Az automatikus raszter-vektor konverzió lehetőségéről

Elek István

2004

Bevezetés

A térkép digitalizálás régóta áhított technológiája az automatikus raszter-vektor konverzió. Számos próbálkozás történt már a probléma megoldására, nemcsak a geoinformatika, hanem egyéb grafikus alkalmazások területén is. Több-kevesebb sikert mindenki fel tudott mutatni, de kifogástalan működést egy sem (az általam ismert implementációk közül). Csak egészen speciális esetekben (pl. csak szintvonalakat tartalmazó fedvények) lehetett jó minőségűnek mondani az előállt vektoros állományt. Általában alig lettek használhatók az előállt adatproduktumok.

A legtöbb konverziós eljárás valamiféle vonalkövetést próbált megvalósítani, amikor is azonos vagy hasonló intenzitású, színű pixelek által jelölte ki a leendő vektoros állomány nyomvonalát. Az elképzelés első ránézésre akár működőképesnek is vélhető, sajnos azonban a legtöbb gyakorlati esetben nagyon rossz minőségű lett a kapott vektoros anyag, amely csak jelentős utólagos emberi beavatkozás után vált használhatóvá. Igen sok esetben kétséges volt, hogy vajon nem egyszerűbb-e a rosszul működő automatikus konverzió helyett a lassabban, de kvázi hibátlanul dolgozó emberrel végeztetni el a munkát. A válasz sokszor az emberi munka mellett döntött, különösen olyan országokban, ahol olcsón áll rendelkezésre képzett munkaerő. Fejlettebb implementációk jelentős interaktivitást is megengedtek a szoftvert működtető embernek. Ezzel mindenképpen gyorsult a vektorizálási eljárás, lényegesen javult a kapott vektoros anyag minősége, de állandó emberi jelenlétet és döntést igényelt.

A következőkben megvizsgáljuk egy nem a vonalkövetés elvén alapuló raszter-vektor konverziós eljárás elvi és gyakorlati működését. A neve *IRIS*, az angol *Intelligent Rasterimage Interpretation System* szavak kezdőbetűiből származik. Az ELTE Informatikai Karán működő Informatikai Kooperatív Kutatási Központ (IKKK) és az MTA Térképtudományi és Térinformatikai Kutató Csoportjának egyik kiemelt kutatási témája.

1.1. Elvi megfontolások

Először is vizsgáljuk meg a látás pszichológiájából származó tapasztalatokat, tekintsük át a szempontokat, amiket a térképet olvasó ember figyelembe vesz. Gaetano Kanizsa [5] óta tudjuk, hogy a szem éleket detektál, amik révén szegmentálja a nézett képet. Az élek detektálása után „szemrevételezi” az élek közötti foltokat, vagyis poligonokat értékel ki. Az éldetektálás különös esetei az éltalálkozások, sarkok, szögletek, amelyek többnyire valamilyen speciális szituációt jeleznek (intelligens, a látást szimuláló gépek esetében a szögletek alkalmasak lehetnek a tárgyak térbeli helyzetének, egymás fedésének, takarásának megállapítására, míg

térképek esetében a vonaltalálkozások szintén kritikus pontjai a térkép kiértékelésének, olvasásának).

A tapasztalat azt mutatja, hogy látásunk nagy megbízhatósággal képes kiértékelni a térképen látható vonalak és foltok rendszerét. Ennek oka két fő csoportban keresendő.

Egyrészt szemünk képfeldolgozó képessége rendkívüli. Kiváló éldetektor. Megbízhatóan szegmentál. Felületként értelmezi az élek közötti területet. Ezek a képességek - bármennyire hasznosak is - csak előkészítik a terepet a képek tényleges értelmezéséhez, például a térkép „olvasásához", egy arc, egy ujjlenyomat felismeréséhez.

Másrészt mit jelent az a kifejezés, hogy valaki „olvasni" tudja térképet? Mindenekelőtt azt, hogy ismeri a térképkészítés, a felszínábrázolás konvencióit, a térkép jelkulcsát, rendelkezik azzal az ismeretanyaggal, ami által felismeri, hogy a térképen látott helyzet milyen valóságos állapotot szimbolizál. Ha tehát egy gépet (computert) meg akarunk tanítani a térképek olvasására, akkor mindenek előtt fel kell ruháznunk a térképet olvasni képes ember tudásával. Ez egyrészt annyit jelent, hogy képessé kell tenni a gépünket a tudás tárolására, olyan elemi tudásrészek révén, amiből saját tudásunk is felépül, másrészt hatékony keresési algoritmusokkal kell, hogy felruházzuk annak érdekében, hogy gyorsan hozzáférjen a szükséges tudáshoz.

A fenti megállapításokból számos következmény ered. Biztosra vehető, hogy a tudás más kell legyen az egyes térképfajták számára (mint például kataszteri, topográfiai térképek). Más kell legyen az egyes országok térképei számára is, hiszen a jelkulcsok nem egységesek a világban, sőt különböző konvenciók létezhetnek országról országra. Elképzelhető, hogy egyes térképészeti iskolák, akár egy országon belül is, más tudásbázist igényelnek. Világosan látnunk kell, hogy csak akkor várhatunk hibátlan működést a térképolvasó gépünktől, ha az ahhoz legjobban megfelelő tudásbázis használjuk.

