_{xx} az x szerinti második parciális deriváltfüggvényt jelöli, hasonlóan a többi. A vékonylemez modell nem viseli el az éles töréseket, ezért a szintvonalaknál sima átmenetet alkot, a csúcsok feldomborodnak és a mélyedések besüllyednek.

Magasabb fokú modellek már nem eredményeznek javulást a terep minőségében, ugyanakkor jelentősen növelik a számításigényt. Ezért egyértelmű, hogy *terepmodellezésre elsősorban a vékonylemez modell alkalmas*, de – amint alább látni fogjuk – bizonyos esetekben a membrán modellre is szükség van.

3.2. Törésvonalak és szakadásvonalak kezelése

Terzopoulos (1988) szerint a *törésvonalak*at egy $t(x, y)$ függvény segítségével lehet leírni, amelynek értéke 0 a törésvonalak mentén, 1 egyébként. Finomított modell esetén 0 és 1 közötti értékek is alkalmazhatók. Mivel a törésvonalak mentén a parciális deriváltak hirtelen változása megengedett, de a felszín folytonosságát megköveteljük, így ezen vonalak mentén membrán modellt, míg a többi helyen vékonylemez modellt alkalmazunk, vagyis az alábbi energiafüggvényt használjuk:

$$E_f = \iint \left\{ [1-t(x,y)]\left(f_x^2 + f_y^2\right) + t(x,y)\left(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2\right) \right\} dx\, dy$$

A *szakadásvonalak* egy $r(x, y)$ függvénnyel írhatók le, amelynek értéke 0 a vonalak mentén, 1 egyébként. Finomított modell esetén itt is alkalmazhatók 0 és 1 közötti értékek. Szakadásvonal mentén a felszín folytonossága megszűnik, ezért itt a teljes energiamodellt ki kell kapcsolni (Szeliski, 1990):

$$E_f = \iint r(x,y) \left\{ [1-t(x,y)]\left(f_x^2 + f_y^2\right) + t(x,y)\left(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2\right) \right\} dx\, dy$$

Jelen dolgozat további részében a törésvonalak és szakadásvonalak modellezésétől eltekintünk.

3.3. A variációs feladat végeselemes megoldása

A megoldás részleteit számos publikáció tartalmazza (Briggs 1974, Terzopoulos 1983, Katona 2002), itt csak az eredményeket ismertetjük. Végeselemes módszerrel dolgozunk, éspedig négyzetrács szerint felosztjuk a síkot: az $f(x, y)$ függvény helyett egy $Z[i,j]$ mátrixot veszünk, ahol $z_{i,j}$ az f függvény átlagos értéke az (i, j) négyzeten. A továbbiakban az egyes $z_{i,j}$ értékeket tekintjük változóknak, és ezek függvényeként írjuk fel az E energiaértéket. A minimum megtalálásához a $\partial E_Z^{(r)}/\partial z_{i,j}$ parciális deriváltak zérushelyeit kell megkeresni, amely egy lineáris egyenletrendszerhez vezet, ennek megoldásával kapjuk a minimális energiájú Z mátrixot. Mivel ritka mátrixú egyenletrendszerekről van szó, megoldásuk iterációs módszerrel célszerű. Jacobi-iterációt alkalmazva membrán esetén a

$$z_{i,j}' = (z_{i+1,j} + z_{i-1,j} + z_{i,j+1} + z_{i,j-1})/4 \qquad (3)$$

iterációs formula használható, amely annyit jelent, hogy a *Z* mátrixra ismételten *konvolúciót* alkalmazunk az

$$\begin{bmatrix} & 1/4 & \\ 1/4 & 0 & 1/4 \\ & 1/4 & \end{bmatrix} \tag{4}$$

maszkkal mindaddig, amíg a mátrix be nem konvergál. Membrán modell esetén azonban a konvergencia biztosításához különös gonddal kell eljárni a konvolúciós maszk megválasztásánál. Fourier-analízis (Katona 2001) segítségével kimutatható, hogy az

$$\frac{1}{32}\begin{bmatrix} & & -1 & & \\ & -2 & 8 & -2 & \\ -1 & 8 & 12 & 8 & -1 \\ & -2 & 8 & -2 & \\ & & -1 & & \end{bmatrix} \tag{5}$$

maszkkal végzett konvolúció konvergál és a vékonylemez-modellnek megfelelő megoldást adja.

3.4. Illeszkedési feltétel

Ha pontos illeszkedést követelünk meg, akkor az $f(x, y)$ függvény az adott (x_1, y_1), ..., (x_m, y_m) pontokban pontosan az előre adott d_1, ..., d_m értékeket kell, hogy felvegye. Ebben az esetben az iteráció kezdetén az adott pontok értékét a d_1, ..., d_m értékekre állítjuk be, és az iteráció során csak a többi pont értékét változtatjuk.

