SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1690

POSTUPCI PRIKAZA VELIKE KOLIČINE PROSTORNIH PODATAKA GIS SUSTAVA

Hrvoje Keserica

Zagreb, listopad 2007.

Sadržaj

1. Uvod

Mnoge postojeće aplikacije sakupljaju i referenciraju podatke po njihovim geo-prostornim lokacijama. Većina uređaja koji se nadgledaju sustavom za nadzor i upravljanje (SCADA sustavi) rasprostiru se preko velikog geografskog područja. Broj takvih uređaja može biti vrlo velik i informacije o njima obično sadrže, uz druge atribute, geografsku lokaciju na kojoj se nalaze. Velike nakupine podataka koje sadrže više tisuća zapisa gotovo su nemoguće za brzo razumijevanje gledanjem u čiste podatke. Vizualizacija je neophodna za geografski smještaj i pregled. Tako se u SCADA sustavima javlja potreba za prikazom velike količine podataka pomoću GIS sustava.

Iako je geografska vizualizacija proučavana već više desetljeća, ovaj problem donosi nove izazove. Prikaz velike količine skupova točaka na uobičajenim mapama je problematično. Područja s gusto popunjenim podatkovnim točkama nisu jasna zbog toga što postoji velika vjerojatnost da će se točke međusobno precrtavati (engl. overplotting), dok rijetko popunjena područja bespotrebno iskorištavaju mnogo prostora koji može biti bolje iskorišten.

Podatkovna točka GIS sustava koja može predstavljati bilo koji element (npr. trafostanica kao element elektroenergetske mreže) može se naći u alarmantnom stanju, ali ona možda neće biti u prvi tren vidljiva jer se nalazi u području gdje je gusta raspodijeljenost podatkovnih točaka. Da bi se otkrila geo-prostorna lokacija takve podatkovne točke, bilo bi potrebno promijeniti mjerila pogleda. S druge strane, ako se SCADA sustavom razmatra samo dio karte, niti u jednom trenutku neće sve točke biti prikazane, a alarmantno stanje se može pojaviti baš na onom području koje u tom trenutku nije vidljivo. Radi toga je važno da u svakom trenutku bude vidljiva cijela karta područja koje se nadzire SCADA sustavom kako bi sve podatkovne točke bile prikazane, ali je isto tako važno da, radi bolje vidljivosti, one ne budu previše zgusnute kako bi operater koji nadgleda sustav mogao pravovremeno reagirati na promjene u sustavu.

Nedavna istraživanja vizualizacije informacije bavila su se tipovima transformacijskih funkcija kojima se dobiju prostorno transformirane karte s prepoznatljivim oblicima. Ovakvi tipovi prostornih transformacija se zovu funkcije globalnog oblika. Osobito su se istraživale deformacije kod mapa baziranih na kartogramu[2].

Ovaj rad opisuje metodu deformiranja područja karte uz pomoć kartograma koja sadrže važne točkaste geo-prostorne informacije tako da one budu podjednako gusto raspoređene kako bi se smanjila potreba za precrtavanjem točaka po istom te omogućio prikaz velike količine podataka.

2. SCADA (Supervisory Control And Data Acquisition)

2.1. Što je to SCADA sustav?

SCADA sustavi se koriste za sakupljanje podataka i ostvarivanje kontrole na nivou nadgledanja. Neki sustavi služe samo za praćenje i nemaju mogućnosti kontrole, ali svejedno spadaju u skupinu SCADA sustava.

Sustavi za nadzor i upravljanje su sustavi koji se postavljaju povrh sustava za kontrolu u realnom vremenu kako bi nadzirali procese koji ne pripadaju SCADA sustavu (npr. računalo samo za sebe nije SCADA sustav iako ono kontrolira vlastitu potrošnju električne energije i hlađenje). Ovo implicira da sustav nije ključan za kontrolu procesa u stvarnom vremenu, budući da za to postoji odvojeni ili integrirani automatizirani sustav koji može dovoljno brzo reagirati kako bi kompenzirao promjene u procesu unutar vremenskih konstanti procesa. Procesi mogu biti industrijski, infrastrukturni ili procesi unutar objekata:

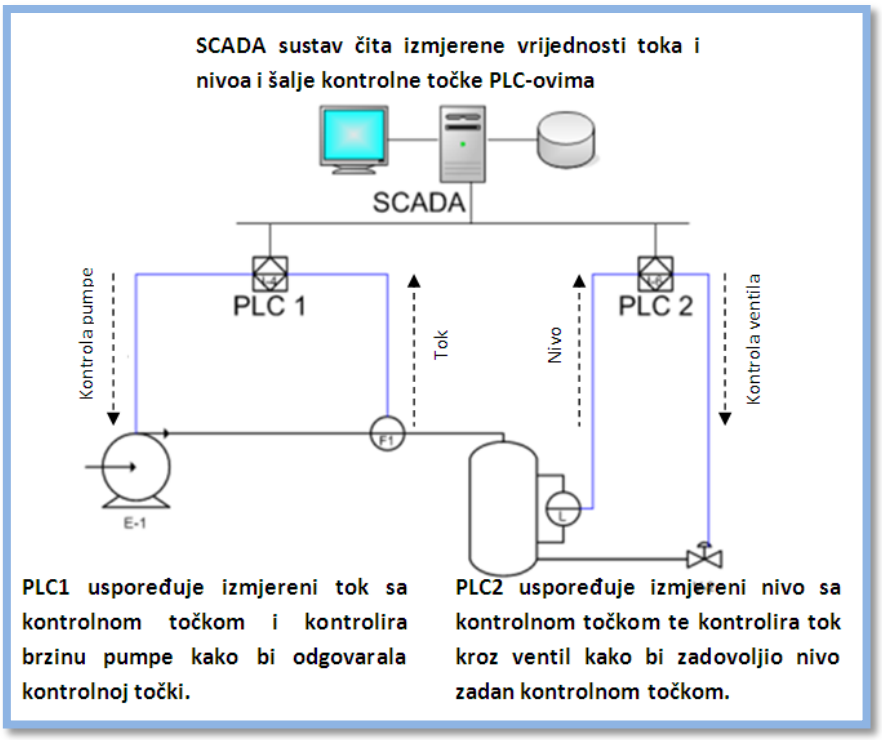
- Industrijski procesi mogu biti: manufaktura, proizvodnja, proizvodnja energije, izgradnja i razne prerade.
- Infrastrukturni procesi mogu biti javni ili privatni i uključuju: upravljanje vodama i njena distribucija, sakupljanje otpadnih tekućina i njihovo zbrinjavanje, plinovodne cijevi, transmisija i distribucija električne energije te veliki komunikacijski sustavi.
- Procesi unutar objekata mogu biti u privatnim ili javnim objektima: zgrade, aerodromi, brodovi ili svemirske postaje koje je potrebno nadgledati i kontrolirati (kontrola pristupa, potrošnja energije i slično).

Svi SCADA sustavi za navedene procese vrše kontrolu i nadgledanje uz prikupljanje podataka (engl. Supervisory Control And Data Acquisition) iako se oni funkcionalno u potpunosti razlikuju.

2.2. Koncept SCADA sustava

SCADA sustavi se sastoje od uređaja za primanje i slanje signala, kontrolnih uređaja, sučelja čovjek-stroj (engl. HMI - Human Macine Interface), komunikacije, baze podataka i programske podrške. Najčešći su kod sustava koja koriste razna mjerenja.

Termin SCADA se obično odnosi na centralni sustav kojim se nadgleda i kontrolira neki drugi sustav raširen na velikom području. Većina kontrole sustava se zapravo obavlja preko RTU (engl. Remote Terminal Unit) ili preko programibilnog logičkog kontrolera (engl. PLC – Programmable Logic Controller). No funkcije tih uređaja su obično ograničene na osnovne reakcije pri promjenama u sustavu ili na samo nadgledanje bez mogućnosti kontrole. Na primjer, PLC može kontrolirati tok vode za hlađenje kroz dio industrijskog procesa, ali SCADA sustav može dozvoliti operateru da promjeni kontrolne točke toka i dopušta zapisivanje i prikazivanje alarma pod bilo kojim uvjetima kao što je manjak protočnosti ili visoka temperatura. Krug povratnih veza je zatvoren između RTU ili PLC, SCADA sustav prati ukupne performanse tog zatvorenog kruga (Slika 2.1).



Slika 2.1: Primjer SCADA sustava.

Prikupljanje podataka započinje još na razini RTU-a ili PLC-a i uključuje rezultate mjerenja i stanje opreme koja se, po potrebi, šalju SCADA sustavu. Podaci se tada pripreme tako da operater kontrolne sobe korištenjem HMI-a može donijeti potrebne odluke da prilagodi ili premosti ranije definirane kontrole RTU-a ili PLC-a. Podaci se također mogu pohraniti u povijesne događaje kako bi bili mogući trendovi i druge analize.

2.3. Sustav čovjek-stroj (HMI)

Odnos čovjek-stroj ili HMI je instrument koji prezentira procesne podatke operateru i preko kojih operater kontrolira proces. HMI industrija se razvila iz potrebe za standardizacijom načina nadgledanja i upravljanja više udaljenih kontrolera, PLC-ova i drugih kontrolnih uređaja. PLC-ovi nude automatiziranu, unaprijed programiranu kontrolu procesa, ali su često distribuirani na velikom području što sakupljanje podataka bez SCADA sustava čini kompliciranim. Ranije PLC-ovi nisu imali standardni način prezentiranja informacije operateru. SCADA sustav sakuplja informacije od PLC-ova i kontrolera preko formirane mreže, uskladi ih i formatira. HMI također može biti povezan s bazom podataka kako bi se omogućili trendovi, dijagnoze nad podatcima i upravljanje informacijom.

SCADA sustav je popularan zbog njegove kompatibilnosti i pouzdanosti. Koristi se od malih sustava, kao što je kontrola temperature sobe, do velikih kao što je kontrola sustava za distribuciju i transmisiju električne energije.

2.4. SCADA i GIS

SCADA sustavi za kontrolu transmisije i distribucije električne energije sastoje se od kompleksne mreže elektronskih uređaja za mjerenje i slanje podataka preko LAN-a, WAN-a do kontrolnog centra. SCADA sustav poput alarmnog „psa čuvara“ (engl. watch dog) nadzire električnu mreže u stvarnom vremenu i nudi udaljene kontrole sklopkama, transformatorima i ostaloj opremi. Tako se objedinjuje mogućnost održavanja i oporavka od pogreške distribuiranih sustava.

Tipičan sustav električne energije se sastoji od proizvodnje, transmisije i distribucije. Struktura transmisije i distribucije električne energije je gusta mreža raširena na velikom geografskom području i sastoji se od raznolike opreme, trafostanica i elektrana. Veliku ulogu u prikazu i kontroli takve mreže uz SCADA sustave ima GIS (engl. Geographical Information System) koji se koristi za nadgledanje cjelokupne električne mreže od proizvodnje električne energije do točaka potrošnje uz pristup njihovim geoprostornim podacima.

Integracija GIS sustava sa SCADA sustavom uzdiže računalne mogućnosti kako bi se pružile vrlo korisne informacije velikom broju korisnika u vrijeme dok se susrećemo s povećanjem potrebe za informacijama u stvarnom vremenu. Napredne mogućnosti kao što su analize predviđanja, simulacijski alati, analize slučajnosti i nadgledanje mreže sežu do te mjere da iskorištenje električne energije bude uspješno.

2.4.1. Potreba za integracijom GIS-a i SCADA-e

S problemima kao što su: veliki gubitci električne energije, loša kvaliteta i pouzdanost opskrbe, krađa energije i sl. se, čak i u današnje vrijeme, susreće mnogo tvrtki za distribuciju električne energije. Ovo rezultira neefikasnom djelatnošću, nezadovoljstvom potrošača i smanjenjem tržišne vrijednosti kompanija. Ovi su problemi uglavnom usko vezani sa sustavima mreža transmisije i distribucije električne energije. Tradicionalni pristupi razvoja mreže trebali bi se zamijeniti pristupom baziranim na tehničkim zahtjevima i zahtjevima pouzdanosti, ekonomskim razmatranjima cijene gubitaka energije i proširivanjem sustava kako bi se postigao rast budućih zahtjeva uz manju cijenu.

Potrebne su pouzdane i dovoljno detaljne informacije postojećih mreža kako bi se olakšalo donošenje odluka svih aktivnosti sustava za upravljanje. Poboljšanje učinkovitosti, smanjenje vremena ispada i kontroliranje cijena postaje ključno za kompanije kako bi bile uspješne u današnjem okruženju visoke konkurentnosti.

Kompanije za distribuciju električne energije vide potrebu da sustavi imaju mogućnost sakupljanja, pohranjivanja i objavljivanja različitih podataka kako bi se oni podijelili među njima tako se održi konzistentnost podataka. Implementacija informacijske tehnologije (engl. IT – Information Technology) postaje od velike važnosti. Postoji mnogo različitih sustava za postizanje specifičnih ciljeva. Većina komunalnih kompanija ima implementiran IT sustav kao SAP, ISU-CCS, SCADA, DMS, AMR, OMS i GIS za njihovo poslovanje. Ti su IT sustavi dovoljno fleksibilni i podesivi kako bi se susreli s promjenjivim uvjetima tržišta na raznim nivoima organizacije, ali su izolirani jedni od drugih. Podaci se mogu održavati i dostupni su samo onima koje koriste iste sustave. Postojeći podaci jednog sustava ipak mogu biti dostupni drugim sustavima integracijom ovih sustava.

Zahtjevi za podatcima raznih korisnika mogu biti – podaci u stvarnom vremenu dobiveni od SCADA sustava i kompletna mreža iz GIS-a. Ovi podaci se mogu pohraniti i kasnije pronaći i koristiti od strane različitih ljudi iz bilo kojeg dijela svijeta za različite namijene. Ako se podacima rukuje ručno, vrlo je vjerojatno da će oni biti izgubljeni ili uništeni. Niti jedan sustav se ne može koristiti kao zamjena za drugi sustav i zato je potrebna tehnika integracije potrebna za razmjenu podataka različitih sustava.

Iako se SCADA već koristi za nadzor i upravljanje, GIS kao dodatak integrira različite sustave kako bi ujedinio njihove prednosti.