1.2. Az alkalmazott eljárások

Ez után az elvi áttekintés után vizsgáljuk meg a részleteket. Vegyük először az előfeldolgozásnak nevezett eljárás-csoportot.

1.2.1. Előfeldolgozás

Előfeldolgozás alatt azon eljárások gyűjteményét értjük, amit még azelőtt használunk, mielőtt a tudásbázishoz fordulnánk, vagyis mielőtt bármilyen gépi intelligenciát vetnénk be. Az előfeldolgozás két fő eljárás-csoportól áll: az egyik a

képelőkészítés, a másik a nyers vektorizálás. Terjedelmi korlátok okán csak a legfontosabb képelőkészítő eljárásokat tekintjük át.

Képelőkészítés

- Canney-féle éldetektor
- Medián szűrő
- Zaj és frekvencia szerinti szűrések (Alul és felülvágók, sávszűrők)
- Szín műveletek (szín szerinti leválogatás, színcsere, kivonás, stb.)
- Szegmentáló eljárások

Ezek közül néhányat vizsgáljunk meg egy kicsit behatóbban, hogy lássuk mekkora hatékonysággal tisztul le általuk a kezdetben igencsak változatos kép. Mielőtt áttekintenénk a legfontosabb képelőkészítő eljárásokat, foglaljuk össze a konvolúció fogalmát, amely a képfeldolgozásban nagyon fontos szerepet játszik.

Konvolúció

Az egyszerűség kedvéért nézzünk először csak az egydimenziós esetet. Legyenek f_1 és f_2 folytonos függvények. Jelölje konvolúciójukat $h = f_1 * f_2$, melyet a következő kifejezés definiál:

$$h(t) = f_1(t) * f_2(t) = \int_{-\infty}^{\infty} f_1(\tau) f_2(t-\tau) d\tau$$

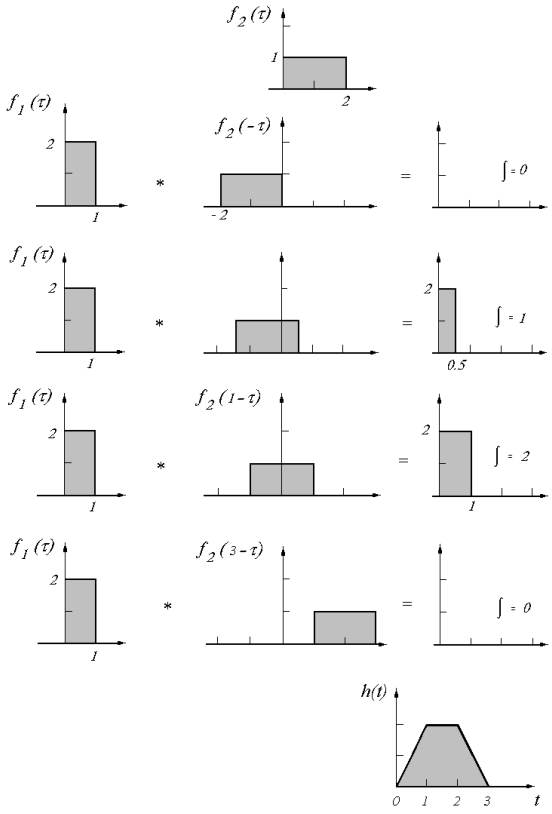
Az 1.1. ábra grafikusan szemlélteti két négyszög függvény konvolúcióját.

Digitális jelekre alkalmazva az összefüggést:

$$h(t) = f_1(t) * f_2(t) = \sum_{\tau=-\infty}^{\infty} f_1(\tau) f_2(t-\tau)$$

Vizsgáljuk meg egy konkrét esetet: Legyen $h(t)$ a t-edik pillanatban az f_1 és f_2 függvények konvolúciója, amit úgy kapunk, hogy az f_1 t-edik pillanatban felvett értékét összeszorzunk az f_2 $(t-\tau)$-edik értékével, majd végigfutunk f_2 egész intervallumán (ami valóságos esetekben véges intervallum, jelen esetben M) és összegezzük a szorzatokat (futó összegzés). A folyamatot a 1.2. ábra szemlélteti.

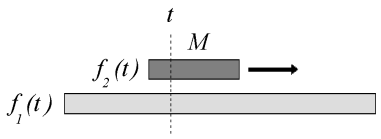
Az eddigiekben csak egydimenziós függvényekkel foglalkoztunk. Könnyen általánosítható a konvolúció fogalma kétdimenziós függvényekre is, mint amilyen a digitális kép.



