UDK 528.44(084.3—11):681.3
Prethodno saopćenje

AUTOMATIZIRANO UREĐIVANJE VEKTORSKE SLIKE KATASTARSKOG PLANA

Tomaž GVOZDANOVIĆ — Ljubljana*

SAŽETAK: Članak opisuje prototip programskog paketa namijenjenog automatskom uređivanju vektorskih slika katastrskih planova. Bavi se nekim teoretskim rješenjima i strukturom podataka, što je presudno za efikasnost programa, i objektno orijentiranim pristupom, koji se pokazao u fazi razvoja programa kao veoma pogodan za ovaj problem.

1. UVOD

Ovaj članak nadovezuje se na članak Frasa (1991), zbog čega se u njemu ne opisuju neke inače bitne stvari, kao što su globalni ciljevi rada i hardver. Detaljnije od spomenutog članka bavi se problemom automatskog uređivanja vektorske slike, koje može uz relativno dobro rješenje bitno skratiti vrijeme ručnog uređivanja i time ukupne obrade. Tako bi se mogli prilično rastereti operateri, što je kod velikog broja planova koje je potrebno pretvoriti u digitalni oblik i te kako važan aspekt.

2. ANALIZA ULAZNIH PODATAKA

Rezultat vektorizacije rasterske slike katastarskog plana je ASCII-datoteka u formatu DXF (Data eXchange Format), koji je standardni format za prijenos podataka između gotovo svih CAD sistem. Taj je zapis za naš primjer veoma neracionalan, međutim ipak smo ga usvojili zbog mogućnosti unosa slike u AutoCAD, u kojem dalje uređujemo slike.

Osnovna DXF-datoteka, u kojoj su bili testni podaci za 326 mm × 212 mm velik isječak lista, što predstavlja otprilike četvrtinu jednoga lista, bila je velika 2,6 Mb. Manipulacija tolikom količinom podataka u AutoCAD-u odmah se pokazala besmislenom jer su sva zumiranja slike u radu s dosta brzim diskom (13 ms) potrajala do 2,5 minute, a samo nešto manje u radu sa RAM-diskom. Kada smo natjerali AutoCAD da upotrebljava EMS-memoriju, maksimalno vrijeme za zumiranje skratilo se na otprilike 1 minutu, što je još uvijek previše za normalan rad. Naša odluka da vlastitim programima automatiziramo uređivanje bila je dakle sasvim na mjestu.

* Tomaž Gvozdanović, dipl. inž., FAGG, Jamova 2, Ljubljana.

Svi podaci u ulaznoj datoteci su organizirani u poligone, za koje DXF-standard predviđa ime POLYLINE. Lomne točke polilinija zovu se VERTEX-i.

Kako su sve koordinate lomnih tačaka bile ispisane na 4 decimalna mjesta, cijelu smo datoteku prvo prepisali u oblik s dva decimalna mjesta — na stotinku mm. Time smo datoteku skratili za otprilike 160 Kb, uz točnost koja je još uvijek za red veličine iznad grafičke točnosti, što je sasvim dovoljno.

Ulaznu datoteku smo statistički obradili. Za nas interesantni parametri bili su slijedeći:

— srednja dužina polilinije,
— srednji broj stranica u poliliniji,
— srednja dužina stranice.

Na osnovi rezultata koji se nalaze u prilozima 1 i 2 u koloni A konstatirali smo da program za vektorizaciju generira polilinije s dosta velikim brojem vrlo kratkih stranica. To nije nedostatak programa za vektorizaciju, već rezultat činjenice da velik dio cjelokupnog sadržaja plana (po našoj procjeni 40—60%) čini opisni dio, to jest brojevi parcela i napisi. To je za naš zadatak irelevantno, što znači da eliminacijom slova i brojki praktički možemo prepoloviti količinu podataka.

Zbog toga je naš rad u vezi s automatizacijom uređivanja bio usmjeren u odstranjivanje kratkih polilinija koje predstavljaju dijelove slova i brojki.