2.4.2. Zašto koristiti GIS sustav kao osnovu integracije?

Nakon što se uvidjela potreba za integracijom bilo je dobro istražiti što bi mogao biti osnovni sustav kojim bi se ona mogla ostvariti. Od točke generacije električne energije do potrošača, električna mreža ima geografske reference. One se mogu kreirati ili preslikati u GIS s dodatkom geografskih obilježja na jedinstveni način koji nije moguć u nekim drugim sustavima. GIS-om se dobije brz, točan i jedinstven prikaz mrežnih podataka. Pogled na električnu mrežu ostvarenu GIS-om se može lako interpretirati. Zaposlenicima je lako razumjeti (u okviru vizualizacije) i upravljati podatcima prikazanim na taj način. Korištenje GIS tehnologije obećava dobrobit, ne samo u povećanju efikasnosti operabilnosti već i poboljšanju sigurnosti, donošenju odluka, bolje komunikacije i širenju informacija.

GIS se može koristiti za:

- vizualizaciju mreže,
- rekonfiguraciju/optimizaciju mreže,
- pripremanje dizajna/sheme,
- upravljanje greškama u sustavu,
- analizama – što ako?,
- razvoju poslovanja,
- strateškom planiranju.

Ako se razmotri tipični scenarij gubitka električne energije; kad se gubitak prijavi SCADA ili bilo kojem sustavu koji radi u stvarnom vremenu, integrirani GIS sustav može, uz pomoć algoritma za predviđanje, trenutno odrediti najvjerojatniji dio električne mreže (npr. uređaj za napajanje) kod kojeg je došlo do pogreške zajedno s njegovom lokacijom. Može također prijaviti koji će dio mreže pasti zbog određenog problema koji se dogodio ranije. Operateri koji nadgledaju sustav će brzo dobiti informaciju i moći će obavijestiti korisnike koliko je, od prilike, vremena potrebno da se kvar otkloni. U međuvremenu ekipa (ili više njih) mogu odmah krenuti na lokacije kvara s odgovarajućom opremom te obaviti popravak u znatno bržem vremenu.

Svaki od sustava (npr. SCADA, Analizator mreže, i slični) ima specijaliziranu ulogu. GIS sustav neće zamijeniti niti jedan od tih sustava, ali ako je integriran uz njih može znatno poboljšati njihove mogućnosti.

3. Korišteni zapisi datoteka

Za potrebe ovog rada korišteni su slijedeći zapisi datoteka: ESRI-jev shapefile zapis i SVG zapis. Skupina shapefile datoteka se koristi za opis ulaznih podataka nad kojima se vrši obrada te kao format zapisa rezultata obrade. Osnovna ideja je bila da se struktura podataka pročitana iz shapefile skupine datoteka konvertira u SVG format i tako prikazuje na zaslonu putem internet preglednika. Rukovanje podacima se obavlja pomoću dostupnog otvorenog koda u MapScript dll-ovima koji od svoje verzije 4.5 podržavaju i korištenje SVG-a kao izlaznog formata vektorske grafike. MapScript je ukratko opisan kasnije (Poglavlje 4.1).

3.1. ESRI-ev vektorski zapis za razmjenu podataka - SHP

Popularni geoprostorni vektorski format za Geografske Informacijske Sustave (GIS) su ESRI-eve Shapefile datoteke[4]. ESRI (Environmental Systems Research Institute) je razvio i regulirao format otvorene specifikacije za potrebe prijenosa podataka između vlastitih i drugih programskih paketa. Dok se govori o „shapefile-u“ obično se to odnosi na skupinu datoteka istog prefiksa (npr. „jezera.*“) s ekstenzijama „.shp“, „.shx“, „.dbf“, itd.. No zapravo se „shapefile“ specifično odnosi na datoteku s „.shp“ ekstenzijom, ali je sama „.shp“ datoteka nekompletna i nije dovoljna za distribuciju jer su i druge datoteke potrebne za potpuno razumijevanje podataka koje one opisuju. U daljnjem tekstu se shapefile odnosi na „.shp“ datoteku, a ne na skupinu datoteka istog prefiksa u slučaju da nije drugačije napomenuto.

Shapefile-ovi prostorno opisuju točke, poligone i skupine linija (engl. polyline) koje mogu, na primjer, reprezentirati trafostanice, jezera i vodove između trafostanica. Svaki element također može imati i atribute koji ga opisuju kao što su to ime elementa ili nominalni napon.

Shapefile je digitalni vektorski format za spremanje geografske lokacije objekta te pripadnih atributnih podataka. Predstavljen je uz programski paket ArcView GIS v2 početkom 1990-ih godina, a do danas su ga prihvatili razni komercijalni i besplatni programi koji podržavaju čitanje i zapisivanje ovog formata.

Velika prednost shapefile-a je njegova jednostavnost. Oni sadrže samo primitivne geometrijske oblike kao što su točke, linije i poligoni. No takav način zapisa objekta je nepotpun i nedovoljan za korištenje u slučaju da objekti nemaju atribute koji opisuju što oni predstavljaju. Radi toga je potrebno uz sami shapefile dodati tablicu u kojoj su zapisana svojstva/atributi svakog primitivnog oblika u shapefile datoteci. Oblici (točke, linije i poligoni) zajedno s njihovim atributnim podacima nude nam beskonačno mnogo mogućnosti za reprezentaciju geografskih podataka.

3.1.1. Komponente datoteka

Shapefile se mora gledati kao cjelina, iako je zapravo shapefile skupina datoteka. Postoje tri neizostavne datoteke koje sadržavaju srž podatka. Uz njih postoji još osam opcionalnih datoteka koje, u osnovi, sadrže indeksne podatke kojima se poboljšavaju performanse rada s geoprostornim informacijama. Svaka individualna datoteka bi trebala odgovarati MS DOS 8.3 konvenciji imenovanja (osam znakova za prefiks imena datoteke, točka i tri znaka za sufiks, npr. „shapefil.shp“) da bi bila kompatibilna sa starim aplikacijama koje rukuju s shapefile-ovima. Zbog istog razloga, sve datoteke trebaju biti locirane u istom direktoriju.

Osnovne (neizostavne) datoteke su:

- .shp – datoteka koja sadrži geometriju objekta,
- .shx – datoteka koja sadrži indekse geometrije objekta i
- .dbf – baza podataka za atribute objekta.

Opcionalne datoteke:

- .sbn i .sbx – spremaju prostorne indekse objekta,
- .fbn i .fbx – spremaju prostorne indekse objekta koji se nalaze u shapefile-ovima predodređenim samo za čitanje (read-only),
- .ain and .aih - spremaju indekse atributa aktivnih polja u tablici,
- .prj – datoteka sadrži koordinatni sustav informacija korištenjem „dobro poznatog teksta“ (engl. WKT – Well Known Text),
- .shp.xml – meta podaci za shapefile datoteku i
- .atx – indeksi za atributne podatke zapisane u .dbf datoteci, format zapisa je <shapefile>.<columnname>.atx (predstavljeno programskim paketom ArcGIS 8).

3.1.1.1. Shapefile zapis - (.shp)

Glavna datoteka koja sadrži primarne referentne podatke skupine shapefile datoteka. Sastoji se od glavnog zaglavlja fiksne veličine za kojim slijedi jedan ili više zapisa varijabilne veličine. Svaki zapis varijabilne veličine sadrži dvije komponente: zaglavlje zapisa i sadržaj zapisa (Tablica 3.1). Detaljan opis formata datoteke može se naći u tehničkom opisu ESRI-jeve Shapefile datoteke (www.esri.com/library/whitepapers/pdfs/shapefile.pdf).

Tablica 3.1: Organizacija osnovne datoteke[4].

Zaglavlje osnovne datoteke	(100B)		
Zaglavlje zapisa	(8B)	Sadržaj zapisa	(varijabilno)
Zaglavlje zapisa	(8B)	Sadržaj zapisa	(varijabilno)
…		…	
Zaglavlje zapisa	(8B)	Sadržaj zapisa	(varijabilno)

Glavno zaglavlje je veličine 100 bajtova i sadrži 17 polja (devet polja od četiri bajta i osam polja od osam bajtova):

- Bajtovi 0-3: kôd datoteke (uvijek $0000270A_{16}$),
- Bajtovi 4-23: se ne koriste,
- Bajtovi 24-27: sadrže zapis o veličini datoteke (broj 16-bitnih riječi),
- Bajtovi 28-31: verzija datoteke,
- Bajtovi 32-35: tip oblika (Tablica 3.2) i
- Bajtovi 36-99: Granični okvir (engl. bounding box) koji sadrži minimalne i maksimalne vrijednosti X, Y, Z i M.

Nakon toga slijede zapisi varijabilne veličine kojima je zaglavlje fiksno (8 bajtova) i sadrži dva polja: podatak o broju zapisa i veličini sadržaja; te sam sadržaj zapisa koji je različit za svaki tip oblika (prikazan je u Tablici 3.2).

Tablica 3.2: Tipovi oblika i njihova polja u sadržaju zapisa[4].

TIP OBLIKA	VRIJEDNOST	SADRŽAJ ZAPISA
Null Shape	0	Shape Type
Point	1	Shape Type, X, Y
Polyline	3	Shape Type, Box, NumParts, Num Points, Parts, Points
Polygon	5	Shape Type, Box, NumParts, NumPoints, Parts, Points
Multi point	8	Shape Type, Box, NumPoints, Points
PointZM	11	Shape Type, X, Y, Z, M
PolylineZ	13	Neizostavno: Shape Type, Box, NumParts, Num Points, Parts, Points, Z range, Zarray Opcionalno: M range, M array
PolygoZ	15	Neizostavno: Shape Type, Box, NumParts, Num Points, Parts, Points, Z range, Zarray Opcionalno: M range, M array
MultiPointZ	18	Neizostavno: Shape Type, Box, NumPoints, Points, Zrange, Zarray Opcionalno: M range, Marray
PointM	21	Shape Type, X, Y, M
PolylineM	25	Neizostavno: ShapeType, Box, NumParts, NumPoints, Parts, Points Opcionalna polja: M range, M array
MultiPointM	28	Neizostavno: Shape Type, Box, NumPoints, Points Opcionalno Fields: M range, Marray
MultiPatch	31	Neizostavno: Shape Type, Box, NumParts, NumPoints, Parts, PartTypes, Points, Z range, Z array Opcionalno: M range, M array

3.1.1.2. Indeksna datoteka (.shx)

Sadrži zaglavlje veličine 100 bajtova koje je organizirano jednako kao i zaglavlje glavne datoteke „.shp“ opisano ranije za kojim slijede zapisi fiksne veličine od 8 bajtova (Organizacija indeksne datoteke prikazana je Tablicom 3.3). Veličina datoteke zapisana u zaglavlju je ukupna veličina indeksne datoteke mjerene 16-bitnim riječima (pedeset 16 bitnih riječi za zaglavlje plus broj zapisa pomnožen s četiri).

Tablica 3.3: Organizacija indeksne datoteke[4].

Glavno zaglavlje indeksne tablice	(100B)
Zapis	(8B)
Zapis	(8B)
…	
Zapis	(8B)

U svakom zapisu indeksne datoteke se nalazi pomak (engl. offset) i veličina sadržaja odgovarajućeg zapisa glavne datoteke. Pomak zapisan u indeksnoj datoteci je broj 16-bitnih riječi od početka glavne datoteke do prvog bajta zaglavlja zapisa na koji se pomak odnosi. Tako pomak za prvi zapis u glavnoj datoteci iznosi pedeset 16-bitnih riječi zbog zaglavlja duljine 100 bajta. Veličina sadržaja sačuvana u indeksnom zapisu odgovara vrijednosti spremljenoj u zaglavlju zapisa glavne datoteke.

3.1.1.3. Atributni podaci (.dbf)

Datoteka dBASE (ili xBASE) sadrži bilo koji atribut objekta ili ključ prema kojem se druge tablice mogu spajati. Format ove datoteke je standardna DBF datoteka korištena u raznim tablično baziranim aplikacijama u Windowsima i DOS-u. Detaljni opis formata se može naći na (www.inprise.com).

Moraju biti zadovoljena tri uvjeta:

- Ime datoteke mora imati isti prefiks kao glavna i indeksna datoteka, a sufiks mora biti „.dbf“,
- tablica mora imati barem jedan zapis za svaki objekt,
- poredak zapisa mora biti isti kao u glavnoj „.shp“ datoteci i
- godina u dBASE zaglavlju se zapisuje kao godina od 1900-te godine.

3.1.2. Nedostaci

3.1.2.1. Topologija i shapefile-ovi

Najveći nedostatak skupine datoteka shapefile-a je taj što nemaju mogućnost spremanja topološke informacije.

3.1.2.2. Prostorna reprezentacija

Rubovi skupine linija (engl. polyline) i poligona su definirani korištenjem točaka čime se mogu dobiti nazubljeni rubovi. Kako bi se dobio „gladak oblik“ potrebno je unositi dodatne točke što zauzima puno podatkovnog prostora. Ovaj problem bi se mogao riješiti korištenjem Bezierovih krivulja kojima se postižu „glatki oblici“ bez korištenja puno točaka, ali trenutno niti jedan od shapefile tipova ne podržava Bezierove krivulje. No, mogućnost zapisa Bezierovim krivuljama bi narušilo jednostavnost oblika u shapefile-u.

3.1.2.3. Baza podataka

Za razliku od većine baza podataka, xBASE format nema mogućnost spremanja NULL vrijednosti u svoja polja. Ovim se ograničenjem smanjuje fleksibilnost spremanja atributnih podataka.

3.2. SVG zapis

3.2.1. Osnove SVG zapisa

Skalabilna vektorska grafika (SVG – Scalable Vector Graphics) je jezik za opis dvodimenzionalne grafike i grafičkih aplikacija u XML-u. Ona daje mogućnost web programerima, dizajnerima i korisnicima izbjegavanje ograničenja HTML-a i stvaranje robusnog vizualnog sadržaja i interakcije preko jednostavno deklariranog programskog modela.

SVG[5] je prikladan za Web aplikacije bazirane na podatkom-vođenu, interaktivnu grafiku. Podatci mogu pristizati u stvarnom vremenu kao što je to kod sistema elektronskog poslovanja korporativnih baza podataka ili SCADA sustava. Programeri mogu prilagoditi SVG za široku publiku, kulturu i demografiju, bez obzira kako se korisnik služi aplikacijom.