Bizonytalan mérési adatok esetén nem célszerű pontos illeszkedést megkövetelni, ilyenkor az eltérést a hibák négyzetösszegével szokás mérni:

$$E_d(f) = \Sigma_i \, (f(x_i,y_i) - d_i)^2$$

Az optimális felület leírására Szeliski (1990) szabályozás-elméleti alapon az

$$E(f) = E_s(f) + E_d(f)$$

egyesített energiafüggvényt használja, ahol E_s fejezi ki a *símasági feltétel*t (smoothness constraint, amelyet az eddigiekben vizsgáltunk), E_d pedig az *illeszkedési feltétel*t (data compatibility constraint). Vagyis olyan $f(x, y)$ függvényt keresünk, amelyre az egyesített energiafüggvény értéke minimális, azaz a lehető legsimább, és illeszkedik adott pontokra. Ha az egyes pontokban a mérés pontossága eltérő, ezt w_i súlyok segítségével fejezhetjük ki:

$$E_d(f) = \Sigma_i \, w_i[f(x_1,y_1) - d_i]^2$$

Fizikai modell szintjén a fentiek úgy képzelhetők el, mint ha az adott pontokhoz rugók rögzítenék a felszínt, ahol w_i a i-edik rugó erősségét fejezi ki. Végeselemes módszer esetén az $f(x, y)$ függvény helyett a $Z[i, j]$ mátrixot vesszük. Az illeszkedési energiafüggvényt felírva levezethető, hogy a relaxáció során a $z_{i,j}$ pont új értékét az iterációs maszkkal számolt érték és $d_{i,j}$ súlyozott átlagaként kell képeznünk (Katona 2001).

3.5. A szélek kezelése

A Z mátrix szélein a hiányzó szomszédok miatt az iterációs maszkot módosítani kell. Erre Terzopoulos (1988) ad javaslatot, amelynek lényege: az iterációs maszkot komponensekből építi fel, és a széleken csak azokat a komponenseket veszi figyelembe, amelyek teljes egészében a mátrixba esnek.

Saját alkalmazásunk esetén a fenti helyett egyszerűen körülkereteztük a mátrixot a szélső pixel értékek kiterjesztésével, így a szélső elemek a belsőkkel azonos módon kezelhetőkké váltak (5. ábra). Ez a megoldás programozástechnikailag egyszerűbb és gyorsabb. Membrán modell esetén a keretezéses megoldás egybeesik a komponensekből építkezővel, vékonylemez modell esetén a kettő némileg eltér, de tapasztalataink szerint ez érdemben nem befolyásolja a terepmodellt.

$$
\begin{array}{cccccccc}
z_{11} & z_{11} & z_{11} & \cdots & z_{1m} & z_{1m} & z_{1m} \\
z_{11} & z_{11} & z_{11} & \cdots & z_{1m} & z_{1m} & z_{1m} \\
z_{11} & z_{11} & z_{11} & \cdots & z_{1m} & z_{1m} & z_{1m} \\
 & & & & & & \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
 & & & & & & \\
z_{n1} & z_{n1} & z_{n1} & \cdots & z_{nm} & z_{nm} & z_{nm} \\
z_{n1} & z_{n1} & z_{n1} & \cdots & z_{nm} & z_{nm} & z_{nm} \\
z_{n1} & z_{n1} & z_{n1} & \cdots & z_{nm} & z_{nm} & z_{nm}
\end{array}
$$

5. ábra. A szélek kezelése keretezéssel

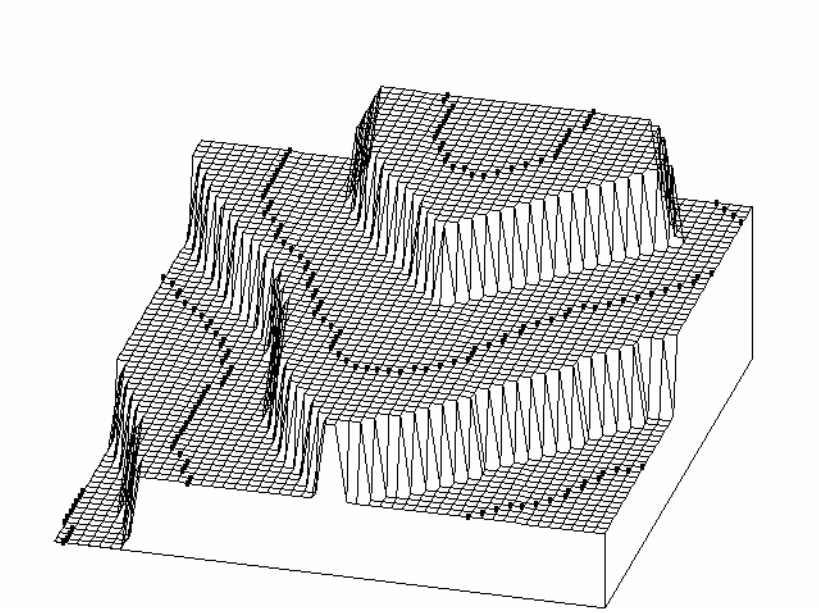
3.6. Kezdőértékadás

A számítás során használt munkamátrixok négyféle pixelt tartalmazhatnak, amelyeket egy kétbites jelzőkód különböztet meg egymástól:

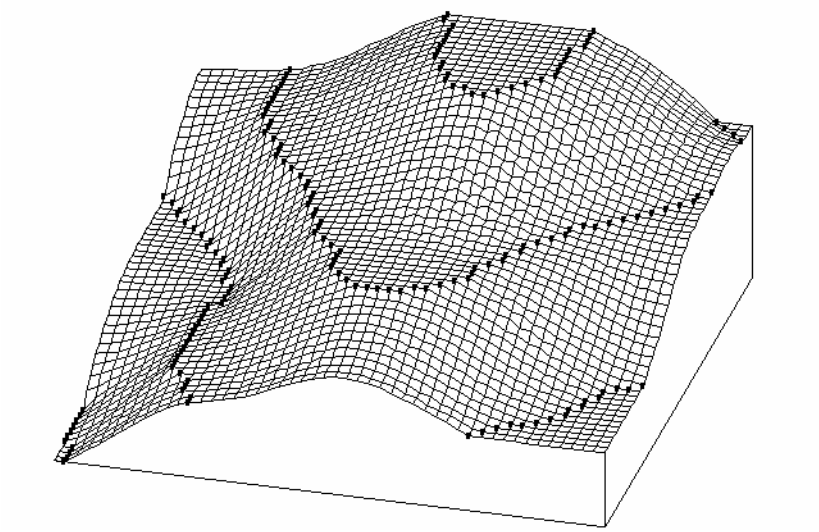
– definiálatlan pont (C0 kód),
– gyenge szintvonalpont (C1 kód),
– normál szintvonalpont (C2 kód),
– egyedi magasságpont (C3 kód).

A kezdeti Z mátrixban csak C0, C2 és C3 jelzőkódú pontok vannak, ahol az iteráció előtt a C0 pontoknak valamilyen alkalmas kezdőértéket kell adni. Szintvonalrajz esetén

kézenfekvő gondolat, hogy minden pont a hozzá legközelebbi szintvonal magasságértékét kapja. Ehhez a *Z* mátrixból egy *D* távolság-fedvényt készíthetünk, amely minden definiálatlan ponthoz a legközelebbi értékes ponttól való távolságát rendeli. Most nem valós távolságra, hanem az úgynevezett „city block distance"-re gondolunk, amely 4-szomszédság szerint lépkedve definiálja a távolságot. A távolság-fedvény lineáris időben számítható (Borgefors, 1984). Ezt az algoritmust úgy módosítjuk, hogy közben a *Z* mátrix definiálatlan értékeit feltöltjük (Katona 2001). A leírt algoritmussal a *Z* mátrixból egy *D* távolság-fedvényt és egy *Z'* kezdőérték-mátrixot állítunk elő, ez utóbbit a 6. ábra szemlélteti.



6. ábra. Kezdőértékadás után előálló terep

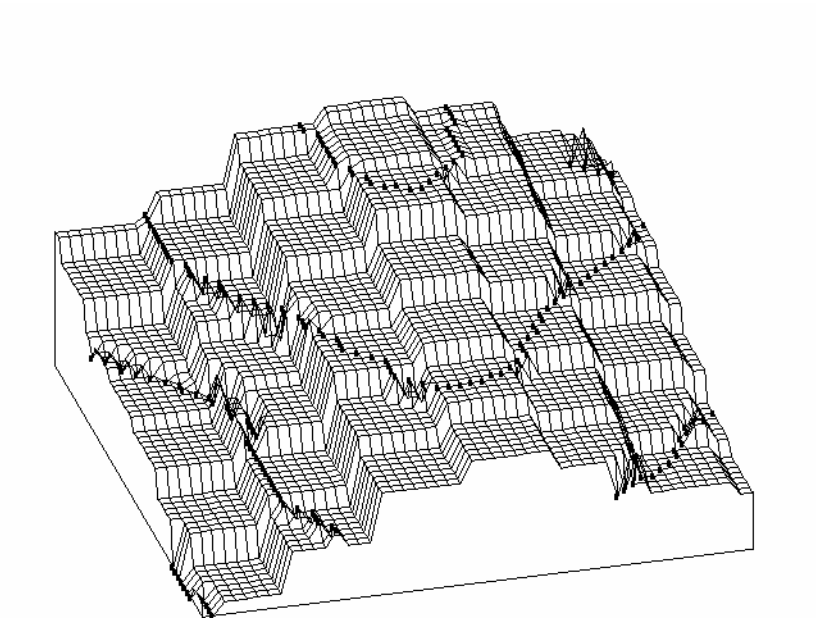


7. ábra. Membrán modell 40 iteráció után

3.7. Relaxáció (iteráció)

Az iteráció során csak a C0 jelzőkódú pontok változnak, a többit fixen tartjuk. Membrán modell esetén a konvergencia gyors, a terep már 40 iteráció után hozzávetőleg kialakul (7. ábra). Ugyanakkor a vékonylemez modell esetén több ezer iteráció lenne szükséges az elfogadható felszín kialakulásához. Terzopoulos (1983) megállapítása szerint n x n-es mátrix esetén elfogadható eredmény eléréséhez n^r iteráció szükséges, ahol r a parciális deriváltak fokszáma (vagyis membrán esetén $r = 1$, vékonylemez esetén $r = 2$). A hiba Fourier-analízise azt mutatja, hogy a nagyfrekvenciás komponensek gyorsan eltűnnek, míg a kisfrekvenciások sokáig megmaradnak. Vagyis, az iteráció a hibák gyors simítására alkalmas, de a teljes bekonvergálás igen sokáig tarthat.