3. PRINCIPI ELIMINACIJE POLILINIJA

Strategiju automatizacije uređivanja, koja zapravo znači eliminiranje polilinija, određuju ovi zahtjevi:

— sve što je moguće automatizirati treba da obavi program, a operater treba da rješava samo sumnjive situacije,
— program u fazi »čišćenja« polilinija neka ove polilinije ne odbacuje, već samo klasificira.

Taktiku koju smo upotrijebili u eliminaciji nepoželjnih informacija zovemo pozitivna i negativna selekcija.

Negativna selekcija eliminira polilinije koje gotovo sigurno moramo odbaciti, dok pozitivna utvrđuje polilinije koje svakako predstavljaju traženu informaciju — parcelne granice i zgrade.

Kriteriji za eliminaciju u prvom redu zasnivaju se na dvije karakteristike svake polilinije: na dužini i povezanosti s ostalim polilinijama.

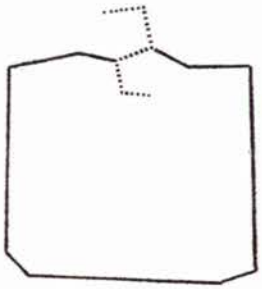
Dužinski elementi po kojima klasificiramo polilinije jesu dužina i udaljenost između krajnjih točaka. Osnova za računanje oba parametra jesu koordinate lomnih tačaka, koje su zapisane u DXF-datoteci.

Druga karakteristika — povezanost sa susjednim polilinijama — zasniva se na teoriji grafova i dopunjuje kriterije dužine. Povezanost polilinije ocjenjujemo težinom krajnjih točaka, to jest brojem koji kaže koliko je polilinija vezano za jedan čvor.

U narednim paragrafima opisani su razni primjeri polilinija, njihove karakteristike i kriteriji po kojima ih možemo eliminirati. Svi dužinski kriteriji određeni su empirijski.

Zgrade predstavljaju polilinije duže od 8 mm, kod kojih je udaljenost između krajeva manja od 1 mm. Uzrok što početna i konačna točka obično

nisu identične jest način vektorizacije i činjenica da su na gotovo svim zgradama nacrtana slova »Z« — znak pripadnosti (sl. 1).

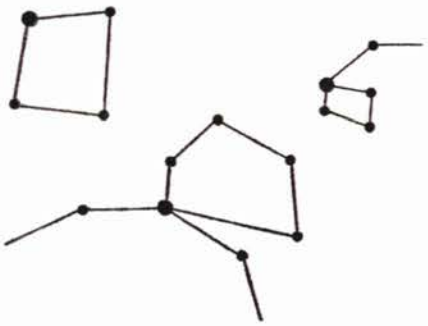


Sl. 1. Zgrade

Sl. 2. Slobodne polilinije

Slobodne polilinije su one koje na krajevima nisu povezane ni s jednom drugom polilinijom i kraće su od 4 mm (sl. 2).

Zatvorene polilinije imaju krajeve u istoj točki. Budući da takav kriterij ispunjavaju i zgrade bez znaka pripadnosti, za dužinski kriterj preuzeli smo donju granicu veličine zgrada, to jest 8 mm. Zatvorene polilinije mogu biti slobodne ili oslonjene na neku drugu poliliniju (sl. 3).



Sl. 3. Zatvorene polilinije

Sl. 4. Slijepe polilinije

Slijepe polilinije su kraće od 2 mm, a s drugim polilinijama povezane su samo na jednom kraju (sl. 4).

Grupe polilinija čine polilinije dužine do 6 mm, koje se drže zajedno i nisu povezane sa zgradama i granicama. Takve grupe generira softver za vektorizaciju u slučaju kada se slova ili brojke dotiču (sl. 5).

Sl. 5. Grupe polilinija

Čvorove — točke na krajevima polilinija, dijelimo na:

— prave čvorove, u kojima se sreću barem 3 polilinije (sl. 6a),
— viseće čvorove, koji imaju težinu 1 i leže na kraju slijepih polilinija (sl. 6b),
— pseudočvorove, koji imaju težinu 2; takve čvorove program za vektorizaciju ne generira, već nastaju prilikom eliminiranja polilinija. Polilinije, koje su povezane u pseuedočvoru, sastavljamo u jednu, s kojom radimo dalje (sl. 6c).