SVG je predočen kao otvoreni standard od World Wide Web Consortium-a (W3C) 1999. godine za objavljivanje animacija i za interaktivne aplikacije koje koriste vektorsku grafiku na web-u. Godine 2004., većina proizvođača mobilnih telefona odabire SVG kao osnovu za njihovu grafičku platformu.

3.2.2. Tehnički detalji

Moguće je korištenje tri tipa grafičkih objekata: oblici zadani vektorskom grafikom, slike i tekst. Grafički se objekti mogu grupirati te im se mogu dodjeljivati stilovi, mogu se transformirati i pridružiti ranije prikazanim (engl. renderiranim) objektima. Tekst može biti u bilo kojem XML prostoru imena (engl. namespace) prikladnom za aplikaciju čime se dobije poboljšana pretraživost i pristup SVG grafici.

SVG crteži mogu biti dinamički i interaktivni. Objektni model dokumenta (engl. DOM – Document Object Model) za SVG uključuje potpuni DOM XML-a čime se neposredno omogućuje efikasna animacija vektorske grafike uz pomoć skripti. Bogati skup rukovanja događajima može se dodijeliti bilo kojem grafičkom SVG objektu kao što je to prelaženje mišem preko objekta (engl. onmouseover) ili klik miša na objekt (engl. onclick).

3.2.3. Osnovna obilježja SVG-a

Obilježja SVG-a:

- Mala veličina datoteke – U prosjeku, SVG datoteke su manje od ostalih grafičkih formata na Web-u kao što su JPEG i GIF te su brze za skidanje.
- Neovisne su za prikaz – SVG slike su uvijek svježe na ekranu i ispisuju se rezolucijom koju podržava printer, bilo da je to 300 dpi, 600 dpi ili printer veće rezolucije. Nikad se ne dobiju nazubljeni rubovi zbog mogućnosti skaliranja vektorskog zapisa (engl. anti-aliasing).
- SVG nudi paletu od 16 milijuna boja i podržava ICC profile boja, sRGB, gradijente i maskiranje.
- Interakcija i inteligencija – Budući da je SVG baziran na XML-u, nudi neusporedivu dinamičku interakciju. SVG slike reagiraju na korisničke akcije pomoću označavanja (engl. highlight), korisnih natuknica (engl. tool tip), specijalnih efekata, zvuka i animacije.
- Zumiranje – Korisnici mogu povećati sliku do 1600% bez gubitaka oštrine, detalja ili bistrine. Tekst u SVG-u ostaje tekst, na slikama su i dalje moguće promjene (unutar izvornog koda), a najvažnija osobina je da je moguće pretraživanje (za razliku od rasterskih i binarnih dvojnika). Ne postoje ograničenja u fontovima i razmještaju (engl. layout), a svi korisnici vide sliku na isti način.

- Datoteke bazirane na tekstu – SVG datoteka je bazirana na tekstu (nije binarna). Na taj je način ona čitljiva od strane čovjeka kao što je to i HTML. Čak i početnik može gledati SVG izvorni kod i odmah naslutiti kako se opisni sadržaj povezuje s grafičkom reprezentacijom. Npr.

Slika 3.1: Primjer SVG zapisa.

Kod za ovaj SVG izgleda ovako:

```
... obrisan kod ...
<rect id="Rectangular_shape" fill="#FF9900" width="85.302"
height="44.092"/>
<ellipse id="Elliptical_shape" fill="#FFFF3E" cx="42.651"
cy="22.046" rx="35.447" ry="16.426"/>
<text transform="matrix(1 0 0 1 16.2104 32.2134)" font-
family="ÅfMyriad-RomanÅf" font-size="31.2342">SVG</text>
... obrisan kod ...
```

Tri objekta od kojih se sadrži ovaj primjer su prepoznatljivi čitanjem koda. Jasno se vidi da se tu nalazi jedan pravokutni (engl. rectangle) <rect>, jedan eliptični <ellipse> i jedan tekstualni <text> element. Pozicija i stil su čitljive i lako se mogu modificirati jednostavnim tekstualnim editorom. Na primjer, može se promijeniti boja za popunu elipse jednostavnom promjenom vrijednosti boje u *fill* opisu.

3.2.4. SVG i GIS

Geografski informacijski sustavi imaju specifične zahtjeve: bogate grafičke performanse, podršku za rasterske i vektorske sadržaje i mogućnost rukovanja s velikom količinom podataka. SVG je dobar odabir za to i mnogo GIS sustava nudi prikaz podataka u SVG-u.

Mogućnost proširivanja SVG-a tako da se pridružuju meta podaci čini SVG korisnim za korištenje na kartama. Na primjer, grafički se elementi mogu identificirati kao ono što oni u stvari jesu (kao što je to jezero) čime aplikacije imaju mogućost interakcije s objektima temeljene na grafici.

3.2.5. XML – osnova SVG standarda

Slijedi opis XML jezika budući da je SVG baziran na njemu.

XML je kratica za EXtensible Markup Language[6] - odnosno jezik za označavanje i prijenos podataka. Ideja je bila stvoriti jedan jezik koji će biti jednostavno čitljiv i ljudima i računalnim programima. Princip realizacije je vrlo jednostavan: odgovarajući sadržaj treba se uokviriti odgovarajućim

oznakama koje ga opisuju i imaju poznato ili lako shvatljivo značenje. Format oznaka u XMLu vrlo je sličan formatu oznaka u npr. HTML jeziku. Danas je XML jezik vrlo raširen i koristi se za različite namjene: odvajanje podataka od prezentacije, razmjenu podataka, pohranu podataka, povećavanje dostupnosti podataka i izradu novih specijaliziranih jezika za označavanje. XML je standardizirani jezik i za njegovu standardizaciju brine se World Wide Web Consortium.

3.2.5.1. Kako je nastao XML?

Problem HTMLa je što ima mali skup zadanih oznaka. Kada se želi proširiti s novim oznakama mora se mijenjati standard što ga čini nepraktičnim. Osim toga, iako je HTML izvorno zamišljen kao jezik za opisivanje sadržaja, zbog potreba i želja tržišta te razvoja preglednickih tehnologija (naročito za vremena "preglednickog rata" između Microsofta i Netscape-a proširivan je nestandardnim oznakama koje su prvenstveno služile za formatiranje sadržaja u smislu njegovog prikaza u internet pregledniku. Za opisivanje sadržaja SGML je bolji izbor od HTMLa, ali ima veliki nedostatak što je preglomazan za korištenje i izvršavanje unutar internet preglednika. Zbog toga je trebalo kreirati jezik koji će s jedne strane biti dovoljno malen i jednostavan da se može izvršavati unutar Internet preglednika, a s druge strane dovoljno prilagodljiv da se može proširivati korisničkim oznakama. Izradu specifikacije takvog jezika prihvatio se početkom 90ih godina 20. stoljeća World Wide Web Consortium. Željeli su razviti jezik koji će objediniti jednostavnost HTMLa i izražajnu snagu SGMLa. Na početku su odredili su deset ciljeva kojih su se u razvoju trudili pridržavati:

1. XML mora biti izravno primjenjiv preko Interneta,
2. XML mora podržavati širok spektar primjena,
3. XML mora biti kompatibilan s SGLM-om,
4. Mora biti lako pisati programe koji procesiraju (engl. parsiraju) XML dokumente,
5. Broj opcionalnih mogućnosti u XML-u mora biti minimalan, u idealnom slučaju jednak nuli,
6. XML dokumenti moraju biti čitljivi ljudima, te u razumnoj mjeri jednostavni,
7. Standard mora biti specificiran što prije,
8. Dizajn XML-a mora biti formalan i precizan,
9. Kreiranje XML dokumenata mora biti jednostavno,
10. Sažetost kod označavanja dokumenta XML-om je od minimalnog značaja.

World Wide Web Consortium je 10. veljače 1998. objavio prvu verziju XML preporuke.

3.2.5.2. Primjer XML dokumenta

```
<?xml version="1.0" encoding="UTF-8"?>
<poruka>
        <za>Pero</za>
        <od>Kate</od>
        <naslov>Podsjetnik</naslov>
        <tijelo>Otiđi kupiti kruh</tijelo>
</poruka>
```

XML dokument se sastoji od dva dijela. Prvi dio je prolog ili zaglavlje:

```
<?xml version="1.0" encoding="UTF-8"?>
```

U njemu se navode podaci koji opisuju XML dokument kao što su verzija XML preporuke prema čijim pravilima je dokument napravljen i kodnu stranicu. Ako se ne navede ispravna kodna stranica programi koji barataju s XML dokumentom kada naiđu na nestandardni znak (npr. naše slovo č) javiti će grešku.

Drugi dio je sadržaj dokumenta u kojem se nalazi korisni sadržaj omeđen XML oznakama.

```
<poruka>
        <za>Pero</za>
        <od>Kate</od>
        <naslov>Podsjetnik</naslov>
        <tijelo>Otiđi po kruh</tijelo>
</poruka>
```

Unutar korijenskog elementa ugniježđeni su svi ostali.

3.2.5.3. XML elementi

XML elementi opisuju određeni dio XML dokumenta. Sastoje se od korisnog sadržaja omeđenog XML oznakama. Sadržaj mora biti omeđen s lijeve i s desne strane početnom i završnom oznakom. Primjer XML elementa:

```
<naslov>Podsjetnik</naslov>
```

Ovaj element kaže da je sadržaj unutar njega naslov poruke. Početna oznaka je:

```
<naslov>
```

Oznaka ima svoj naziv omeđen znakovima "trokutastih zagrada" (< >). Završna oznaka je:

```
</naslov>
```

Završna oznaka identična je početnoj osim što s lijeve strane prije naziva ima kosu crtu. Sadržaj ovog elementa je:

```
Podsjetnik
```

3.2.5.4. Međusobni odnos XML elemenata

XML elementi međusobno mogu biti u odnosu roditelj-dijete (engl. parent-child) ili sestrinskom (engl. siblings). Kod odnosa roditelj-dijete jedan element hijerarhijski je nadređen drugome.

```
<poruka>
 ...
 <naslov>Podsjetnik</naslov>
 ...
</poruka>
```

U ovom primjeru element poruka je roditelj elementu naslov (nadređen je elementu naslov). Kaže se da je element naslov ugniježđeni unutar elementa poruka. U XML-u elementi moraju biti pravilno ugniježđeni što znači da ne smije doći do preklapanja oznaka.

Kod sestrinskog odnosa elementi se nalaze na istoj razini.

```
<naslov>Podsjetnik</naslov>
<tijelo>Otiđi po kruh</tijelo>
```

U ovom primjeru element i naslov i tijelo hijerarhijski se nalaze na istoj razini.

3.2.5.5. XML atributi

XML atributi nude podatke koji dodatno opisuju XML elemente. Podaci koji se navode kao atributi mogu biti zanimljivi čovjeku koji gleda sadržaj XML dokumenta (npr. ime osobe koja je kreirala poruku) ili računalnom programu koji parsira XML dokument (npr. vrsta i veličina fonta kojim se treba prikazati sadržaj poruke na ekranu). Atributi imaju svoj naziv i vrijednost koja se navodi pod navodnicima.

```
<poruka>
        <za font_face="Arial" font_size="10">Pero</za>
        <od font_face="Arial" font_size="10">Kate</od>
        <naslov font_face="Arial" font_size="14">Podsjetnik</naslov>
        <tijelo font_face="Arial" font_size="10">Otiđi po kruh</tijelo>
</poruka>
```

Oko korištenja atributa unutar XML dokumenta postoje dvije struje mišljenja. Jedna struja zastupa "čistoću elemenata" i smatra da se atributi ne bi smjeli koristiti za obogaćivanje smisla sadržaja. To u praksi znači da se npr. kao atribut XML dokumenta poruke iz primjera ne bi smjelo navoditi ime osobe koja je autor poruke, već bi se to trebalo navesti kao poseban element u dokumentu. Atributi su dozvoljeni samo za dodavanje dodatnih informacija koje služe programima za parsiranje XML dokumenata da na željeni način prikažu sadržaje. Prema njihovom mišljenu npr. dozvoljen je atribut koji opisuje vrstu i veličinu fonta jer on ništa ne dodaje u osnovnu informaciju poruke. On samo pomaže da se poruka na odgovarajući način prikaže na određenom mediju. Druga struja smatra da se atributi mogu slobodno koristiti za dodavanje korisnih informacija u XML dokument jer se time smanjuje inflacija elemenata. Prema njihovom mišljenju sasvim je u redu da se u XML dokument dodaju atributi koji nose korisni sadržaj. XML dokument iz ovog primjera mogao bi prema njima izgledati i ovako:

```
<poruka za="Pero" od="Kate">
        <naslov>Podsjetnik</naslov>
        <tijelo>Otiđi po kruh</tijelo>
</poruka>
```

3.2.5.6. Provjera ispravnosti XML dokumenta

XML dokument koji je kreiran u skladu sa sintaksnim pravilima je formalno ispravno kreiran (engl. Well Formed). Osim te formalne ispravnosti, ispravnost XML dokument se može kontrolirati u odnosu

na propisanu shemu. Shema određuje koje elemente smije sadržavati XML dokument te redoslijed i broj tih elemenata. Ispravnost provjerava se u odnosu na određeni tip definicije dokumenta (engl. DTD – Document Type Definition) ili XML schemu. Dokument sheme navodi se u zaglavlju XML dokumenta.