1.1. ábra. Két négyszög függvény konvolúciója az időtartományban

$$h(x,y) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} f_1(u,v) f_2(x-u, y-v)$$

A digitális szűrési módszerek egyik legfontosabb fogalma a kernel. Jelentése mag. A szűrési eljárások, amikor az időtartományban dolgoznak, a kernellel konvolválják a szűrendő képet. A szűrés hatása attól függ, hogy milyen függvény értékeit tesszük be a kernelbe, ami egy $n \times n$ méretű táblázat (n a szűrő hossza). Ha meg tudjuk adni, hogy milyen átviteli függvényt kívánunk megvalósítani a frekvencia tartományban, akkor annak inverz Fourier-transzformálásával megkapjuk az időtartománybeli függvényt, amelyet megfelelően mintavételezve megkapjuk a szűrőegyütthatókat, vagyis a kernelbe töltendő számokat. A digitális konvolúció tehát a szűrések végrehajtásának egyik lehetséges módja. Ebben az esetben a szűrést az időtartományban végezzük a következő módon:



1.2. ábra. A konvolúció szemléletes jelentése

$$g'(t) = g(t) * s(t)$$

ahol $g(t)$ az eredeti adatrendszer az időtartományban, $g'(t)$ a szűrt adatrendszer és $s(t)$ a kernel.

Egy másik lehetséges megoldás, hogy a szűrendő adatrendszert Fourier-transzformáljuk, majd a frekvencia tartományban végezzük el a szűrést (a Fourier-transzformáltat megszorozzuk a kívánt hatást biztosító átviteli függvénnyel), majd az így kapott spektrumot inverz Fourier-transzformáljuk.

$$G(f) = \mathcal{F}\{g(t)\}$$

$$G'(f) = G(f)S(f)$$

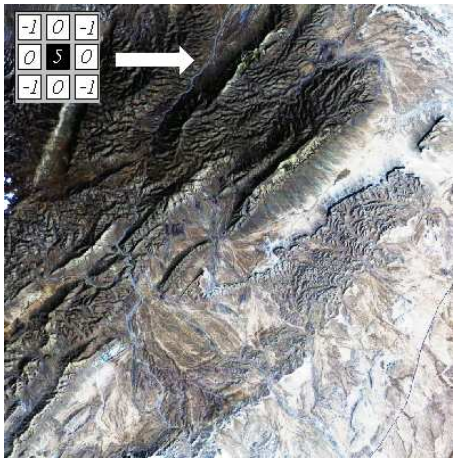
$$g'(t) = \mathcal{F}^{-1}\{G'(f)\}$$

ahol $g(t)$ az eredeti adatrendszer, $G(f)$ az adatrendszer Fourier-transzformáltja, $S(f)$ a kívánt átviteli függvény, $G'(f)$ a szűrt adatrendszer a frekvencia tartományban, $g'(t)$ a szűrt adatrendszer az időtartományban, $\mathcal{F}$ a direkt, és $\mathcal{F}^{-1}$ az inverz Fourier-transzformációt szimbolizálja.

A kernel megállapítása nemcsak a frekvencia szerinti szűrők esetén játszik kulcs fontosságú szerepet, hanem más egyéb esetekben is, mint például az élmegőrző, élkiemelő szűrők. Az elérendő cél néha olyan, hogy nem adható meg egy egyszerű átviteli függvénnyel a művelet, hiszen pontról pontra változhat az algoritmus által előírt tennivaló. Ilyenek az éldetektorok, élmegőrzők, a kép deriválók, stb.

Nézzük meg részletesen, hogy mi is történik a kernel és a kép konvolúciójakor. Vegyünk például egy 3×3 pixel méretű kernelt (1.3. ábra). Egyelőre fogadjuk el, hogy valamilyen számokkal fel van töltve a kernel. A kernellel pixelenként végigfutunk a képen a 1.3. ábrán látható módon.

Helyezzük rá a kernelt a képre (mondjuk a bal felső sarokba). A sötét mező nyilván valamelyik pixelre fog esni. A kernel közepén lévő sötét mező kitüntetett szerepű, mivel a kernel hatása mindig arra a pixelre vonatkozik, ami fölött a kernel



1.3. ábra. Az erősen felnagyított kernel mozgása a képen

középpontja áll. A kernel hatása a sötét mező alatti pixelre a következőképpen állapítható meg:

1. A kernel első mezőjében lévő számot szorozzuk meg az alatta lévő pixel színével (RGB szín), majd tároljuk az eredményt.

2. Vegyük a kernel második mezőjében lévő számot és szorozzuk meg az alatta lévő pixel színével, majd az így kapott számot adjuk hozzá az előző eredményhez. $\cdots$

   $\vdots$

3. Vegyük a kernel ötödik (sötét) mezőjében lévő számot és szorozzuk meg az alatta lévő pixel színével, majd az így kapott számot adjuk hozzá az előzőek eredményéhez. $\cdots$

   $\vdots$

4. $\cdots$

Minden szorzatot hozzáadtunk az előzőhöz, majd az összeget elosztottuk 9-cel (vagy annyival, ahány elemű a kernel), ami egyébként a szűrés eredménye is, vagyis a sötét mező alatti pixel szűrt értéke. A művelet eredménye csak egyetlen pixelnek a szűrt színe. Az egész kép szűréséhez mindezt annyiszor kell végrehajtani, ahány pixeles a kép.