Sl. 6. Čvorovi: a — pravi čvorovi; b — viseći čvorovi; c — pseudočvorovi

4. OPREMA

Računalna oprema na kojoj smo razvijali i testirali programski paket opisana je u članku (Fras 1991), zbog čega ćemo više pažnje posvetiti softverskom dijelu sistema.

Programski jezik koji smo izabrali, jest C++ i predstavlja objektno orijentiranu nadgradnju danas tako popularnog jezika C. Jezik je relativno nov, ali stručnjaci misle da će to biti jezik 901ih godina. Premda smo u programiranju tim jezikom bili početnici, prednosti objektno orijentiranog pristupa pokazale su se vrlo brzo i omogućile nam relativno lako savlađivanje velike količine koda.

Razvojni sistem za koji smo se odlučili jest Zortech C++ V2.0 Developement Edition, koji je bio u to vrijeme praktično jedini pravi C++ sistem (sada je tu i Borlandov Turbo C++). Gotovo svi ostali sistemi nemaju, naime, pravi prevodilac (kompilator), već samo translator, koji kod C++ prepisuje u C.

Sistem uključuje kompletnu integriranu radnu sredinu po uzoru na Borlandove turbo-jezike s editorom, prevodiocima za C i C++, brzim linkerom i drugima programima.

Testovi koje smo obavili pokazli su da je sistem veoma brz i siguran, a programi prktički jednako veliki i eefikasni kao oni što su napisani u C-u.

5. PROGRAMSKI PAKET SKAN

Program SKAN sastoji se od 6 manjih programa. Prva četiri nazvali smo po fazama — F1, F2, F3, F4, dok se preostali zovu TODXF i ALIGN.

Uzrok što je paket podijeljen na tolik broj programa jest u načinu njegova razvijanja. Poslije svake obrađene faze analizirali smo rezultate i ustanovili što treba raditi slijedeći program. Premda svaki program učitava sve podatke i ispisuje rezultate za slijedeći program, vrijeme, koje je potrebno za obradu svih podataka neprimjerno je kraće od vremena koje bi za isti efekat trebao operater.

Cjelokupan paket, zajedno s čitavom analizom podataka i idejnim rješenjima, izradio je za malo više od mjesec dana jedan programer.

5.1. *Opis pojedinih programa*

Pojedini programi obavljaju ove zadatke:

F1 : — eliminira zgrade i zatvorene polilinije,
— indeksira i određuje dužinu svim polilinijama,
— indeksira sve čvorove i svakome određuje težinu,

F2 : — sastavlja stabla za traženje čvorova i polilinija,
— interativno traži i eliminira slijepe i slobodne polilinije,

F3 : — sastavlja polilinije i eliminira sve čvorove s težinom 2,
— eliminira polilinije, koje smo upotrijebili za sastavljanje novih, i sastavljanje polilinije, koje ispunjavaju neke od kriterija (zgrada),

F4 : — eliminira grupe,

TODXF : — zapisuje preostale polilinije u DXF-datoteku u izabrani sloj i u izabranoj boji,

ALIGN : — izravna polilinije u DXF- dadoteci.

5.2. *Organizacija podataka*

Najveći je problem velika količina podataka. U ovom je dijelu objašnjeno na koji smo način organizirali podatke i na kakve smo probleme naišli.

Sve polilinije imaju u DXF-datoteci koordinate svih lomnih tačaka, tako da možemo izračunati sve dužinske parametre polilinija. Do svih podataka o međusobnoj povezanosti polilinija moramo doći sami.

Model organizacije podataka veoma je jednostavan model relacijske baze — spisak veza između čvorova (polilinije) i spisak samih čvorova. Svaki od tih elemenata predstavljen je u programu kao objekat s određenim atributima.

Podaci o čvoru:

— redni broj,
— koordinate x, y,
— spisak susjednih polilinija.