Ilyen lassú konvergencia mellett a vékonylemez modell a gyakorlatban használhatatlan. A megoldást a multigrid technika jelenti, amely már 20...50 iteráció után jó eredményt ad, ezt részletezzük a továbbiakban.



8. ábra. Kezdőértékadás 8-szoros kicsinyítésű fedvényből

3.8. A multigrid elv

A megoldást a következő ötlet kínálja: készítsünk egy kicsinyített (durva felbontású) fedvényt a szintvonalpontok átlagolásával, és előbb erre végezzünk iterációt! A fent említett távoli szintvonalak most közel kerülnek egymáshoz, a konvergencia gyors lesz, és a globális hatások is jól érvényesülhetnek a kicsinyített fedvényen. Ugyanakkor a számításigény drasztikusan csökken (a mátrix kisebb méretéből adódóan is). Az így kapott kicsinyített terepmodellt használjuk fel kezdőértékként az eredeti szintvonalmátrix definiálatlan pontjaiban! Ekkor az iteráció már a végeredményhez közeli kezdőértékkel indul (8. ábra), tehát az iterációs lépések száma drasztikusan csökken.

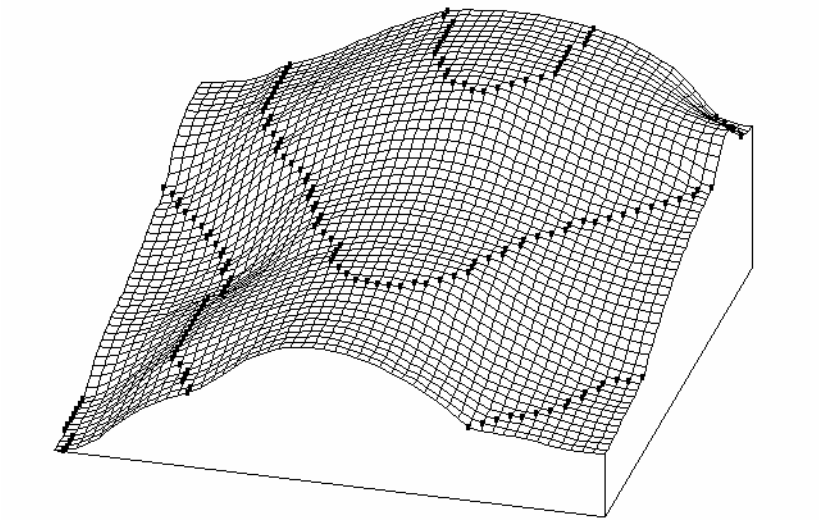
Alkalmazzuk a fenti technikát több szinten: sorra 1x1-es, 2x2-es, 4x4-es stb. kicsinyített mátrixokra iterálunk, míg el nem érjük az eredeti méretet. Így jutunk a *multigrid relaxáció* elvéhez (részletesebben lásd Katona, 2002), amelyet számos változatban alkalmaznak, például a digitális képfeldolgozásban, differenciálegyenletek megoldására, stb. (Brand, 1982). Alábbiakban saját multigrid eljárásunkat mutatjuk be, amelyet speciálisan szintvonalas fedvényekhez dolgoztunk ki.

3.9. A redukció kérdése

A redukált mátrixok előállítási módja kulcsfontosságú a módszer hatékonysága szempontjából: a szükséges iterációs lépések száma jelentősen csökkenthető, ha megfelelően érzékeny eljárást alkalmazunk a redukciónál.

Elnevezés. Ha egy n x m méretű Z mátrixot q x q méretű négyzetekre osztunk, és minden négyzetet egyetlen elemmel helyettesítünk, akkor egy n/q x m/q méretű *q-redukált R mátrix*ot kapunk.

Egyszerű redukciós eljárás. Ha egy q x q méretű négyzet tartalmazott szintvonal-ponto(ka)t, akkor azok átlaga lesz a redukált pixel értéke, és ezt adott pontnak tekintjük az iteráció során. Ha a q x q méretű négyzet nem tartalmazott szintvonalpontot, akkor a redukált pixel értéke definiálatlan lesz. Ezen kicsinyítési eljárás jó eredményt ad egyedi magasságpontok esetén. Szintvonalrajznál azonban alapvető hibája, hogy többnyire egy pixelnél vastagabb szintvonalak keletkeznek az R mátrixban, ami teraszhatást eredményez az iteráció során (folyamatos lejtő helyett a szintvonal mentén enyhébb lejtésű sáv keletkezik.