Povezivanje XML dokumenta s DTD shemom:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE poruka SYSTEM "PorukaPodsjetnik.dtd">
<poruka>
        <za>Pero</zaKoga>
        <od>Kate</odKoga>
        <naslov>Podsjetnik</naslov>
        <tijelo>Otiđi po kruh</tijelo>
</poruka>
```

Povezivanje XML dokumenta s XML schemom:

```
<?xml version="1.0" encoding="UTF-8"?>

<poruka xmlns="http://hr.wikipedia.org"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="poruka.xsd">
<poruka>
        <za>Pero</zaKoga>
        <od>Kate</odKoga>
        <naslov>Podsjetnik</naslov>
        <tijelo>Otiđi po kruh</tijelo>
</poruka>
```

3.2.5.7. Prednosti i nedostaci XML-a

Prednosti XML-a:

- Jednostavno je čitljiv i čovjeku u običnom tekstualnom editoru i računalu.
- XML dokument je obična tekstualna datoteka čitljiva na svakoj platformi koja može čitati tekstualne podatke. To ga čini neosjetljivim na tehnološke promjene jer bez obzira na napredak tehnologije, tekstualni podaci će još jako dugo ostati nešto što će svaki računalni sustav trebati moći pročitati.
- Podržava Unicode i omogućuje prikaz teksta na svim danas poznatim jezicima.
- Format je samo-dokumentirajući - oznake opisuju sadržaj koji se nalazu unutar njih.
- Ima stroga sintaksna pravila tako da je jednostavno kontrolirati ispravnost nastalog dokumenta. Računalni programi koji obrađuju dokument zbog toga mogu jednostavno obrađivati XML sadržaj.
- Međunarodno je prihvaćen standard. Mnogi proizvođači programa su ga prihvatili i koriste u svojim proizvodima.
- Hijerarhijska struktura je pogodna za opisivanje mnogih sadržaja (ali ne i svih!).
- Kompatibilan je sa SGMLom koji se koristi od 80ih godina 20. stoljeća, a za SGML postoji dosta računalnih programa koji ga mogu obrađivati. Tako da kada je XML standard objavljen

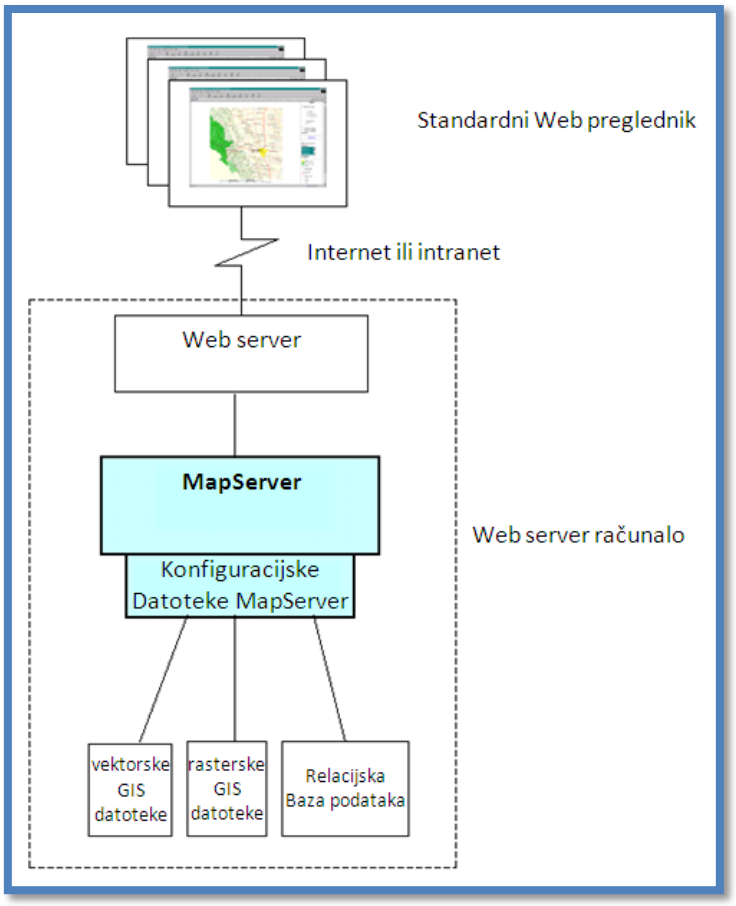
već je postojala određena baza korisnika koji nisu gotovo ništa morali učiti o novom jeziku niti kreirati nove programe već su ga jednostavno usvojili.

Nedostaci XML-a:

- Sintaksa je redundantna i opširna što može zamarati i zbunjivati osobu koja čita XML dokument. Računalni program koji obrađuje dokument morati će obraditi veliku količinu podataka što će ga djelomično usporiti.
- Da bi dokument bio dovoljno dobro "samoopisan", nazivi oznaka moraju biti dovoljno precizni što dovodi do dugih "kobasičastih" naziva (npr. u nekom dokument nije dovoljan opis podatka Ime_i_prezime jer to može biti ime i prezime osobe koja je kreirala dokument, ali i osobe na koju se slog odnosi, osobe kojoj se dokument šalje, ... Nije dovoljno osloniti se na "samoopisivanje" sadržaja.
- Redundancija i velika količina podataka stvaraju velike zahtjeve za propusnosti mreže (iako to danas uglavnom više nije problem).
- Programi koji obrađuju XML podatke su dosta složeni jer moraju obrađivati velike količine ugniježđenih podataka u više razina.
- Nedostatak formalno propisanih formata za podatke može stvarati probleme ako sudionici u razmjeni nisu dobro opisali (npr. da li se decimalni brojevi prikazuju s decimalnom točkom ili zarezom.
- Pohrana XML podataka u relacijske baze podataka nije prirodan način i to dovodi do smanjenja performansi sustava koji koriste takav način pohrane. S druge strane XML baze podataka koje su razvijene za pohranu XML podataka još su u fazi razvoja.

4. MapServer

MapServer[7] je, u osnovi, CGI (engl. Common Gateway Interface) program koji se nalazi na Web serveru i čeka zahtjeve. MapServer odgovara na zahtjev uz pomoć prenesenih informacija kroz URL i Map datoteke te stvori sliku zatražene karte. Zahtjev može također biti za slikom legende, slikom mjerila, slikom referentne karte te vrijednosti prenesene kao CGI varijable. Slika 4.1 prikazuje tipični dijagram MapServer aplikacije.



Slika 4.1: Konceptualni dijagram tipične MapServer aplikacije.

MapServer je dosta fleksibilan i može se proširivati i prilagođavati. Osnovni ulazni zapis je ESRI-jev shapefile, ali se može izgraditi tako da prihvaća razne ulazne podatkovne zapise. Od verzije 4.5 omogućen je SVG zapis kao izlazni format.

4.1. MapScript - sučelje MapServer-a

MapScript pomoću skupine skripti nudi sučelje MapServer-a za konstrukciju Web i samo-stojećih aplikacija. Koristi se neovisno od CGI MapServer programa kao modul za razne programske jezike kao što su: Php, Perl, Python, Ruby, Tcl, Java i C#.

Kako je već spomenuto ranije, MapScript je u ovom radu korišten za rukovanje ulaznim i izlaznim zapisima karte (SHP i SVG) te se pomoću njega ostvaruje funkcija za ispitivanje pripadnosti točke poligonu.

4.2. Standardni sadržaj MapServer aplikacije

Jednostavna MapServer aplikacija se sastoji od:

- Map datoteke – strukturirana tekstualna datoteka koja opisuje konfiguraciju MapServer aplikacije. Definira područje karte, govori MapServeru gdje se ulazni podaci nalaze te definira slojeve karte uključujući njihove projekcije i simbole pomoću kojih će se prikazati.
- Geografski podaci – MapServer podržava razne geografske podatke, ali standard mu je ESRI shapefile.
- HTML stranice – sučelje između MapServera i korisnika. MapServer se poziva kako bi postavio interaktivnu kartu na html stranicu.

Slijedi primjer dijela Map datoteke korištene u ovom radu na kojem je prikazana osnovna struktura datoteke te jedan od slojeva karte:

```
MAP

      NAME "Zone Samples"
      SHAPEPATH "C:\training\usa"
      SIZE 400 400
      STATUS ON
      EXTENT -125.46 19.89 -66.96 49.37
      UNITS METERS
      FONTSET "C:\training\usa\fonts\fonts.list"

      WEB
            IMAGEPATH "C:\Inetpub\wwwroot\temp"
            IMAGEURL "C:\Inetpub\wwwroot\temp"
      END

      LAYER
            NAME "usa_st"
            TYPE POLYGON
            STATUS ON
            DATA "usa_st"
            LABELITEM "STATE_NAME"
            CLASS
                  STYLE
                              COLOR 255 235 190
                              OUTLINECOLOR 0 0 0
                              SYMBOL 0
                  END
                  LABEL
                        COLOR 0 0 0
                        FONT arial
                        TYPE TRUETYPE
                        SIZE 7
                        POSITION CC
                  END