Canney-féle éldetektor

Az éldetektálás különösen fontos szerepet játszik az alakfelismerésben, a raszteres térképek vektorossá alakításában. Az élek a képnek azon helyei, ahol az intenzitás megváltozása a legnagyobb. Először is döntsük el, hogy mennyire kifinomult élek kimutatását szeretnénk. A legtöbbször érdemes simító vagy medián szűrésnek alávetni a képet, hogy ne mutassunk ki minden apró, jelentéktelen élt. Egyik ismert és egyszerű módja a simításnak a kép és egy Gauss-függvény konvolúciója:

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

Legyen h az f és g függvények konvolúciója. Kimutatható, hogy

$$h = (f * g)' = f * g'$$

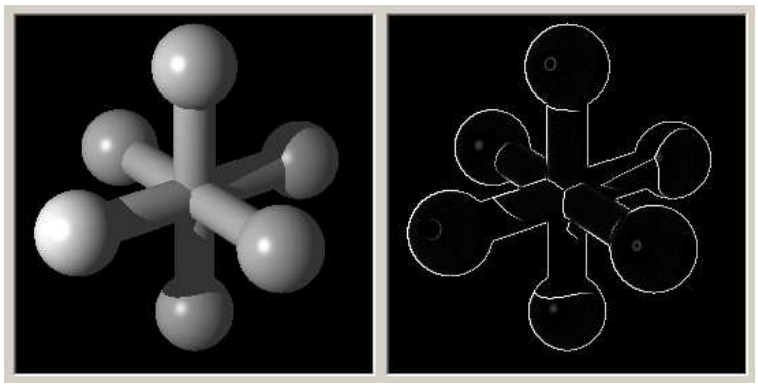
vagyis egy jel (jelöljük f-el) Gauss-függvénnyel (g) való konvolúciójának a deriváltja egyenlő a jel és a Gauss-függvény deriváltjának a konvolúciójával. Ezek alapján az éldetektálás algoritmusa a következő:

1. Konvolváljuk f-t g'-vel
2. Számítsuk ki h abszolút értékét
3. Definiáljuk éleknek mindazokat a helyeket, ahol a h abszolút értéke meghalad egy előre meghatározott küszöb értéket.

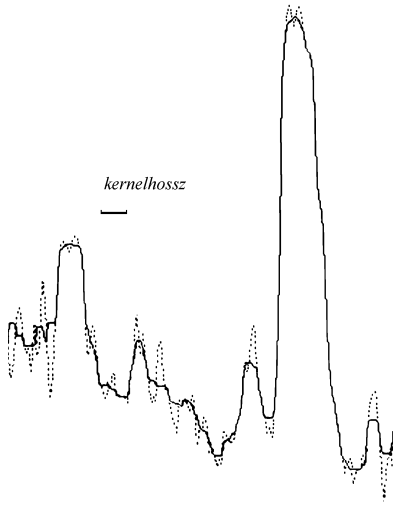
Nem használtuk ki sehol a gondolatmenet során, hogy egy vagy kétdimenziós esettel van-e dolgunk, így az éldetektálás fenti módja képek esetére is működőképes. Ez az eljárás a Canny-féle éldetektor. Eredménye a 1.4. ábrán látható egy szintetkus test példáján.

Medián-szűrő

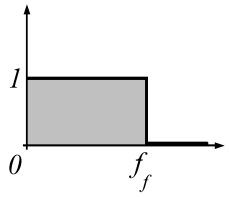
Az élmegőrző rangszűrők olyan speciális szűrők, amelyek átviteli függvényei nem adhatók meg. Működésük meglehetősen egyszerű algoritmus szerint történik. A kernelt mozgassuk végig a képen, és töltsük fel az éppen alatta lévő pixelek értékeivel. Rendezzük nagyság szerint sorba a kernel elemeit, és a rendezett adatsor valamelyik elemét rendeljük hozzá a kernel szimmetria középpontja alatt lévő pixelhez, amelynek ez lesz a szűrt értéke. Ezek a szűrők az úgynevezett rangszűrők. Az egyik legismertebb rangszűrő a medián szűrő, amely a sorba rendezett értékek sorban középső elemének értékét rendelik a pixel szűrt értékének. Az 1.5. ábrán egy idősorra alkalmaztuk a medián szűrőt.



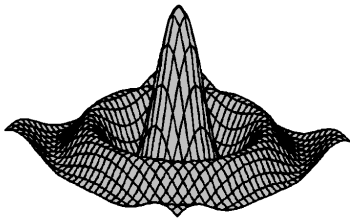
1.4. ábra. Az éldetektálás tárgya (bal oldali ábra) és eredménye (a derivált, jobb oldali ábra). A jobb oldali ábra olyan, mintha vektoros lenne, pedig nem az. Egy végletekig letisztult kép deriválása révén kapott képre már a nyers vektorizálás is jó eredményt adhat.