Podaci o vezama između čvorova:

— redni broj,
— redni brojevi čvorova na kraju,
— dužina.

Za svaku poliliniju znamo koje čvorove povezuje, a za svaki čvor koje polilinije se u njemu sastaju. Kao prvo, moramo oba spiska sastaviti.

Spisak veza nije teško napraviti jer svaku poliliniju predstavljamo kao jednu vezu. Spisak čvorova napravimo tako da koordinate krajeva polilinije usporedimo s koordinatama čvorova u spisku. Pri tome dolazi do dva slučaja:

— ako naiđemo na čvor s identičnim koordinatama, povećamo mu težinu,
— ako ovakav čvor ne nađemo, u spisak uvodimo novi čvor s težinom 1.

Čvorove usput indeksiramo i svakoj vezi, kao atribute, pripisujemo indekse čvorova. Princip je veoma jednostavan, ali dolazi do jednog nepoželjnog

efekta. Linearno traženje čvorova je problem s kvadratnim vremenskim rastom. Pogledajmo kratak račun na primjeru koji je adekvatan testnom.

Svaka polilinija ima dva kraja. Kod 10000 čvorova moramo svaki kraj usporediti u prosjeku s 5000 čvorova, što znači da moramo za 10000 polilinija obaviti 10000 × 2 × 5000 = 100 milijuna uspoređivanja! Testovima smo ustanovili da naše računalo (386, 25 MHz) obavi u 1 sekundi manje od 10000 komparacija (zajedno s čitanjem podataka), što znači da bi proces trajao tri sata! Za cijeli list računalo bi trebalo dva dana, što znači da operater to može brže obaviti. Za takve probleme postoji lijek — stabla, koja smo objektno orijentiranim pristupom relativno lako savladali.

Sve čvorove smo svrstali u stablo s 4 grane, organizirano po obje koordinate. Svaki čvor ima 4 pokazivača na čvorove, koje smo u stablo svrstali kasnije. Dubina takvog stabla bila je u testnom primjeru 58, ali uz malu modifikaciju smanjena je na 20. To znači da svaki čvor možemo naći za najviše 20 komparacija i vrijeme od tri sata smanji se na 40 sekundi! Najbitnije pri tome je da vrijeme ne raste kvadratno, čak ni linearno, već logaritamski s osnovom 4. Tako bi po našim procjenama vrijeme za obradu cijelog lista bilo manje od 1 minute.

Cijena za tako kratko vrijeme je 200 Kb memorije za svakih 10000 čvorova! Potrebe za memorijom rastu linearno, što znači da bismo za cijeli list trebali 800 Kb, što je kod 4 Mb, koliko ima naše računalo, tek 20%.

Time smo riješili velik dio problema. U nastavku sve čvorove zapišemo po indeksima u tabelu i dalji rad je mnogo jednostavniji.

6. REZULTATI TESTNOG PRIMJERA

Ulazni podaci testnog primjera opisani su u glavi 2.

Progrm F1 prepoznao je 103 zgrade, što predstavlja 1% cjelokupne informacije, i eliminirano 280 zatvorenih polilinija, što iznosi još 3%.

Program F2 je u četiri iteracije eliminirao 3160 slijepih i 1470 slobodnih polilinija, što zajedno predstavlja 46% cjelokupnog sadržaja. Kao što vidimo, već prve dvije faze smanjuju količinu podataka na 50%.

Program F3 sastavio je 870 novih polilinija i eliminirao 2400 polilinija, koje su uključene u nove ili su nove i ispunjavaju jedan kriterij.

Program F4 potražio je 422 polilinije u 100 grupa, čime se cjelokupna količina smanjila na 32%.

Preostale polilinije smo pomoću programa TODXF u DXF-datoteku, čija dužina iznosi nešto više od 900 Kb. Ako smo nakon toga upotrijebili još program za ravnanje polilinija, dužina izlazne datoteke bila je samo 500 Kb, što predstavlja samo 1/5 početne.

Ovolika datoteka omogućava dosta normalan rad sa maksimalnim vremenom od 8 sekundi za operacije, povećanja i regeneriranja slike.