            END

      END
      ... obrian kod ...
END
```

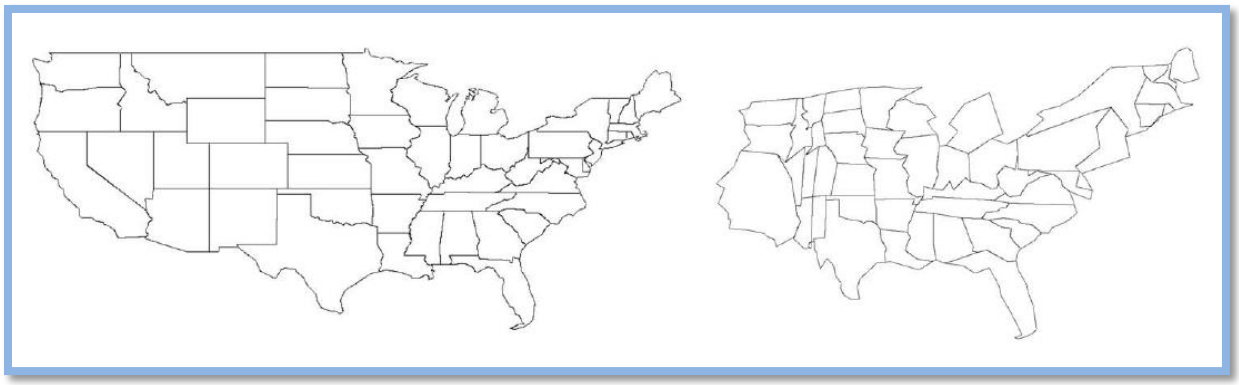
5. Brzi algoritam za generaciju kontinuiranih kartograma

Kartogrami su poznata tehnika za prikaz geografski vezanih statističkih informacija kao što su populacijski, demografijski i epidemiologijski podaci. Cilj kartograma je da se deformira mapa tako da se promjene veličine regije u odnosu na neki statistički parametar, ali tako da karta ostane prepoznatljiva. Pokazat će se kako, u općem slučaju, nisu rješive čak i jednostavne varijante generacije kartograma. Heuristika je potrebna za rješavanje jer su izvedive varijante problema NP rješive. Da bi kartogram bio prepoznatljiv, važno je da se očuva globalni oblik ili da se obilježi ulazna mapa. Radi toga ciljna funkcija za iscrtavanje kartograma uključuje očuvanje globalnog i lokalnog oblika. Za mjerenje stupnja očuvanja oblika predlaže se funkcija za sličnost oblika koja je bazirana na razlici između funkcija zakrivljenosti poligona. Brzina izvođenja u ovoj aplikaciji nije toliko važna jer je dovoljno da se algoritam za izračun kartograma obavi jednom nakon unošenja novih podatkovnih točaka, ali se ovom metodom postižu veće brzine izvođenja nego što je to kod nekih drugih tehnika. Ova metoda se obavlja uz pomoć učinkovitog iterativnog algoritma linije uzorkovanja (engl. Scanline Algorithm) za pomicanje vrhova uz očuvanje lokalnog i globalnog oblika. Linije uzorkovanja se mogu generirati automatski ili se mogu unositi interaktivno kako bi se bolje vodila optimizacija procesa. Algoritam se primjenjuje na nekoliko primjera različitih podatkovnih točaka (pogledati 6. poglavlje).

5.1. Kartogrami

Kartografi i geografi su morali koristiti kartograme još mnogo ranije nego što su računala bila dostupna za prikazivanje. Osnovna ideja za kartograme je da se deformira mapa tako da se promijene veličine regija po nekom geografski određenom parametru. Zbog toga što su kartogrami kompliciran za izradu rukom, izučavanje programa koji bi ih crtali postaje interesantno.

Kartogram se može gledati kao generalizacija običnih geografskih karti. Kod takvih karti se vektor statističkih parametara namjeravane veličine regija kartograma zadaje tako da je on proporcionalan s površinama tih regija. U ovom slučaju vektor statističkih parametara se zadaje brojem podatkovnih točaka koje ta regija sadrži jer su one često koncentrirane u gusto naseljenim područjima, a malo ih se širi kroz rijetko naseljena područja. Tako kartogrami zapravo prikazuju površine regija u relaciji s nekim dodatnim parametrom, kao što je to naseljenost ili broj podatkovnih točaka. Uzorci se tada mogu prikazati u proporciji s tim parametrom umjesto čiste veličine područja. Na slici 5.1 je prikazan kartogram baziran na populaciji koji prikazuje da se mogu dobiti različiti utisci pogledom na kartogram u usporedbi s originalnim geografskim kartama.



Slika 5.1: Kartogram osnovan na podacima o naseljenosti.
Na lijevoj strani je originalna karta, a na desnoj kartogram.

Da bi kartogram bio efektivan, čovjek mora moći brzo razumjeti prikazane podatke i povezati ih s originalnim geografskim modelom. Prepoznatljivost ovisi o osnovnim svojstvima očuvanja kao što su to oblik, orijentacija i udaljenost. Ovo je, naime, teško postići u općem slučaju jer je čak, kako će biti kasnije prikazano, nemoguće zadržati originalnu topologiju mape. Većina dostupnih algoritama je veliki potrošač vremena zbog toga što je generacija kontinuiranih kartograma uz simultanu optimizaciju ovih ciljeva dosta kompliciranа.

Krajnji cilj je da se kartogramom ostvare veličine područja proporcionalne s brojem podatkovnih točaka u njima. Razlog tome je da se područja s gustom raspodjelom podatkovnih točaka prošire kako bi svaka pojedinačna podatkovna točka bila vidljiva bez promjene mjerila mape. SCADA sustavi zahtijevaju mogućnost da u svakom trenutku bude prikazana mapa u cijelosti kako bi promjene bile vidljive na ekranu bez obzira na kojem području se dogodile.

5.2. Crtanje kartograma

Ovdje se govori o nekoliko osnovnih koncepata koji sačinjavaju crtanje kartograma. Prvo se formalno definira nekoliko varijanti problema. Zatim slijedi diskusija o složenosti i teoretskim ograničenjima potencijalnih rješenja. Na kraju se ističu neka ključna zapažanja koje su baza za efektivno i učinkovito rješenje – CartoDraw[1].

5.2.1. Definicija problema

Pod pretpostavkom da je ulaz karta definirana skupinom spojenih jednostavnih poligona (poligonalna mreža) $\mathcal{P}$ i vektor parametara $\vec{X}$ koji sadrži željene vrijednosti površine svakog od poligona. Cilj je kreirati kontinuirani kartogram i radi toga je željeni izlaz također skupina spojenih jednostavnih poligona $\bar{\mathcal{P}}$. Neka je $p \in \mathcal{P}$, $|p|$ označava broj vrhova, $A(p)$ površinu i $S(p)$ oblik poligona p, a neka $T(\mathcal{P})$ označava topologiju skupa poligona. Tada se idealno rješenje crteža kontinuiranog kartograma može opisati ovako:

Definicija 1 (Kontinuirani kartogram – idealno rješenje).

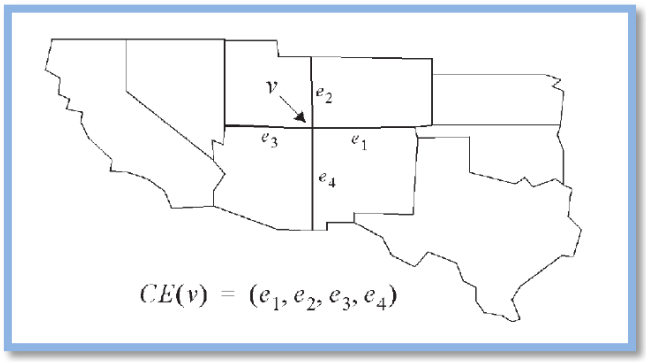
Kontinuirani kartogram skupine spojenih poligona $\mathcal{P} = \{p_1, \dots, p_k\}$ je, uz dotične parametre sačuvane u vektoru $\vec{X} = \{x_1, \dots, x_k\}, (\forall j\ xj > 0)$, vizualizacija transformiranog skupa poligona $\bar{\mathcal{P}}$ za koje vrijedi:

$$T(\bar{\mathcal{P}}) = T(\mathcal{P}) \quad \text{(očuvanje topologije)} \quad (1)$$
$$S(\overline{p_j}) = S(p_j), \forall j = 1, \dots, k \quad \text{(očuvanje oblika)} \quad (2)$$
$$A(\overline{p_j}) = \tilde{x_j}, \forall j = 1, \dots, k \quad \text{(promjena površine)} \quad (3)$$

Željena površina $\tilde{x_j}$ poligona p_j se definira kao:

$$\tilde{x_j} = x_j \cdot \frac{\sum_{i=1}^{k} A(p_j)}{\sum_{i=1}^{k} x_j} \quad (4)$$

Radi jednostavnosti pretpostavlja se da imamo samo jednu skupinu spojenih poligona (kao što je kontinentalni dio SAD-a), a ne više ne spojenih poligona (kao što je karta svijeta). Neka je v_j^i označava i-ti vrh poligona p_j, α_j^i kut i-tog vrha, e_j^i i-ti rubi, $|e_j^i|$ duljinu ruba e_j^i i $CE(v)$ kružni poredak rubova u odnosu na vrh v (vidi sliku 5.2).



Slika 5.2: Kružni poredak rubova[1].

Uz pretpostavku da transformirani poligon ima isti broj vrhova (npr. $|\overline{p_i}| = |p_i|$), tada je jedan način za ostvarenje uvjeta očuvanja topologije i oblika:

Definicija 2 (Očuvanje topologije – očuvanje spajajućih vrhova).

Očuvanje topologije $T(\bar{\mathcal{P}}) = T(\mathcal{P})$ znači da je za svaki vrh $v \in \mathcal{P}$, kružni poredak rubova ostaje isti kao u $\mathcal{P}$. Formalnije, $\forall v_j^i \in \mathcal{P}, j = 1, \dots, k; i = 1, \dots, |p_j| : \exists \overline{v_j^i} \in \bar{\mathcal{P}}, j = 1, \dots, k; i = 1, \dots, |\overline{p_j}|$

$$CE\left(v_i^j\right) = CE(\overline{v_i^j}) \tag{5}$$

Definicija 3 (Očuvanje oblika – očuvanje omjera duljina rubova i kutova).

Očuvanje oblika $S(\overline{p_i}) = S(p_i)$ znači da su sačuvani omjeri duljina rubova poligona i kutovi među njima.

1.

$$\forall j = 1, \dots, k \; \exists c_j \in \mathbb{R} : \left|\overline{e_j^i}\right| = c_j |e_j^i| \tag{6}$$

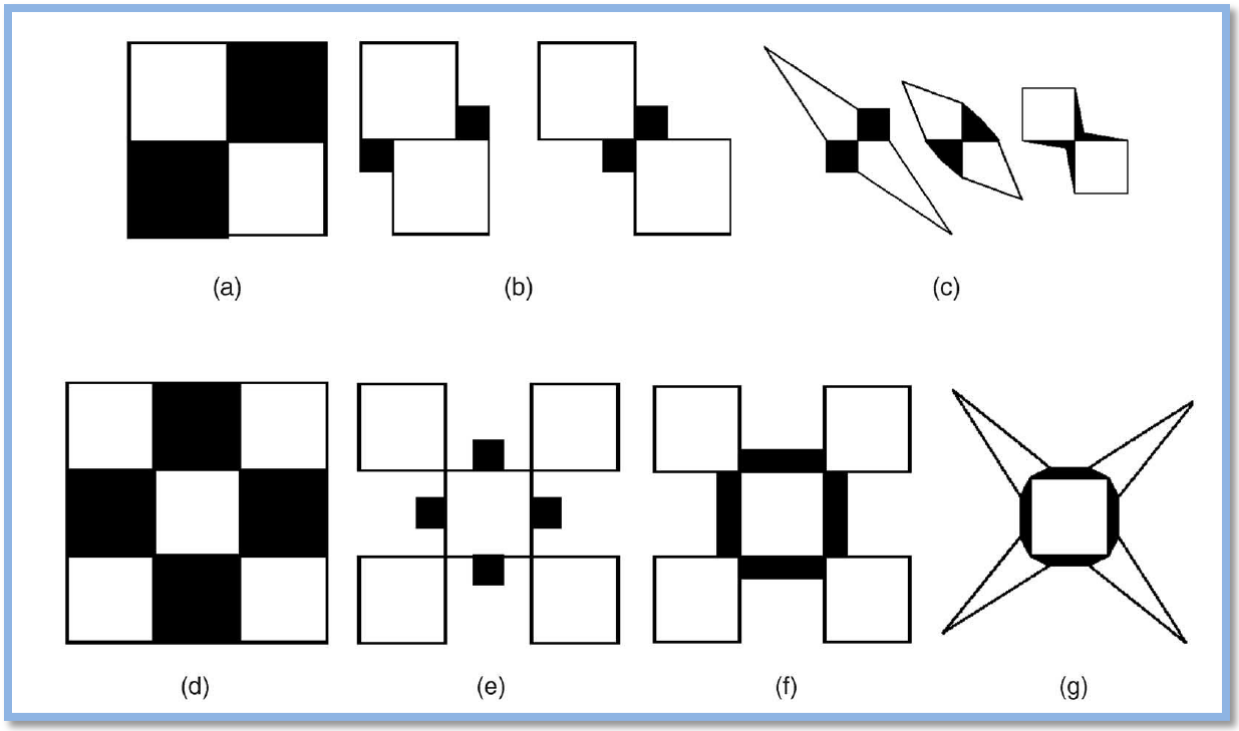
$$i = 1, \dots, |p_j|, e_j^i \in \mathcal{P}, \overline{e_j^i} \in \bar{\mathcal{P}} \tag{7}$$

2.

$$\forall j = 1, \dots, k, \forall i = 1, \dots, |p_j| : \overline{\alpha_j^i} = \alpha_j^i \tag{8}$$

Slijedi jednostavni primjer. Pretpostavka je da imamo mapu s topologijom šahovske ploče (slika 5.3) i želi se promijeniti veličinu mape prema boji polja tako da se skaliraju bijela polja faktorom 1.5, a crna

polja faktorom 0.5. Ovakvo skaliranje je nemoguće bez promjene topologije ili oblika. Tako da je, u općem slučaju, nemoguće postići idealno rješenje. Ovo zapažanje zapisujemo idućom lemom.



Slika 5.3: Primjer šahovske ploče. (a) 2x2 polja, (b) promjena topologija, (c) promjena oblika (d) 3x3 polja, (e) promjena topologije, (f) promjena topologije i oblika, (g) promjena oblika[1].

Lema 1 (Nemogućnost idealnog rješenja).

Problem crtanja kartograma po Definiciji 1 nije moguće riješiti u općem slučaju jer postoji skup poligona i vektora parametara takvih da je nemoguće doći do idealnog rješenja.

Dokaz: Na slici 5.3 su prikazani primjeri skupova poligona koji nemaju idealno kartogramsko rješenje prema Definiciji 1.

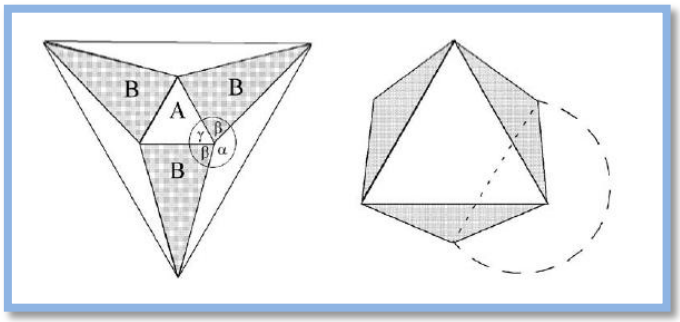
Da bi problem bio rješiv moraju se olabaviti uvjeti očuvanja u nekim svojstvima. Ako je topologija najvažnije svojstvo koje je potrebno održati tada su jedini preostali uvjeti za olabaviti očuvanje oblika i površine. Postoji mnogo načina da se ovo izvede. Problem se može razmatrati uz pomoć dvije funkcije različitosti – funkcija različitosti površine (koja mjeri razliku površine poligona od željene veličine) i funkcija različitosti oblika (koja mjeri sličnost dvaju oblika). U tablici 5.1 su nabrojani mogući uvjeti. Prvi stupac sadrži uvjet koji dopušta maksimalnu razliku svakog od poligona, drugi stupac sadrži uvjet maksimalne razlike svih poligona, a treći stupac sadrži uvjet da suma razlika bude minimalna. Kombiniranjem različitih uvjeta za očuvanje površine i oblika iz tablice 5.1 možemo konstruirati razne varijante problema crtanja kartograma. Korisna kombinacija bi bila, na primjer, rješenje gdje oblik svakog poligona ima barem određenu sličnost svog originalnog oblika, a da je zbroj svih razlika površina minimalan. U nastavku slijedi razmatranje raznih varijanti problema i njihovih složenosti.