1.5. ábra. Egy egydimenziós függvény (szaggatott vonal) és medián-szűrt változata (folytonos vonal)



1.6. ábra. Az ideális felülvágás átviteli függvénye



1.7. ábra. Az ideális felülvágás kernel függvénye az időtartományban két dimenzióban (ami az átviteli függvény inverz Fourier-transzformáltja)

Jól megfigyelhető, hogy a fel- vagy lefutó éleken a szűrő nem változtatja meg az eredeti adatokat, hiszen azok az éleken már eleve nagyság szerint rendezettek. Nem éleken azonban erőteljesen simít. A simítás mértéke a kernel hosszától függ, annál jobban simít, minél hosszabb. E tulajdonsága miatt hatékony zajcsökkentő hatása is van.

Alul és felülvágó szűrők

Akkor használunk felülvágó szűrőt, amikor a frekvencia tartományban egy bizonyos felső határfrekvenciánál (f_f) nagyobb frekvenciákat 0-val szorzunk, és a nála kisebbeket 1-gyel. A 1.6. ábra mutatja az ideális felülvágás átviteli függvényét.

Ami a frekvencia tartományban szorzás, az az időtartományban konvolúció, vagyis a jel időtartományban végrehajtott szűréséhez a négyszög függvény inverz Fourier-transzformáltját kell használnunk a konvolúcióhoz, amit két dimenziós esetre, mint amilyen a digitális kép, a 1.7. ábrán láthatunk.

Az alulvágás teljesen hasonló a felülvágáshoz, csak a két átviteli függvény összege 1 (azonos határfrekvenciára)

$$S(f_a) = 1 - S(f_f)$$

Kép szín leválogató, cserélő, kivonó

Nagyon lényeges előkészítő funkció a kép megadott színű pixeleinek leválogatását lehetővé tevő eljárás. Ezzel levehetjük a képről ezeket a pixeleket, és elmenthetjük egy másik képben további feldolgozás céljára, mint például karakter felismerés. Ilyenkor a „üresen" maradt pixeleket a környezetük színével helyettesítjük. Hasonlóan hasznos lehet, ha a leválogatott pixelek színét más színre állítjuk be.

Szegmentáló eljárások

A papírtérképek szkennelését minimum 24 bites mélységben végezzük, ezért a keletkezett állomány az eredeti papírnyomat színeinél sokkal többet tartalmaz. Alacsonyabb színmélységű szkennelés eredménye nem felel meg a céljainknak az operációs rendszer által alkalmazott színkódtáblák miatt. Ezért az eredeti kép színmélységét meg kell hagynunk, de csak annyi színállapotot engedhetünk meg, ahányat az eredeti nyomat készítői rá kívántak vinni a térképre. Ezért szegmentáló, csoportosító eljárásoknak is alá kell vetnünk a képeket.

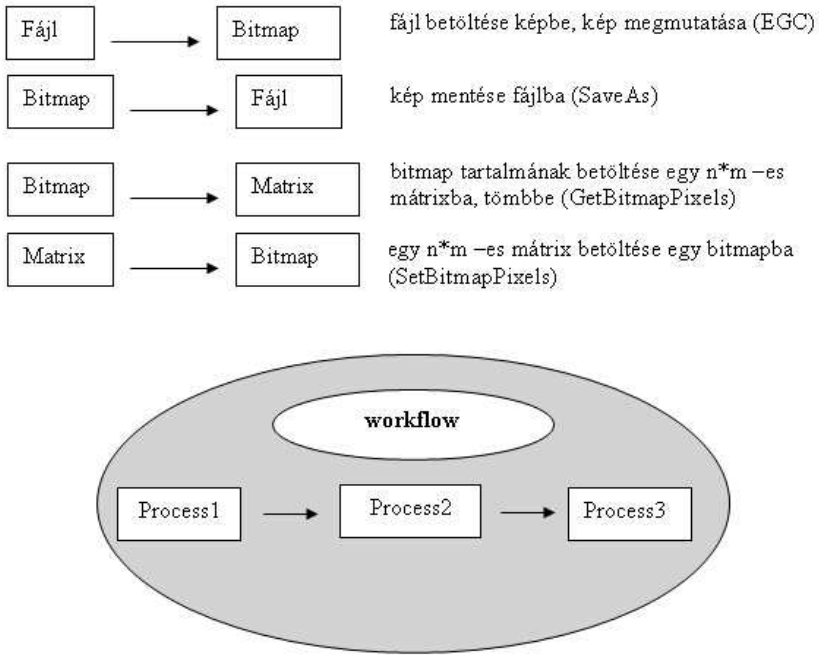
A szegmentálás óriási témakör. Léteznek automatikus, statisztikai ismérvek alapján működő eljárások (pl. klaszter analízis), de a mi céljainkhoz ezeken kívül szükségünk van egy olyan eljárásra is, amely előre megadott számú szín-csoportba sorolja be a kép különböző pixeleit, de megtartja a 24 bites színmodell előnyeit. Egy térképész bármikor megmondja, hogy egy térképen hányféle szín található, és ennek megfelelően végezhető el a raszteres állomány színeinek átdolgozása.