Cjelokupno vrijeme od početka programa F1 do kraja zapisa u DXF-datoteku s izravnanjem polilinija iznosi manje od 8 minuta.

7. PLANOVI ZA BUDUĆNOST

U slučaju da se ova koncepcija unošenja podataka akceptira, naši ciljevi bit će slijedeći:

— integriranje svih programa koji sačinjavaju paket u jedan program, čime će se dosta pojednostaviti manipulacija sa sada relativno velikim brojem datoteka,
— adaptiranje programa za veće količine podataka; naime, trenutna verzija za četvrtinu lista gotovo dokraja iskoristi 640 Kb memorije, zbog čega moramo podatke držati u EMS-memoriji,
— optimiziranje algoritma za eliminiranje i ravnanje u pogledu vremena i upotrebe memorije,
— istraživanje da li je moguće automatskim putem eliminirati još više podataka i, ako jest, izrada novih algoritama,
— smanjenje rada s datotekama (zbog upotrebe diska rad je usporen i mogućnost greške veća) koliko je moguće — to znači da program treba da učita početnu DXF-datoteku i na kraju ispiše konačnu DXF-datoteku,
— razvijanje novih programa za postprocesiranje, tj. obradu datoteke poslije ručnog uređivanja.

8. ZAKLJUČAK

Rješenje zadatka već u testnom primjeru donelo je rezultate mnogo bolje od očekivanih. Program je smanjio količinu podataka koje bi trebao eliminirati operator, s otprilike 80% cjelokupnog sadržaja na otprilike 10%. Već sada je vrijeme za cjelokupnu obradu za polovicu manje od vremena koje trebamo za unos digitalizacijom, i ako nastavimo s razvojem paketa, naše su procjene da možemo vrijeme još jednom prepoloviti.

PRILOG 1.

Analiza rezultata automatskog uređivanja

A : prije obrade
B : poslije eliminiranja polilinija
C : poslije izravnanja polilinija

	A	B	C
Broj polilinija :	10322	3218	3218
Broj stranica :	29932	13246	4543
Prosječna dužina stranice (mm) :	0.80	1.21	3.51
Prosječna dužina polilinija (mm) :	2.33	5.00	4.96

Relativna frekvencija broja stranica u poliliniji:

br. stranica	A	B	C
1	33.6	18.5	74.1
2	27.6	21.3	16.4

3	14.9	20.0	6.4
4	7.8	11.6	1.6
5	4.5	7.8	0.8
6	3.1	4.8	0.4
7	2.5	4.0	0.2
8	1.9	2.3	0.1
9	1.1	2.3	0.0
10	3.0	7.4	0.0
ukupno	100.0	100.0	100.0

PRILOG 2.

Relativna frekvencija stranica po dužini:

	A	B	C
do 1.0 mm	78.9	61.3	40.5
do 2.0 mm	11.5	19.3	16.6
do 4.0 mm	7.1	14.1	15.8
do 8.0 mm	2.3	4.9	15.3
do 16.0 mm	0.2	0.4	9.0
do 32.0 mm	0.0	0.0	2.3
do 64.0 mm	0.0	0.0	0.5
ukupno	100.0	100.0	100.0

Relativna frekvencija polilinija po dužini:

	A	B	C
do 1.0 mm	52.4	29.7	30.6
do 2.0 mm	21.3	18.2	17.9
do 4.0 mm	12.2	15.0	14.7
do 8.0 mm	7.5	18.1	17.9
do 16.0 mm	4.6	13.3	13.3
do 32.0 mm	1.5	4.0	4.0
do 64.0 mm	0.4	1.5	1.5
do 128.0 mm	0.0	0.1	0.1
ukupno	100.0	100.0	100.0

AUTOMATIZED EDITING OF VECTORIZED CADASTRIAL MAP

Paper describes prototype of software package for automatized editing vectorized pictures of cadastrial maps, with special view on theoretical solutions and data structure which are the key for program efectiveness, and object-oriented approach which showed all its benefits during program developement.

Primljeno: 1990-11-12