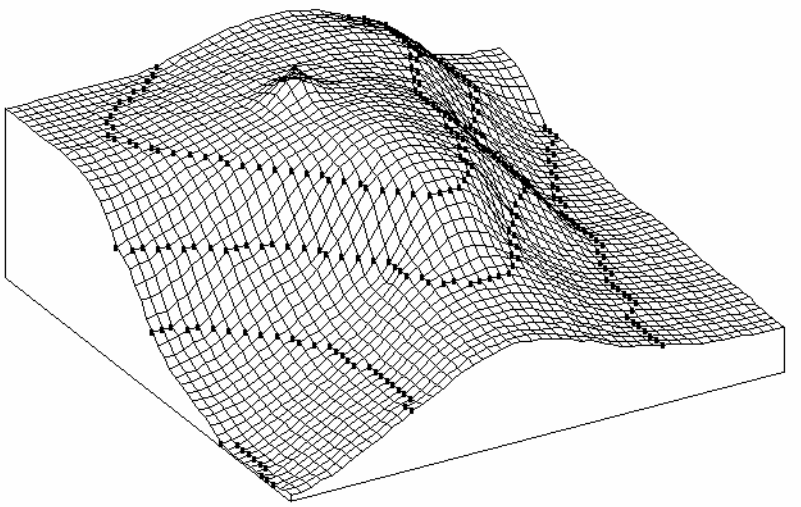
9. ábra. Küszöböléses redukcióval nyert DTM

Küszöböléses redukció. Az egyszerű redukciót annyiban módosítjuk, hogy a redukált mátrix egy pontját csak akkor tekintjük adott pontnak, ha a megfelelő q x q méretű négyzetben a szintvonalpontok száma meghalad egy adott K küszöböt. A küszöb értéke

redukciós fokozatonként változik. A küszöbölés eredményeként a redukált mátrixban a szintvonalak elvékonyodnak, és a teraszhatás minimálisra csökken (9. ábra). Ha azonban a szintvonalrajz egyedi magasságpontot is tartalmaz, az eredmény csúnya torzulással jelenik meg (10. ábra). Mindebből az következik, hogy az egyedi magasságpontok más algoritmikus kezelést igényelnek, mint a szintvonalpontok.

A küszöböléses redukció további hátránya, hogy a redukált mátrixban a szintvonalak helyenként megszakadnak. Egy-két pixelnyi szakadás még nem okoz gondot a relaxáció során, de ha olyan nagy küszöböt választunk, amely a szintvonalak vékonyságát garantálja, akkor szintvonalak akár hosszabb szakaszon is eltűnhetnek a redukciónál, ami már a terep torzulásához vezethet.

A fentiek arra utalnak, hogy a küszöböléses redukciót erősen finomítani kell a jó minőségű terepmodell előállítása érdekében. Erre a célra dolgoztuk ki a szintvonalritkító redukciós eljárást, amelyet a következő pontban ismertetünk.

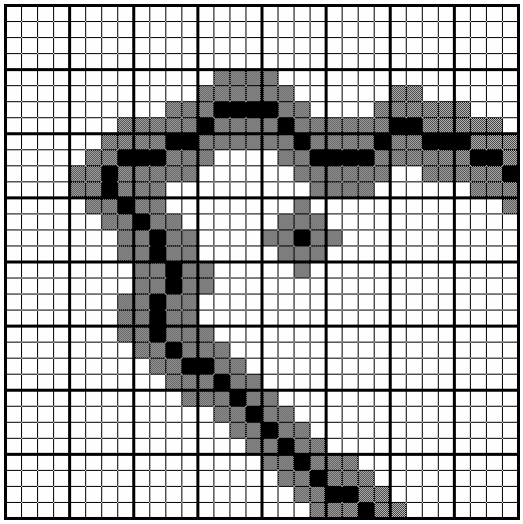


10. ábra. Torzulás a magasságpontnál küszöböléses redukció esetén

3.10. Szintvonalritkító redukció

Az eredeti Z mátrixból egy q-szorosan kicsinyített R mátrixot kívánunk előállítani. Ez azt jelenti, hogy Z-t q x q méretű négyzetekre osztjuk, és minden egyes négyzetből egy pixel keletkezik R-ben. Az eljárás lépései a következők:

1. Övezetképzés: $Z \rightarrow Z^{+}$. A redukált pixelek értékének pontosabb meghatározása érdekében Z-ből egy Z^{+} mátrixot állítunk elő $q/2$ pixeles övezetképzéssel a szintvonalak és magasságpontok körül, 4-szomszédság szerint (11. ábra). Ez lineáris időben megoldható a 3.6. pontban tárgyalt kezdőértékadó algoritmus felhasználásával és $q/2$-es küszöböléssel.