Tablica 5.1: Mogući uvjeti za crtanje kartograma[1].

Uvjeti očuvanja	Jedan poligon	Svi poligoni	Minimum
Površine	$\forall j: d_A\left(A(\overline{p}_j), \widetilde{x}_j\right) \leq \varepsilon$	$\sum d_A\left(A(\overline{p}_j), \widetilde{x}_j\right) \leq \varepsilon$	$\sum d_A\left(A(\overline{p}_j), \widetilde{x}_j\right) \overset{!}{\rightarrow} min$
Oblika	$\forall j: d_s\left(S(\overline{p}_j), S(p_j)\right) \leq \varepsilon$	$\sum d_s\left(S(\overline{p}_j), S(p_j)\right) \leq \varepsilon$	$\sum d_s\left(S(\overline{p}_j), S(p_j)\right) \overset{!}{\rightarrow} min$

5.2.2. Rješivost i složenost problema

Kako je pokazano Lemom 1, u općem slučaju je nemoguće pronaći idealno rješenje za problem iscrtavanja kartograma. Ako se sad razmotre varijante koje mogu biti konstruirane kombinacijom uvjeta u Tablici 5.1, ispada da je mnogo takvih također nerješivo u općem slučaju.



Slika 5.4: Problem crtanja kartograma bez rješenja[1].

Lema 2 (Nemogućnost rješenja varijanti problema).

Bilo koja varijanta problema iscrtavanja kartograma koja ima uvjet očuvanja površine jednog poligona ili uvjet očuvanja površine svih poligona je u općem slučaju nerješivo, npr. postoji skup poligona $\mathcal{P}$ i vektor parametara $\vec{X}$ takav da, za bilo koji ε, varijante problema nemaju odgovarajuće rješenje uz očuvanje topologije.

Dokaz: Na slici 5.4 (lijevi crtež) prikazan je primjer simetričnog kartograma koji sadrži sedam poligona. Ako vektor parametara za skaliranje poligona zahtijeva da bijeli poligoni postanu veći, a crni manji, lako možemo konstruirati nemogući slučaj. Zbog simetrične konstrukcije poligona, bez gubljenja općenitosti, možemo pretpostaviti da je jedan kut $\gamma \leq \frac{\pi}{3}$.

Od tuda,

$$\alpha = 2\pi - 2\beta - \gamma \geq 2\pi - 2\beta - \frac{\pi}{3}. \tag{9}$$

Zbog gore spomenutih zahtjeva za promjenu veličine (trokut A vrlo velik i trokuti B vrlo mali), $\beta \rightarrow 0$ i zato je,

$$\alpha \geq 2\pi - \frac{\pi}{3} = \frac{5}{3}\pi \quad \Rightarrow \quad \alpha > \pi \tag{10}$$

čime topologija ne može biti očuvana kako je prikazano slikom 5.4 (desni crtež).

Ovo znači da su rješive samo one varijante problema koje koriste uvjet za minimalnu razliku površina. Rješivost ovog slučaja je trivijalna budući da postoji barem jedno rješenje koje ima savršeno očuvanje oblika, ali izrazito lošu vrijednost za razliku površina. No, određivanje dobrog rješenja s minimalnom razlikom površina je računski težak problem.

Korištenjem ove varijante problema lako se opaža da postoji malo slobode da se poboljša drugi najvažniji parametar – oblik. U većini slučajeva će se uvjetom da razlika površina bude minimalna dobiti neka rješenja koja nam za funkciju razlike površina daju vrlo male iznose, ali ne uzimaju sličnost oblika u obzir. Moglo bi postojati, na primjer, rješenje koje puno bolje zadržava oblike, ali je malo lošije u postizanju željene površine. Kako bi se dopustilo da uvjet očuvanja oblika ima utjecaj na rješenje, ograničenja se moraju prilagoditi. U principu postoje dvije mogućnosti. Prva je da se odredi minimalna moguća razlika površine i tada da se dopusti određena maksimalna devijacija od te minimalne razlike za pronalaženje najboljeg oblika. Formalnije ovo može biti definirano slijedećom definicijom.

Definicija 4 (Varijanta 1 problema kontinuiranog kartograma).

Dan je skup poligona $\mathcal{P}$, vektor parametara $\vec{X}$, i vrijednost pogreške ε, problem kontinuiranog kartograma može biti definiran kao transformirani skup poligona $\overline{\mathcal{P}}$ za koje vrijede slijedeća dva uvjeta:

$$\sum_{j=1}^{k} d_A\left(\tilde{x_j}, A(\overline{p_j})\right) \leq {}^{MIN}_{\bar{\mathcal{P}}}\left(d_A\left(\tilde{x_j}, A(\overline{p_j})\right)\right) + \varepsilon \tag{11}$$

$$\sum_{j=1}^{k} d_A\left(S(p_j), S(\overline{p_j})\right) \overset{!}{\rightarrow} min \tag{12}$$

Druga mogućnost je da se normaliziraju razlike površina i oblika te da se koristi težinska sredina normaliziranih razlika kao kriterij optimizacije.

Definicija 5 (Varijanta 2 problema kontinuiranog kartograma).

Dan je skup poligona $\mathcal{P}$, vektor parametara $\vec{X}$, i vrijednost pogreške i težinski faktori važnosti za razlike površine i oblika, problem kontinuiranog kartograma može se definirati kao transformirani skup poligona $\overline{\mathcal{P}}$ za koji vrijedi:

$$a \cdot \sum_{j=1}^{k} d_A\left(\tilde{x_j}, A(\overline{p_j})\right) + b \cdot \sum_{j=1}^{n} d_S\left(S(p_j), S(\overline{p_j})\right) \overset{!}{\rightarrow} \min_{a,b \geq 0}. \tag{13}$$

Postoje druge značajne i rješive varijante problema koje, na primjer, razmatraju ograničenja jednog poligona (vidi tablicu 5.1). Većina trenutno dostupnih algoritama pokušava riješiti problem prema definiciji 4 ili definiciji 5. Ovo se čini dostatno za neke aplikacije, ali neke aplikacije i dalje imaju potrebe za dodatnim ograničenjima.

5.2.3. Važna opažanja

Postoje dva velika problema kod dostupnih algoritama: prvo, velika vremenski zahtjevna složenost algoritma ograničava njihovu iskoristivost na statične aplikacije s manjim brojem poligona i vrhova. Drugo, imaju vrlo ograničeno očuvanje oblika. Globalni oblik je jedan od najvažnijih faktora da bi kartogrami bili efektivni i sigurno je barem toliko važno očuvanje njegovog oblika kao i očuvanje oblika unutarnjih poligona. U CartoDraw algoritmu za crtanje kartograma, uz uvjete očuvanja oblika i površine također se eksplicitno uključuje i uvjet očuvanja globalnog oblika. Neka $G_r(\mathcal{P})(r = 1, \dots, l, l \leq k)$ označava skup globalnih poligona koji mogu biti izvedeni iz skupa poligona $\mathcal{P}$. Uvjet očuvanja globalnog oblika se može formalno opisati kako je zapisano u tablici 5.2. Konačna definicija problema kartogramskog crteža koristi težinski zadane uvjete minimalizacije razlike površina, očuvanje oblika i globalnog oblika.

Tablica 5.2: Ograničenja za globalnog poligona[1].

Jedan poligon	**Svi poligoni**	**minimum**
$\forall r : d_S\left(S\left(G_r(\overline{\mathcal{P}})\right), S(G_r(\mathcal{P}))\right) \leq \varepsilon$	$\sum d_S\left(S\left(G_r(\overline{\mathcal{P}})\right), S(G_r(\mathcal{P}))\right) \leq \varepsilon$	$\sum d_S\left(S\left(G_r(\overline{\mathcal{P}})\right), S(G_r(\mathcal{P}))\right) \overset{!}{\rightarrow} min$

Definicija 6 (Problem kontinuiranog kartograma).

Dan je skup poligona $\mathcal{P}$, vektor prametara $\vec{X}$ i faktori važnosti ograničenja za površinu, oblik i globalni oblik a, b i c. Problem kontinuiranog kartograma se može definirati kao transformirani skup poligona $\overline{\mathcal{P}}$ za koji vrijedi:

$$a \cdot \sum_{j=1}^{k} d_A\left(\tilde{x}_j, A(\overline{p_j})\right) + b \cdot \sum_{j=1}^{k} d_S\left(S(p_j), S(\overline{p_j})\right) + c \cdot \sum_{r} d_S\left(S\left(G_r(\overline{\mathcal{P}})\right), S(G_r(\mathcal{P}))\right) \overset{!}{\rightarrow} \min_{a,b,c \geq 0} . \quad (14)$$

Sada slijedi pregled važnijih zapažanja, ključnih za djelotvorno rješenje problema. Jedno od njih je da je samo nekoliko vrhova zapravo važno za definiciju oblika poligona. Kad se gleda, na primjer karta SAD-a, vidimo da je mali broj vanjskih i unutarnjih vrhova zapravo važno te da eliminacijom nevažnih vrhova ne gubimo veliku, čovjeku primjetljivu, razliku oblika. Zamjećuje se također da važnost poligona i njegovih vrhova uvelike ovisi o njihovoj veličini (koja je direktno vezano za vektor parametara) i o duljini rubova i kutova između rubova. Ovim se činjenicama u CartoDraw algoritmu daje velika pažnja te se određuju važnosti vrhova bazirano na tim zapažanjima. Drugo zapažanje je da, kako bi dobili dobre rezultate, greška oblika mora biti eksplicitno kontrolirana. Zadnje zapažanje je da velika vremenska složenost većine ostalih algoritama nastaje zbog kompleksne, vremenski zahtjevne, optimizacije. No u većini je slučajeva moguće lokalno pomaknuti vrhove i poboljšati pogrešku površine uz očuvanje oblika. Kako bi se došlo do dobrog rješenja, CartoDraw algoritam

iterativno pomiče vrhove prema lokalnim mjerenjima uz pomoć linija uzorkovanja uz eksplicitnu kontrolnu funkcije pogreške oblika.

5.3. Algoritam za crtanje kartograma

Glavni cilj CartoDraw algoritma za crtanje kartograma je brza generacija kartograma prihvatljive kvalitete. Zbog činjenice da ulazne karte često imaju više vrhova nego što je potrebno da se izračunaju dobri kartogrami, prvi je korak inteligentno odbacivanje točaka. Zatim slijedi određivanje linija uzorkovanja preko kojih se iterativno izvršava algoritam. One mogu biti određene automatski (horizontalne i vertikalne linije) ili mogu biti interaktivno unošene ljudskom rukom. Tada slijedi CartoDraw algoritam. U svakom koraku algoritma se oblik modificirane poligonalne mreže kontrolira funkcijom pogreške oblika.