1.2.2. Processzek és workflowk

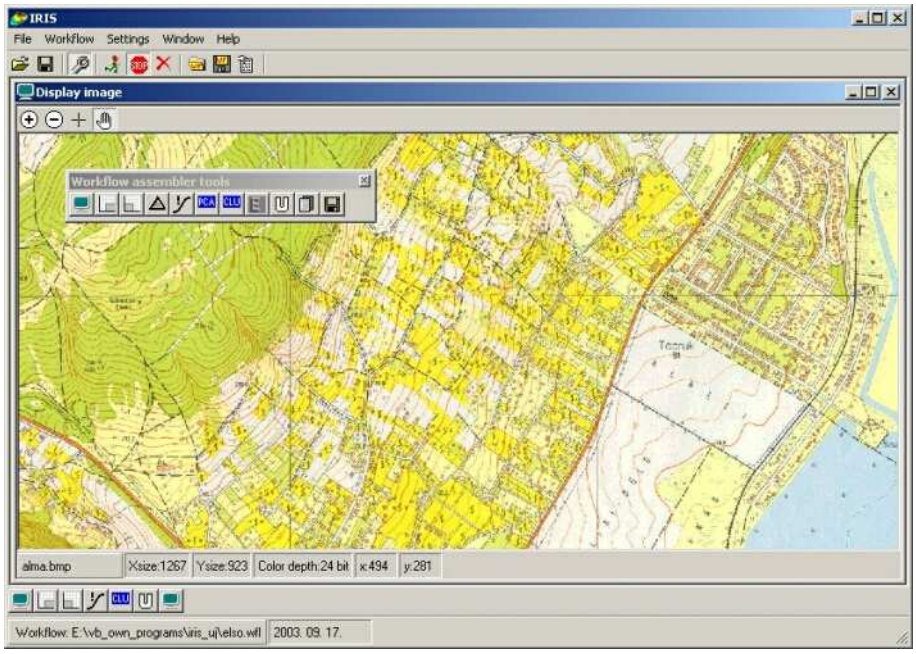
A felsorolt eljárások csak kiragadott példák a fontosabb eljárások közül. Felvetődhet a kérdés, hogy ebben az eljárás dömpingben honnan fogja tudni a raszter-vektor konverziót végrehajtani kívánó felhasználó, hogy neki melyik eljárásra van szüksége, a lehetséges néhányszor tíz közül.

Ennek megkönnyítésére vezessünk be két fogalmat. Az egyik legyen az elemi process fogalma (1.8. ábra). Ezen elemi processzek legyenek azok a képmanipuláló eljárások, amelyekből válogathat a tapasztalt felhasználó, sőt maga is létrehozhat egy általa jónak tartott eljárást (plug-in).

A másik fogalom a workflow (1.8. ábra), amely több egymásba kapcsolódó, egymás után végrehajtandó elemi processz láncolatából áll. Egy cél elérése érdekében, (mint például, simított, zajszűrt kép) néhány elemi processzből álló workflow egy kényelmes eszközzé válhat. Így névvel hivatkozható, elmenthető, komplex eljárásokat kapunk. Tetszőleges workflowt állíthatunk elő az elemi processzekből attól függően, hogy mely folyamatok támogatják a leghatékonyabban az általunk elérni kívánt célt. Ez a tény azért lehet előnyös, mert egyrészt a



1.8. ábra. A workflow több elemi processzből épül fel, amelyek egymás után hajtódnak végre. Az egyik kimenete az utána következő bemenete lesz.



1.9. ábra. Az ábra közepétől balra láthatjuk az elemi processzek kollekcióját, amelyből összeállíthatjuk a workflowkat. Az ábra bal alsó sarkában egy már létező workflow elemi processzeinek grafikus szimbólumait láthatjuk. Ebből láthatjuk, hogy mit fog csinálni a workflow (megjeleníti az eredeti képet, alulvág, felülvág, medián szűr, szegmentál, megjeleníti az eredményt)

felhasználó, a raszter-vektor konverziót végző szakember már kész workflowkat kaphat, másrészt maga is előállíthat az eddigiektől eltérő képességű workflowkat, amivel saját tudását is képes már az előfeldolgozó eljárások során beépíteni a konverziós eljárásba.

Az IRIS rendszer workflow editorának képét láthatjuk a 1.9. ábrán egy vektorizálásra váró raszteres állománnyal a háttérben.