11. ábra. Övezetképzés q/2 lépésben (q = 4)

			C2	C1		C1	
	C1	C2		C1	C2	C1	C2
		C1		C3			
		C2					
		C1	C1				
			C1	C1			
				C1	C1		

Az S redukált mátrix

			C2			C1	
		C2			C2		C2
				C3			
		C2					
			C1				
				C1			
					C1		

Az R redukált mátrix

12. ábra. Jelzőkódok az S és R mátrixokban (a 11. ábra szerinti Z+ mátrixból generálva)

2. Küszöbértékek meghatározása ($0 < k_1 < k_2 \leq 1$):

– $k_1 = (q+1)/2q$. Ez a *minimális metszési arányszámot* adja: ha egy átlós szintvonal egy q x q négyzet negyedrészén halad át, akkor az övezetképzés után a négyzet $q(q+1)/2$ értékes pixelt fog tartalmazni. Ezt leosztva a négyzet területével (q^2) kapjuk k_1-et.

– $k_2 = (3q+2)/4q$. Ez a *maximális metszési arányszámot* adja: ha egy átlós szintvonal egy q x q négyzetnek éppen az átlója mentén halad keresztül, akkor az övezetképzés után a négyzet $q(3q+2)/4$ értékes pixelt fog tartalmazni. Ezt leosztva a négyzet területével (q^2) kapjuk k_2-et.

3. Kezdeti redukált mátrix előállítása: $Z^+ \rightarrow S$. Egy *S*-beli elem magasságértéke a megfelelő q x q méretű négyzetbe eső értékes pontok átlaga. Ha nincs értékes pont, a megfelelő *S*-beli elem definiálatlan lesz. Az *S*-beli *j* jelzőkód meghatározásához legyen a q x q méretű négyzetbe eső definiált pontok darabszáma *n*, ekkor:

– ha $n/q^2 < k_1$, akkor j := C0,
– ha $k_1 \le n/q^2 < k_2$, akkor j := C1,
– ha $k_2 \le n/q^2$, akkor j := C2,
– ha a q x q méretű négyzet magassági pontot tartalmaz, akkor j := C3.

4. Szintvonalpontok ritkítása: $S \rightarrow R$. Az eljárás célja: a fentiekben beállított nemnulla jelzőkódok számának csökkentése úgy, hogy ne legyenek egy pixelnél vastagabb szintvonalak, ugyanakkor szintvonal ne szakadjon meg két pixelnél hosszabb szakaszon. Ezen „ritkítás" során ügyelni kell arra, hogy a fontosabb szintvonalpontok maradjanak meg, és a kevésbé fontosak törlődjenek (12. ábra). Az eljárás:

– *R* kezdőértéke úgy képezendő *S*-ből, hogy a magasságértékeket átemeljük, de csak a C3 jelzőkódokat tartjuk meg (egyedi magasságpontok), a többit C0-ra állítjuk.
– Visszatörlés. *S*-ben törlünk minden olyan jelzőkódot, amely *R*-ben nemnulla jelzőkóddal szomszédos (4-szomszédság szerint).
– C2 jelzőkódok átírása *S*-ből *R*-be, páratlan pozíciókon. Vagyis, ha egy sakktáblát képzelünk az *S* mátrixra, csak a fehér kockákra eső C2 jelzőkódokat írjuk át.
– Visszatörlés (mint fent).
– C2 jelzőkódok átírása *S*-ből *R*-be, páros pozíciókon. Vagyis, ha egy sakktáblát képzelünk az *S* mátrixra, csak a fekete kockákra eső C2 jelzőkódokat írjuk át.
– Visszatörlés (mint fent).
– C1 jelzőkódok átírása *S*-ből *R*-be, páratlan pozíciókon.
– Visszatörlés (mint fent).
– C1 jelzőkódok átírása *S*-ből *R*-be, páros pozíciókon.

Megmutatható, hogy a fent ismertetett eljárással a redukált mátrixban nem keletkezhet egy pixelnél vastagabb szintvonal, továbbá szintvonal csak legfeljebb két pixel hosszan szakadhat meg (Katona 2001). Ez utóbbi nem befolyásolja érdemben a relaxáció eredményét, és a 10. ábrán látható torzulás megszűnik (13. ábra).

3.11. Időigény

A teljes futási idő meghatározásához összegezzük valamennyi redukciós fokozat számításának időigényét! Ha az utolsó Z_n fokozatban a relaxáció időigénye *t*, akkor a Z_{n-1}-es redukált fokozatnál $t/4$, a Z_{n-2}-es redukált fokozatnál $t/16$, és így tovább. Innen a teljes időre a

$$t + t/4 + t/16 + ... + t/4n < 4t/3$$

becslés adódik, tehát a multigrid technikánál az utolsó redukciós fokozat időigénye a meghatározó. Ugyanez vonatkozik a tárigényre is.

Tapasztalataink szerint a 32-bites fixpontos számolás pontosságban és konvergencia-sebességben is lényegében egyenértékű a 64-bites lebegőpontos aritmetikával. Fixpontos

esetben kihasználható, hogy az (5) maszk elemei egy kivétellel kettőhatványok, vagyis szorzás és osztás helyett a lényegesen gyorsabb léptetés művelet alkalmazható. Összességében *a fixpontos számolás 7-szer gyorsabb* a lebegőpontosnál.