5.3.1. Redukcijski algoritmi

Redukcijski algoritmi rubova se koriste za rješavanje kartogramskih problema u raznim ponuđenim algoritmima. Očuvanje globalnog oblika je vrlo važno pri stvaranju prepoznatljivih kartograma. Korišteni redukcijski algoritam uzima to u obzir tako da pojednostavljuje globalne i unutarnje poligone na isti način.

5.3.1.1. Redukcija globalnog poligona

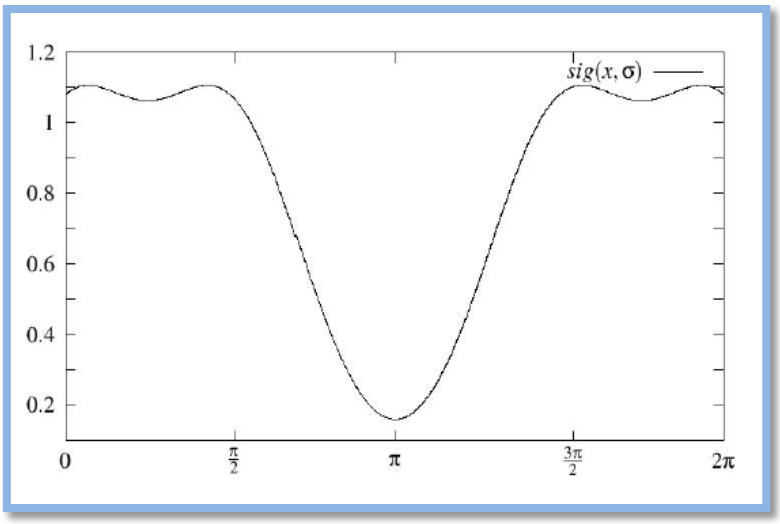
Ključno zapažanje je da važnosti vrhova poligona variraju u velikim omjerima. Vrhovi s kutovima blizu 180 stupnjeva i oni s kratkim rubovima nisu toliko značajni za oblik poligona dok oni s oštrim kutovima ili dugim rubovima imaju značajan učinak. Osnovna ideja algoritma redukcije globalnog poligona je da se ocijene važnosti svakog vrha prema tom kriteriju. Tada se iterativno manje važni vrhovi uklanjaju. Uklanjaju se samo oni vrhovi koji ne pripadaju većem broju poligona kako bi se održala topologija. Kako bi ostvarili algoritam redukcije prvo se definira notacija važnosti vrha na slijedeći način:

$$I(v) = Sig(\alpha^v) \cdot |e_1^v| \cdot |e_2^v|, \tag{15}$$

gdje su e_1^v i e_2^v dva ruba vrha v, a $Sig(\alpha^v)$ je funkcija koja vraća vrijednost važnosti kuta α^v vrha v. Funkcija važnosti $Sig(\alpha^v)$ je bitna jer različiti kutovi imaju drugačiji utjecaj na oblik poligona. Oštri kutovi i kutovi koji su blizu 90° su važniji za oblik poligona i funkcija važnosti zato dodjeljuje veće vrijednosti oštrim kutovima i manje vrijednosti tupim kutovima. U ovom algoritmu se koristi slijedeća funkcija:

$$Sig(\alpha) = \sum_{\mu \in \{0°, 90°, 270°, 360°\}} exp \frac{(\alpha - \mu)^2}{2\sigma^2} \tag{16}$$

Ona ima šiljke za $\alpha = 0°, 90°, 270°, 360°$ i blizu je nule za $\alpha = 180°$. Domena funkcije je $\alpha \in [0°, 360°]$, a σ je odabran da bude $0.2 \cdot \pi$. Slika 5.5 prikazuje težinsku funkciju važnosti kuta.



Slika 5.5: Težinska funkcija važnosti kuta[1].

Za ostvarenje algoritama redukcije vrhova globalnog poligona mora se definirati globalni poligon kao podskup vrhova iz $\mathcal{P}$. Za svaki poligon $p_j, j = 1, \dots, k$, dijelove globalnog poligona $\mathcal{GP}$ se može definirati kao:

$$gp_j = \{v \in p_j : |edges(v)| > |polygons(v)|\} \tag{17}$$

Globalni poligon se definira kao $\mathcal{GP} = \bigcup_{j=1,\dots,k} gp_j$. Algoritam za redukciju vrhova globalnog poligona je prikazan algoritmom 5.1. Može se vidjeti da se kao kandidati za uklanjanje razmatraju samo oni vrhovi koji ne pripadaju većem broju poligona (pogledati inicijalizaciju V u algoritmu 5.1.) i uklanjaju se samo ako je izazvana razlika površina manja od zadane vrijednosti $MaxAreaDiff$.

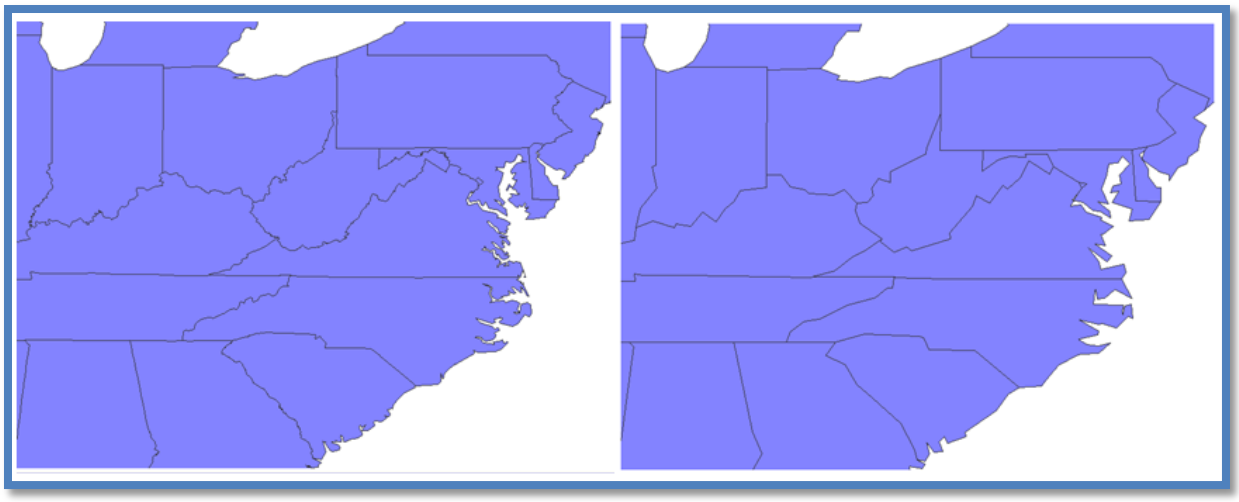
$RedukcijaVrhovaGlobalnogPoligona(\mathcal{P}, \mathcal{GP}$, MaxAreaDiff) {
    $V = \{v \in gp_j \,|\, v \notin gp_m \; za \; m \neq j\}$ \\ Razmatraju se vrhovi koji pripadaju jednom polig.
    radi {
        $\overline{v} = \{v \in V | \underset{v \in V}{MIN} I(v)\}$ \\ Određuje se najmanje važan vrh
        $j = \{j \in \{1 \dots k\} | \overline{v} \in p_j\}$ \\ Određuje se poligon koji sadrži taj vrh
        Ako ($\|A_s(p_j \setminus \{\overline{v}\}) - A_s(p_j)\| \leq MaxAreaDiff$)
            $p_j = p_j \setminus \{\overline{v}\}$
        $V = V \setminus \{\overline{v}\}$
    } dok $V \neq \{\}$
}

Algoritam 5.1: Algoritam redukcije vrhova globalnog poligona.

5.3.1.2. Redukcija vrhova unutarnjih poligona

Za redukciju unutarnjih vrhova može se ponovo koristiti algoritam korišten pri uklanjanju vrhova globalnog poligona uz drugačiju inicijalizaciju skupine vrhova V koje razmatramo za uklanjanje. Vrhovi koji pripadaju toj skupini su svi unutarnji vrhovi koji nisu spajajući odnosno oni koji pripadaju samo dvama poligonima.

Na slici 5.6 se vidi karta prije i nakon redukcije globalnih i unutarnjih vrhova.



Slika 5.6: Primjer redukcijskog algoritma.

5.3.2. CartoDraw algoritam

Osnovna ideja CartoDraw algoritma je da inkrementalno pomiče vrhove uz pomoć serije linija uzorkovanja. Primjer linije uzorkovanja prikazan je slikom 5.8. Linija uzorkovanja je linijski segment proizvoljne duljine i pozicije. Svaka linija uzorkovanja određuje područja djelovanja koja su okomita na tu liniju. Sve točke unutar područja djelovanja se pomiču u jednom koraku. Za svako se područje djelovanja linije uzorkovanja određuje skalirajući faktor poligona koji se dobije prema faktoru pogreške tih poligona. Vrhovi se tada pomiču prema faktoru skaliranja poligona i udaljenosti od linije uzorkovanja. Pomicanje vrhova može biti paralelno ili okomito u odnosu na liniju uzorkovanja. Ako pogreška oblika nastala zbog djelovanja linije uzorkovanja prelazi neku graničnu vrijednost, pomicanje vrhova se odbacuje.

Linije uzorkovanja trebaju biti upotrijebljene na dijelove mape gdje je površinska pogreška velika i postoji potencijal za poboljšanjem. Jednostavni pristup za generaciju linija uzorkovanja je da se koriste horizontalni i vertikalni linijski segmenti. No mnogo se bolji rezultati mogu postići ručnim postavljanjem linija uzorkovanja vođeno oblicima ulaznih poligona i lokalnim potencijalom za poboljšanje. Inkrementalno pomicanje vrhova po liniji uzorkovanja je malo u usporedbi s očekivanim promjenama u području. To znači da istu liniju uzorkovanja treba upotrijebiti više puta kako bi se došlo do velikih promjena u području.

Prije opisa glavnog CartoDraw algoritma moraju se definirati njegove tri glavne komponente – funkcija površinske pogreške, funkcija sličnosti oblika, i algoritam djelovanja linije uzorkovanja.

5.3.2.1. Funkcija površinske pogreške

Cilj generacije kartograma je da se postigne skup poligona kojima površina odgovara vrijednostima danim vektorom parametara $\vec{X}$. U svakom koraku algoritma je potrebna funkcija površinske pogreške kako bi se odredilo smanjenje površinske pogreške postignuto djelovanjem dane linije uzorkovanja. Relativna površinska pogreška E_{rel}^{j} poligona p_j se može računati kao:

$$E_{rel}^{j} = \frac{\left| A_{željena}^{j} - A_{trenutna}^{j} \right|}{A_{željena}^{j} + A_{trenutna}^{j}} \tag{18}$$

Odavde je površinska pogreška za skupinu poligona $\mathcal{P}$ definirana kao:

$$E_{rel}^{\mathcal{P}} = \sum_{j=1}^{k} \left(E_{rel}^{j} \cdot \frac{A_{željena}^{j}}{\sum_{j=1}^{k} A_{željena}^{j}} \right) \tag{19}$$

5.3.2.2. Funkcija pogreške oblika

Uz smanjivanje površinske pogreške, proces generacije kartograma također pokušava održati originalne oblike. Da bi se ocijenilo očuvanje oblika potrebna je funkcija za određivanje sličnost oblika koja uspoređuje novi oblik poligona s njegovim originalnim oblikom. Definiranje korisne funkcije za sličnost oblika je težak problem budući da bi mjera sličnosti trebala biti neovisna o translaciji, skaliranju i rotaciji objekta. Kako bi se dobila neovisnost za navedene transformacije razmatra se funkcija zakrivljenosti poligona (umjesto samih poligona) uz normalizaciju opsega poligona na 2π. Korištenje funkcije zakrivljenosti garantira translacijsku i rotacijsku neovisnost.

U nastavku je pretpostavka da su poligoni transformirani u normaliziranu funkciju zakrivljenosti poligona u opsegu $p : [0, 2\pi] \rightarrow \Re^2$. Tada možemo definirati funkciju zakrivljenosti poligona C definirati kao:

$$C : (\Re \rightarrow \Re^2) \rightarrow (\Re \rightarrow \Re^2) \tag{20}$$

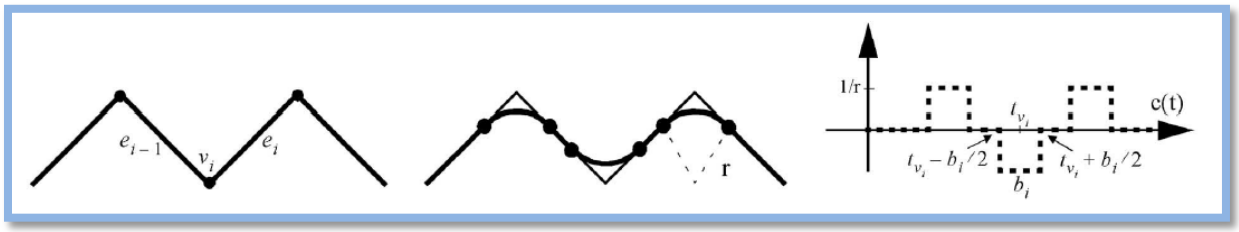
Funkcija sličnost dva poligona p i $\overline{p}$ može biti definirana kao:

$$d_s\big(S(p), S(\overline{p})\big) = d_{Euclid}\big(C(p), C(\overline{p})\big) \tag{21}$$

Slijedi detaljniji opis funkcije zakrivljenosti poligona C.

Određivanje funkcije zakrivljenosti poligona

Funkciju zakrivljenosti poligona matematički nije moguće definirati jer joj druga derivacija nije kontinuirana. Ovaj problem se može izbjeći aproksimacijom poligona tako da se zamjeni njegov svaki vrh s vrlo malim kružnim segmentom, kako je prikazano slikom 5.7. Tako se dobije novi geometrijski objekt kojem je prva derivacija kontinuirana. Zakrivljenost ove strukture definirana je u odlomcima; ulančavajući te odlomke dobijemo zakrivljenost kao pravokutnu valnu funkciju.



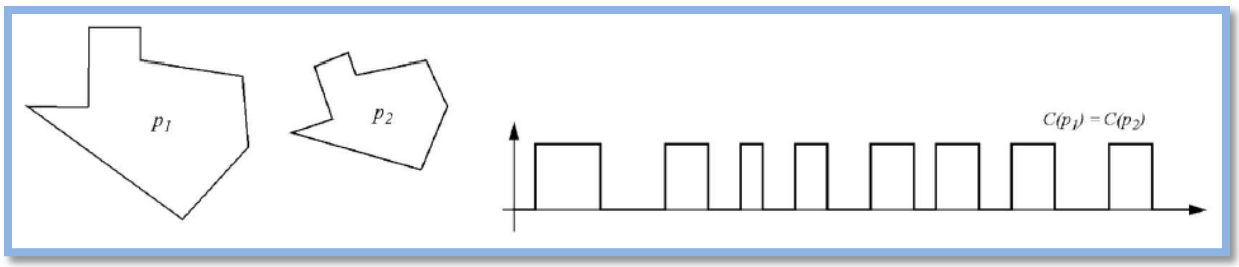
Slika 5.7: Funkcija zakrivljenosti poligona.
Lijevo je originalni poligon, u sredini aproksimirani poligon,
a desno funkcija zakrivljenosti aproksimiranog poligona[1].

Za detaljniji opis funkcije zakrivljenosti poligona razmatraju se dva uzastopna ruba e_{i-1} *i* e_i. Ovi rubovi se nalaze u vrhu v_i pod kutom α_i. Za aproksimirani poligon koji sadrži v_i možemo lako

izračunati funkciju zakrivljenosti $c_i(t)$ zato jer je zakrivljenost kružnog segmenta s radijusom r konstantna funkcija $\frac{1}{r}$, a zakrivljenost ravne linije je konstantna funkcija iznosa nula. Možemo izračunati duljinu luka kružnog segmenta tako da zamijenimo vrh v_i s $b_i = |\alpha_i| \cdot r$. Za $c_i(t)$ tada imamo:

$$c_i(t) = \begin{cases} \frac{1}{r} & ako\ je\ \left(t_{v_i} - \frac{b_i}{2} > t > t_{v_i} + \frac{b_i}{2}\right) \\ 0 & inače \end{cases} \tag{22}$$