1.2.3. Nyers vektorizálás

Miután hatékony előfeldolgozó eljárásokkal előállítottunk ideális állományokat a vektorizáláshoz, megpróbálkozhatunk a konverzió első ütemével. Mint ahogy a bevezetőben említettük a vonalkövetés nem hatékony eszköz, ezért eleve a felültekre fogunk koncentrálni. Minden poligon lesz a nyers vektorizálás után. Az eljárásunkat nevezzük poligon-növesztésnek, amely következőképpen működik: induljunk ki a kép egy sarokpontjából. Vonjuk össze egy poligonná az összes azonos optikai állapotú (intenzitású, színű) pixelt, amelyek szomszédosak. Az azonos

optikai állapotú, de diszjunkt pixelek új poligont eredményezzenek. Mindaddig növelünk egy poligont, amíg el nem fogynak az azonos optikai állapotú, érintkező pixelek. A folyamat eredményeképpen egy hézag és átfedésmentes topológiájú poligon struktúrát mutató vektoros állományt fogunk kapni, amin már minden vektorosan van rajta, ami a raszteres térképen rajta volt, de az adatstruktúra még nem célszerű, mivel sok objektum a természetétől teljesen idegen módon található meg a térképen (pl. a poligonok belsejéből hiányoznak a jelkulcs által kitakart elemek, sőt a jelkulcs elemek is poligonként látszanak, ami természetesen megszüntetendő anomália. Ezek utólagos orvoslása a következő feladat.

Itt ér véget az előfeldolgozás folyamata. Nem lehet további, „buta" eljárásokkal javítani a vektor állomány minőségén, főként a struktúráján. A további javítások már intelligenciát igényelnek, vagy emberit, vagy gépit.

1.3. Értelmezés

A másik fő feldolgozási fázis az értelmezés. Ebben már a térképolvasási képességek jutnak szerephez. A műveletek bemenete az előfeldolgozási fázis kimenete, vagyis a lehető legjobban előkészített kép alapján végzett nyers vektoros állomány.

1.3.1. A tudás reprezentációja

A tudás reprezentációja két tudásfajtát feltételez: egyrészt a jelkulcsban megbúvó tudást, másrészt a konvenciókat. A jelkulcsi elemek az alakfelismerés (pattern recognition) eredményei által válhatnak hozzáférhetővé, mivel a nyers állományon a jelkulcsban szereplő elemeket keressük. Amikor olyan mintázatot találunk, amely megfelel egy jelkulcsi elemnek, akkor a képen lévő poligont kitöröljük, és egy pontszerű jelkulcsi elemmel helyettesítjük.

A vonalelemek esetében a helyzet egyszerűbb, mert a vonal poligonként (vékony poligonként) szerepel a nyers vektoros állományban, de a térképen látható színével, így tehát eleve helyes lesz az optikai megjelenése (struktúrája még nem). Ismert szöveg-felismerési probléma, hogy az OCR (*Optical Character Recognition*) szoftverek megvadulnak, ha a felismerendő szövegrészen áthúzások, vonalak mennek keresztül. Ezzel a jelenséggel térképek esetében is szembe kell néznünk, hiszen gyakran előfordul - főként topográfiai térképeken - hogy vonalak látszanak a megírások alatt. A probléma megelőzése céljából az előfeldolgozás során meg kell kísérelni a szöveget tartalmazó pixelek leválasztását a raszteres állományról (szín leválogató processz). Ha a szöveg színe eltér a többi térképi elemtől, akkor ez nehézség nélkül megtehető. Kevésbé szerencsés esetben, amikor nem szöveges objektumok is ugyanolyan színnel szerepelnek a térképen, is van esély a szétvá-

lasztásra. A szövegek cellákba rendezettek, a vonalak tetszőleges irányultságuak, vagyis a két objektumféleség eltérő habitusa alapján lehetséges a szétválasztás.

A konvenciók figyelembe vétele a következő példában látható módon lehetséges. Tegyük fel, hogy egy folyó középvonalán halad egy megye, egy nemzeti park határa és egy környezetvédelmi felügyelőség illetékességi területének a határa is. Ismert konvenció, hogy ilyen esetben nem rajzoljuk egymásra a három poligon határt, mert az túlzsúfolná a térképet, és ezzel rontaná az olvashatóságot, hanem megszakítjuk az egymásra következő poligonok határát mutató vonalat a közös szakaszon. A térképolvasó ember tudja, hogy a vonalak megszakadása ellenére ott egy poligon határa halad, pontosan az alatta lévő poligon határán. Ez a tudásfajta, mint konvenció, átadható, ráadásul a feldolgozás elején amúgy is definiálnunk kell, hogy az egyes objektumok a végső eredményben milyen geometriai típusúak (poligon, pont, vonal) legyenek.

Tervezés

A felsorolt néhány példából is látszik, hogy a konverziós folyamatot alapos tervezési munka kell, hogy megelőzze, ami persze nem meglepő a térinformatikai rendszerek építésében jártas szakemberek számára. Előre definiálnunk kell a vektorizálás során keletkező objektumcsoportokat. Ezek paramétereit be kell állítanunk (pont, vonal, poligon, megírás), a térképen való megjelenés attribútumait (jelkulcsi elem hivatkozás, vonaltípus, kitöltési mintázat, szín, stb.), a térkép fő típusát (pl. kataszteri, topográfiai, közmű), vagy bármely, ma még nem ismert, a felismerést javító paramétert.