Vizsgálataink szerint 40 iteráció már jó eredményt ad, a relaxáció folytatása érdemben már nem javítja a terepmodell minőségét. Kérdés viszont, hogy csökkenthető-e az iterációszám? A redukált mátrixok esetében nem, mivel ezek szerepe kulcsfontosságú a terep kialakításában, ugyanakkor a számításigény csekély a kisebb mátrixméretek miatt. Az utolsó fokozatban viszont már 10-20 iteráció is jó eredményt adhat, ha a szintvonalak távolsága legalább 10 pixel.

Kérdés azonban, hogy egyáltalán szükség van-e az utolsó redukciós fokozatra, amely a szkennelési felbontásban állít elő DEM-et. A forrásadatok pontosságát figyelembe véve rendszerint elegendő az utolsó előtti (esetleg még korábbi) redukciós fokozatig elmenni, ezzel a futási idő jelentősen csökken (1. táblázat).

Multigrid szint	*Pontosság*	*Tárigény*	*Futási idő*
maximum	0.08 mm	132 MB	8.4 min.
max – 1	0.17 mm	33 MB	2.2 min.
max – 2	0.33 mm	8.3 MB	35 sec.
max – 3	0.67 mm	2.1 MB	12 sec.

1. táblázat. DEM-generálás paraméterei (600 x 400 mm-es térképszelvény, szkennelés 300 dpi-vel, 40 iteráció, 667 MHz Intel processzor)

4. Értékelés, záró megjegyzések

Digitális terepmodell előállítására egy tisztán raszteres eljárást ismertettünk, amely szkennelt szintvonalrajzból és/vagy magassági pontokból raszteres terepmodellt (DEM) állít elő. Az eljárás első részében szintvonalmátrixot állítunk elő több-kevesebb manuális munkával és gépi támogatással, amely a szintvonalak mentén és magassági pontoknál ismert magasságértéket, azokon kívül definiálatlan értékeket tartalmaz. Az eljárás második része teljesen automatikus: vékonylemez modellen alapuló multigrid relaxációs módszerrel értéket adunk valamennyi definiálatlan pontnak, így áll elő a DEM mátrix.

A tisztán raszteres feldolgozás *mellett* az alábbi érvek szólnak:

– A forrásadatok (szkennelt szintvonalrajz) és a generált eredmény (DEM) egyaránt raszteres, ezért természetesebb végig raszteresen dolgozni.

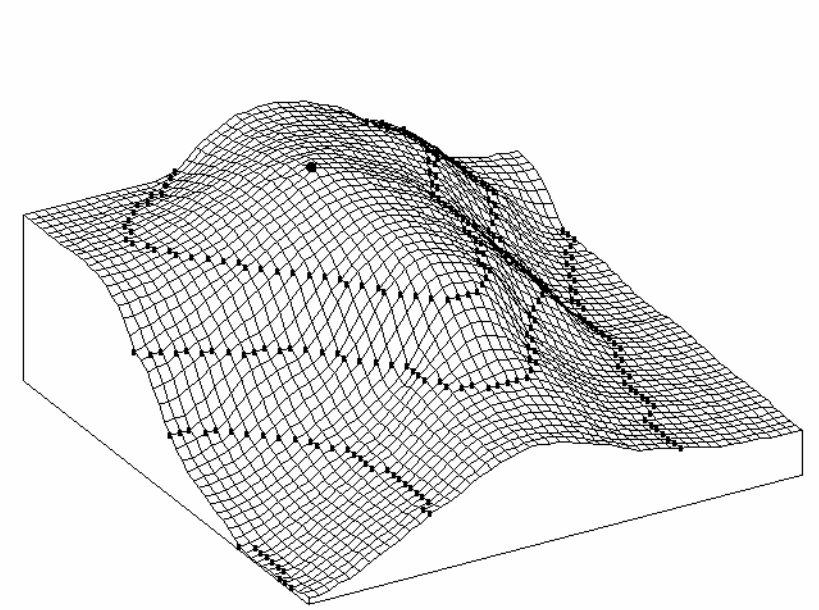
– Az eljárás pontosabb terepmodellt állít elő, hiszen elkerüljük a vektorizálás és raszterizálás okozta kisebb-nagyobb torzulásokat (2. ábra).

A tisztán raszteres feldolgozással *szemben* az alábbi ellenérveket szokták felhozni:

– A vektoros adatok hatékonyabban kezelhetők. A széles körben elterjedt vektoros szoftverek sokrétű szolgáltatásai valóban előnyt jelentenek. Ugyanakkor a 2. pontban

megmutattuk, hogy a raszteres feldolgozás is számos módon támogatható, és a manuális munka mennyisége nemigen több, mint vektoros technológia esetén.

– A raszteres feldolgozás hosszabb futási időt és nagyobb tárkapacitást igényel. Azonban, amint az 1. táblázat adatai is tanúsítják, a mai számítógépek megnövekedett teljesítménye ezt a hátrányt lényegében megszüntette.



13. ábra. Szintvonalritkító redukcióval generált terepmodell. A 10. ábrán látható torzulás megszűnik

A jelen dolgozatban bemutatott módszer egy korábbi változata gyakorlati alkalmazásra került 1991-92-ben, a *Magyar Honvédség Tóth Ágoston Térképészeti Intézete* (MH-TÁTI) által vezetett projektben (Szabó B. 1994, Bakó 1994, Katona 1995), amelynek során Magyarország egész területét lefedő, akkor legnagyobb felbontású domborzatmodelljét állították elő. Az akkori számítástechnikai eszközök színvonala miatt egyetlen szelvény feldolgozási ideje mintegy 100 óra (!) lett volna (386-os processzor, 33 MHz), ezért a Cellware Kft. által kifejlesztett gyorsítóprocesszor (Legendi és tsai 1988, Katona 1992) alkalmazására került sor. Megjegyezzük, hogy a DDM-10 projektben az (5) iterációs maszk helyett az

$$E_f^2 = \iint \left(f_{xx}^2 + f_{yy}^2 \right) dx\, dy$$

energiafüggvényből levezethető, gyorsabb számolást lehetővé tevő maszkot használtunk. A generált terepmodell gyakorlatilag nem különbözik a (5) maszkkal generálttól, amiből azt az érdekes következtetést lehet levonni, hogy az eljárás többi eleme (multigrid technika, szintvonalritkító redukció) erőteljesebben befolyásolja az eredményt, mint maga a konvolúciós maszk.

Végül megjegyezzük, hogy az ismertetett eljárás kiegészíthető a Hutchinson (1989) által kidolgozott, úgynevezett „lefolyás kikényszerítő algoritmussal”, amely a vékonylemez modell szerinti felszínt a valós vízfolyásoknak megfelelően automatikusan korrigálja, ezzel javítva a terepmodell minőségét.

Irodalom

Bakó Z. (1994): Digitális térképészeti adatbázis fejlesztések a Magyar Honvédség Kartográfiai Üzemében. *Térinformatika Magyarországon*. Az NCGIA Core Curriculum melléklete, szerk.: Márkus Béla, EFE FFFK, Székesfehérvár.

Borgefors, G. (1984): Distance Transformations in Arbitrary Dimensions. *Computer Vision, Graphics and Image Processing*, Vol. 27, pp. 321-345.

Brand, K. (1982): Multigrid bibliography. In: *Multigrid Methods*, eds: W. Hackbusch and U. Trottenberg, *Lecture Notes in Mathematics* Vol. 960., Springer-Verlag, Berlin, pp. 631-650.

Briggs, I. C. (1974): Machine contouring using minimum curvature. *Geophysics*, **39**, 39-48.

ESRI (Environmental System Research Institute, Inc.), 1994: *Arc/Info Version 7 – Arc Commands.* ESRI, New York. (Lásd még: http://www.esri.com)

Hutchinson, M. F. (1989): A new procedure for gridding elevation and stream-line data with automatic removal of spurious pits. *Journal of Hydrology*, 106, pp. 211-232.

Hutchinson, M. F., and Gallant, J. C. (2000): Digital elevation models and representation of terrain shape. In *Terrain Analysis – Principles and Applications*, edited by J. P. Wilson and J.C. Gallant (Wiley & Sons, New York), pp. 29-50.

Katona, E. (1992): Cellular processing. In *Fuzzy, Holographic and Parallel Intelligence,* edited by B. Soucek and the IRIS Group (Wiley & Sons, New York), pp.215-230.

Katona E. (1995): Digitális terepmodell számítása multigrid relaxációs eljárással. *Geodézia és Kartográfia* 1995/5, pp. 20-25.

Katona E. (2001): *Automatikus térkép interpretáció.* PhD értekezés, Szegedi Tudományegyetem.

Katona E., 2002: Digitális terepmodellek előállítása. *Polygon*, Vol. XI, No 2, pp. 35-55.

Lam L., Lee S. W., Suen C. Y. (1992): Thinning methodologies - a comprehensive survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 9, pp. 869-887.

Legendi T., Katona E., Tóth J., Zsótér A. (1988): Megacell Machine. Proc. of the Conference "VAPP III.", Liverpool, Aug., 1987, *Parallel Computing* 8, pp. 195-199.

Rosenfeld A., Pfaltz J. (1966): Sequential Operations in Digital Picture Processing. *Journal of the ACM*, 13 (4) pp. 471-494.

Stefano, L. and Bulgarelli, A. (1999): A Simple and Efficient Connected Components Labelling Algorithm. In *Proceedings of 10th Internat. Conf. on Image Anal. and Processing* (IEEE Press), pp. 322-327.

Szabó B. (1994): A katonai térképészet térképművei. Geodézia és Kartográfia, 1994/3, pp. 138-141.

Szeliski, R. (1990): Fast Surface Interpolation Using Hierarchical Basis Functions. *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 12, No. 6, pp. 513-528.

Terzopoulos, D. 1983: Multilevel Computational Processes for Visual Surface Reconstraction. *Computer Vision, Graphics and Image Processing,* Vol. 24, pp. 52-96.

Terzopoulos, D. (1986): Regularization of Inverse Visual Problems Involving Discontinuities. *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 8, No. 4, pp. 413-423.

Terzopoulos, D. (1988): The Computation of Visible-Surface Representations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, pp. 417-438.