1.4. Ami egyelőre nem várható az automatikus raszter-vektor konverziós eljárástól

Intelligensnek mondja a szakmai zsargon azokat az objektumokat, amelyek valamely olyan azonosítóval rendelkeznek, amely külső, például alfanumerikus adatok hozzákapcsolását is lehetővé teszik (például helyrajzi szám, településnév, stb.). Még ha sikerrel fel is tudjuk ismertetni a térképen látható neveket, feliratokat, annak automatikus eldöntése, hogy mely objektumra vonatkoznak, nem része a jelenlegi terveinknek, noha a térképészeti konvenciók ismeretanyagába elvben bevihetőnek látszanak ezirányú ismeretek is. Célkitűzéseink között egyelőre csak annyi szerepel, hogy a feliratok, mint karakterek leképeződjenek egy a beszúrási pontjukhoz rendelt pontszerű objektum attribútum adataként.

1.5. Előfeldolgozó workflowk és tudásbázisok tervezett előállítása

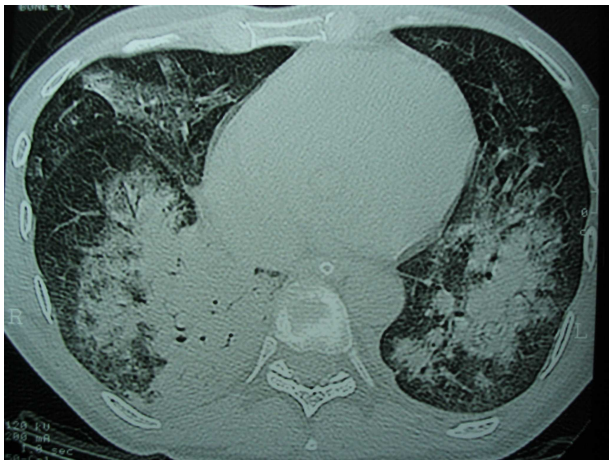
Egyelőre két térképféleség vektorizálásához állítunk elő (és még tervezünk előállítani) workflowkat és tudásbázist: a kataszteri térképekhez és az 1:10.000 topográfiai térképekhez. Az előzetes vizsgálatokból megállapítható, hogy a kataszteri térképek a látszat ellenére nem tűnnek sokkal egyszerűbb vizsgálati terepnek, mivel fekete-fehér mivoltuk miatt a színek nem segítik a feldolgozást. Egyéb szempontból viszont lényegesen egyszerűbb a kataszteri térképek vektorizálása. Itt nyilván nincs szerepe a színmanipuláló processzeknek, viszont annál nagyobb jelentőségű a zajszűrés.

1.6. Az IRIS csomag alkalmazása űrfelvételekre

A bemutatott kísérleti technológiát alapvetően papírtérképek vektorizálására használtuk eddig annak minden hibájával és gyermekbetegségével, és tervezzük használni a jövőben is. Belátható, hogy a műholdak által létrejött raszteres állományok esetleges vektorizálása nagyságrendekkel nehezebb feladat elé állítana bennünket, mivel részletgazdagságuk rendkívüli, és ráadásul semmiféle értelmezésen, generalizáláson nem estek át, mint a vektor térképek, amelyek a valóságnak az emberi intelligencia szűrőjén átment nagyfokú absztrakciói.

Egy dolgot világosan látnunk kell. Az, hogy mit látunk egy képen, nagyban függ az előképzettségünktől, az elvárásainktól, a háttértudásunktól, tapasztalatainktól, kultúránktól, előéletünktől.

Záró példám egy orvosi kép lesz, a mellkas egy computer tomográf által készített leképezése (1.10. ábra). Ez a kép számomra nem mond semmit, de a feleségem számára, aki évtizedes gyakorlattal rendelkező orvos, egy nyitott könyv, amelyből feltárul a mellkas anatómiája, és esetleges anomáliái.



1.10. ábra. A mellkas computer tomográfos képe, amelynek értelmezése kizárólag több éves tanulás útján megszerzett háttértudás révén lehetséges

Irodalomjegyzék

[1] M. Bellanger: "Digital Processing of Signals, Theory and Practice", John Wiley and Sons, 1986

[2] S. Smith: "Digital Signal Processing", Elsevier Science, 2003

[3] R. Plamondon "Pattern Recognition, Architectures, Algorithms & Applications", World Scientific Series in Computer Sciences - Vol.29

[4] M. Nitzberg, D. Mumford, T. Shiota "Filtering, Segmentation and Depth", Lecture Notes in Computer Science 662

[5] G. Kanizsa "Organization in Vision" New York: Preager, 1979, Ch.1-2.

[6] Iványi A. "Informatikai algoritmusok", ELTE Eötvös Kiadó, 2004

[7] J. F. Richards: "Remote sensing Digital image analysis", Springer-Verlag, 1986, Australia

[8] J. Duncan: "The Elements of Complex Analysis", John Wiley & Sons, 1972

[9] S. Russel – P. Norvig: "Mesterséges intelligencia, modern megközelítésben", Panem – Prentice Hall,2000