Funkcija zakrivljenosti proizvoljnog poligona p je $c(t) = \sum_{k=0}^{|p|-1} c_k(t)$. Slika 5.7 (desno) pokazuje graf funkcije zakrivljenosti $c(t)$ za aproksimaciju dijela poligona na slici 5.7 (lijevo). Slika 5.8 prikazuje funkciju zakrivljenosti dva identična poligona uz neovisnost translacije, rotacije i skaliranja.

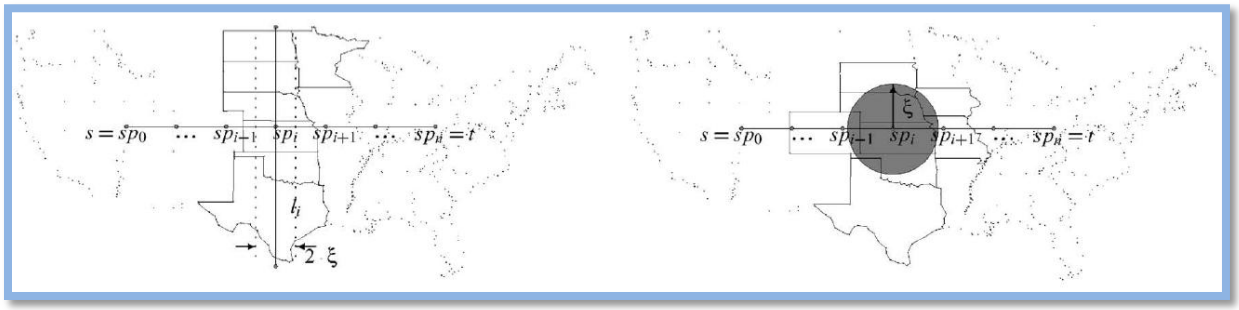


Slika 5.8: Funkcija zakrivljenosti dva identična poligona.
Desni poligon je transformirani lijevi (rotiran, skaliran i translatiran)[1].

Aproksimacija originalnog poligona i, osobito izbor radijusa r, djeluje na funkciju zakrivljenosti. Ako smanjimo radijus r kružnog segmenta, $\frac{1}{r}$ će biti povećan dok će b_i biti smanjen. Zbog toga $c(t)$ postaje oštrija i amplituda pravokutnih valova postaje viša čime aproksimacija poligona $c(t)$ postaje numerički problematična. Vrijednost parametra r koja se pokazala dobrom za ovu aplikaciju je $\frac{\pi}{50}$ za poligone normalizirane duljine 2π. Izbor za r je dobar sve dok je on manji od polovice duljine najkraćeg ruba jer bi se inače individualni pravokutni valovi mogli preklapati.

5.3.2.3. Algoritam linije uzorkovanja

Ključ CartoDraw algoritma je heuristički algoritam linije uzorkovanja koji inkrementalno pomiče vrhove u odnosu na linije uzorkovanja. Linija uzorkovanja sl je linijski segment proizvoljne pozicije i duljine i podijeljena je u n dijelova $\frac{|sl|}{n}$. Točke odlomaka sp_i $(i = 0 \dots n)$ definiraju $n + 1$ područja djelovanja poligonalne mreže, koja su okomita na liniju uzorkovanja (vidi sliku 5.9 lijevo).



Slika 5.9: Primjer algoritma linije uzorkovanja. Lijevo je područje djelovanja sp-a,
a desno ograničeno područje djelovanja[1].

U jednom koraku algoritma linije uzorkovanja se svi vrhovi $v \in V_i$ unutar određene udaljenosti ($\xi = \frac{|sl|}{2n}$) od l_i razmatraju za inkrementalno pomicanje (vidi sliku 5.9 lijevo). Neka SP_i bude skup poligona koji ima barem jedan vrh u području djelovanja i ($i = 0 \dots n$). Faktor skaliranja SF_i određuje se prema površinskoj pogrešci svih poligona p u području i:

$$SF_i = const \cdot \sum_{r \in S_i} \left(\frac{\widetilde{x_r} - A(p_r)}{\widetilde{x_r} + A(p_r)} \cdot \frac{\widetilde{x_r}}{\sum_{l \in S_i}^{n} \widetilde{x_l}} \right) \tag{23}$$

Preostaje još odrediti smjer u kojem će se pomicati vrh $o(v)$ i djelovati faktorom skaliranja SF_i kako bi pomaknuli vrh. Pomicanje se može obaviti ili u smjeru linije uzorkovanja ili u smjeru linije odlomka l_i. Algoritam je prikazan algoritmom 5.2. Područja djelovanja linije uzorkovanja uvijek razapinju cijelo područje okomito na liniju uzorkovanja. Po želji se promjene mogu ograničiti tako da budu lokalne u oba smjera. Poligoni koji se promatraju mogu se opcionalno ograničiti na one koji su blizu linije uzorkovanja (vidi sliku 5.9 desno). Ta opcija neće utjecati na algoritam prikazan algoritmom 5.2.

$Scanline(\mathcal{P}, \tilde{X}, sl)$ {
  za svaki $sp \in \{sp_i | sp_i = s + \frac{i}{n}(t - s), i = (0 \dots n)\}$ { // skup točaka linije uz. koje
    // leže na sl
    $l_i = sp \perp \overrightarrow{sl}$ // linija odlomka
    $V_i = \{v \in V | |l_i - v| \leq \xi\}$ // vrhovi v u području djelovanja i
    // npr. za ξ bliži od l_i
    $SP_i = \{j \in \mathbb{N} | \exists v : v \in V_i \wedge v \in p_j\}$ // indeksi poligona koji sadrže barem
    // jedan vrh iz V_i
    $SF_i = const \cdot \sum_{r \in S_i} \left(\frac{\widetilde{x_r} - A(p_r)}{\widetilde{x_r} + A(p_r)} \cdot \frac{\widetilde{x_r}}{\sum_{l \in S_i}^{n} \widetilde{x_l}} \right)$, // faktor skaliranja
    za svaki $v \in V_i$ {
      ako je (smijer = linija_uzorkovanja)
        $o(v) = \frac{\overrightarrow{v \perp l_i}}{|\overrightarrow{v \perp l_i}|}$
      inače
        $o(v) = \frac{\overrightarrow{v \perp sl}}{|\overrightarrow{v \perp sl}|}$
      $v = v + SF_i \cdot o(v)$
    }
  }
}

Algoritam 5.2: Algoritam linije uzorkovanja za pomicanje vrhova poligona.

5.3.2.4. Glavni CartoDraw algoritam

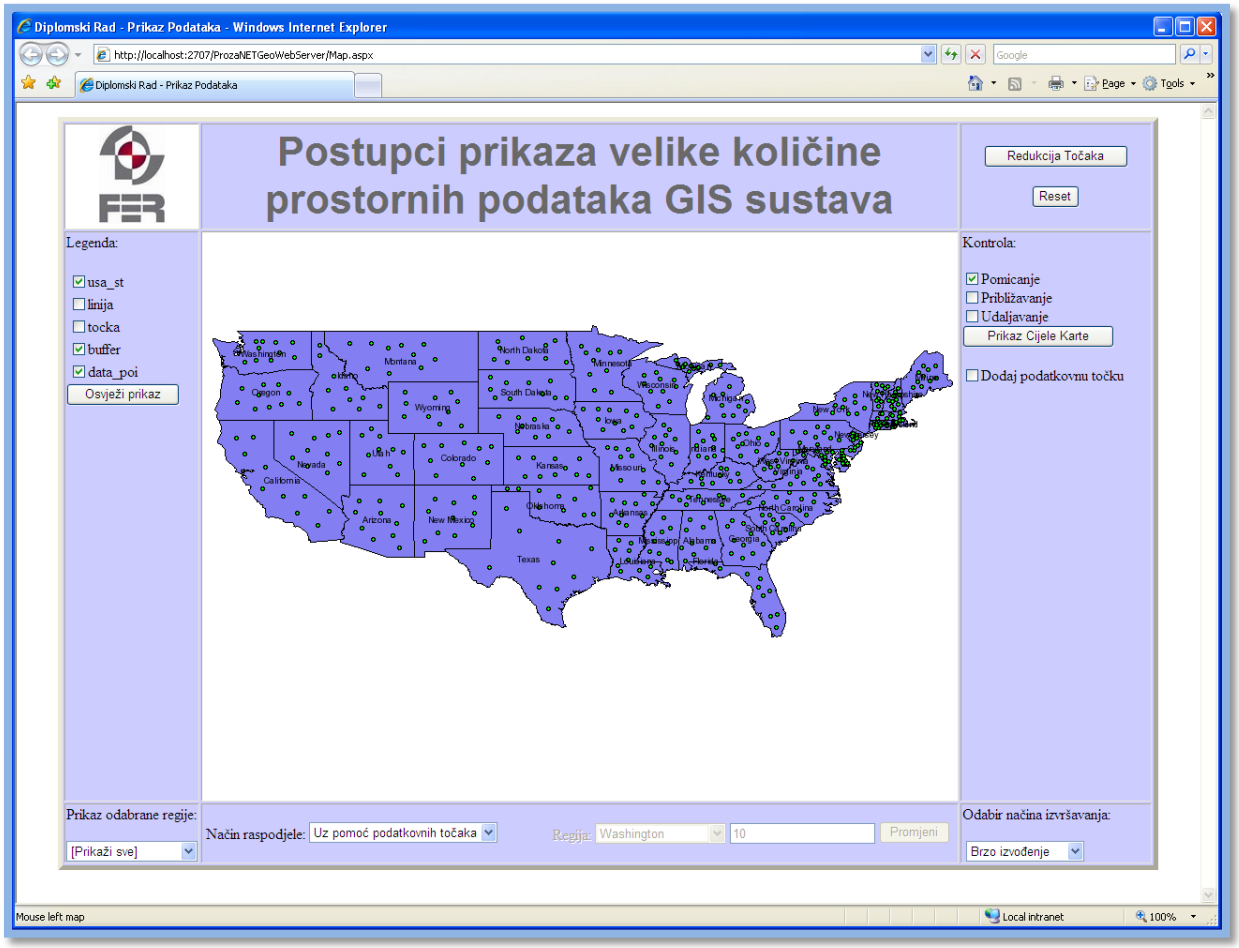
Nakon definiranih glavnih komponenti CartoDraw algoritma može se opisati njegova glavna procedura. Algoritam pretpostavlja za ulaz skup poligona $\mathcal{P}$, vektor za skaliranje željenim statističkim parametrom $\tilde{X}$ i skup linija uzorkovanja SL koje mogu biti određene automatski ili ručno. Izlaz je modificirani skup poligona $\mathcal{P}$ koji predstavlja kartogram. Algoritam radi na slijedeći način (vidi algoritam 5.3). Za svaku liniju uzorkovanja algoritam djeluje transformacijom pomoću linije uzorkovanja i provjerava rezultate. Ako je površinska pogreška E_{rel} do koje se došlo radi transformacije linijom uzorkovanja ispod određene postavljene granice ε_A i ako je distorzija oblika ispod određene granice ε_s tada se promjene zadržavaju, a u suprotnom se odbacuju. Algoritam nastavlja s idućom linijom uzorkovanja sve dok nisu sve linije uzorkovanja djelovale na isti način. Nakon toga algoritam provjerava da li se, pri djelovanju svih linija uzorkovanja, poboljšala površinska pogreška i u slučaju da je, algoritam djeluje sa svim linijama uzorkovanja ponovo i ponavlja proceduru dok se god dolazi do poboljšanja (površinska pogreška ispod ε). Budući da poboljšanje površinske pogreške mora biti pozitivno i iznad granice ε u svakoj iteraciji, površinska pogreška se ujednačeno smanjuje i kraj algoritma je osiguran. Može se primijetiti da se, pri djelovanju individualne linije uzorkovanja, dopušta algoritmu da poveća površinsku pogrešku kako bi se izbjegao lokalni optimum.

$CartoDraw(\mathcal{P}, \tilde{X}, SL)$ {
    Radi {
        $AreaError = AreaDist(\mathcal{P}, \tilde{X})$
        Za svaki ($sl \in SL$) {
            $\overline{\mathcal{P}} = Scanline(\mathcal{P}, \tilde{X}, sl)$
            Ako je ($ShapeDist(\mathcal{P}, \overline{\mathcal{P}}) < \varepsilon_s$ and $\left|AreaError - AreaDist(\overline{\mathcal{P}}, \tilde{X})\right| < \varepsilon_A$)
                $\mathcal{P} = \overline{\mathcal{P}}$
        }
    } dok ($0 < AreaError - AreaDist(\mathcal{P}, \tilde{X}) \leq \varepsilon$)
}

Algoritam 5.3: CartoDraw algoritam.

6. Implementacija

Program za prikaz velike količine prostornih podataka GIS sustava napisan je u programskom jeziku C# i pomoću .NET-a omogućen za pokretanje pomoću Internet preglednika. Korištena je karta SAD-a koja u vektorskom SHP zapisu sadrži listu poligona koji predstavljaju sjedinjene države amerike. Karta je besplatna i dostupna je na adresi http://www.intactforests.org/download/shp_download.htm.

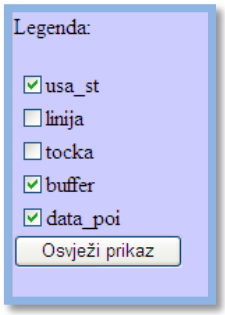


Slika 6.1: Izgled Internet aplikacije.

6.1. Opis sučelja

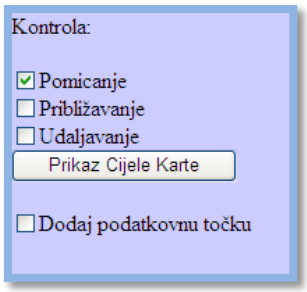
Programsko sučelje (na slici 6.1) sastoji se od sedam dijelova.

Legenda (prikazana slikom 6.2) – nalazi se na lijevoj strani i pomoću kućica je moguće odabrati vidljive slojeve na karti. Klikom miša na tipku *Osvježi prikaz* se obavljene promjene odraze na središnjem prikazu i potrebni slojevi postanu vidljivi (odnosno skriveni). Dostupno je šest slojeva i oni prikazuju redom: originalnu kartu, linije uzorkovanja po kojima se izvodio algoritam, točke linije uzorkovanja kojima se određuju područja djelovanja, kartu na kojoj su izvršene transformacije te podatkovne točke prema kojima je moguće obaviti raspodjelu površina pojedine regije.



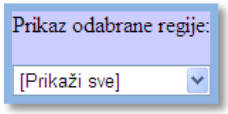
Slika 6.2: Prvi dio sučelja – legenda.

1. Kontrola (prikazana slikom 6.3) – nalazi se na desnoj strani i pomoću nje je moguće obavljati osnovne funkcije: pomicanje po karti (engl. pan), promijeniti mjerilo prikaza (engl. zoom) te vratiti mjerilo na originalnu veličinu. Najvažnija funkcija koja se također nalazi u kontroli je dodavanje podatkovnih točaka. Kad je ova funkcija odabrana moguće je klikom miša na središnjem prikazu dodavati podatkovne točke.



Slika 6.3: Drugi dio sučelja – kontrola.

2. Prikaz odabrane regije (slika 6.4) – nalazi se u donjem lijevom kutu i moguće je, pomoću padajuće liste u kojoj su navedene sve regije po njihovim imenima, odabrati jednu od regija iz *buffer* sloja koju želimo prikazati.

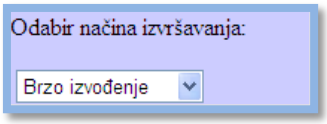


Slika 6.4: Treći dio sučelja – prikaz odabrane regije.

3. Način raspodjele veličina regija (slika 6.5) – nalazi se na dnu aplikacije. Tu je moguće odrediti da li će se raspodjela obaviti po broju podatkovnih točaka koje se nalaze u pojedinoj regiji ili se može ručno upisati broj za pojedinu regiju prema kojima će se proporcionalno obaviti raspodjela.

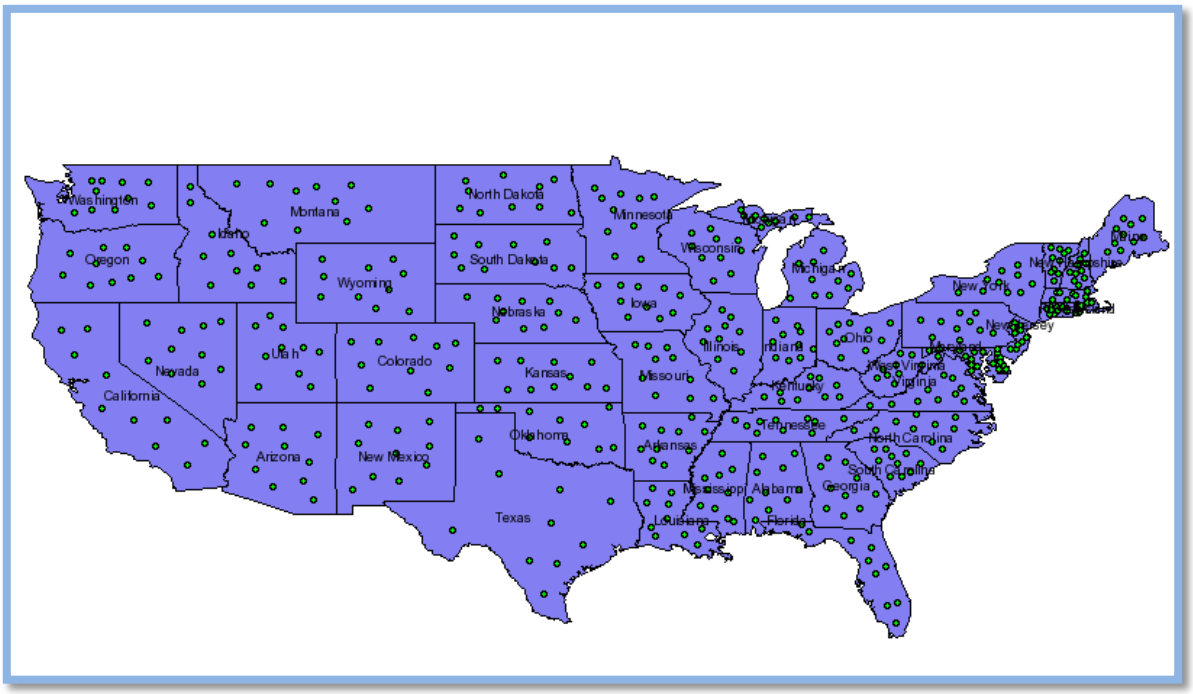
Slika 6.5: Četvrti dio sučelja – način raspodjele.

4. Način obavljanja algoritma (slika 6.6) – nalazi se u donjem desnom kutu. Algoritam je implementiran na dva načina: brzi i kvalitetni. Razlika između njih je način na koji se određuje da li je točka koja određuje područje djelovanja algoritma linije uzorkovanja unutar neke od regija ili nije. Kod brzog algoritma se gleda da li točka pripada graničnom okviru pojedine regije, a kod kvalitetnog se točno određuje da li je točka unutar regije ili ne. Kvalitetnim algoritmom se dobije manja distorzija oblika nego kod brzog, ali, ponekad, i veće odstupanje od traženih površina.



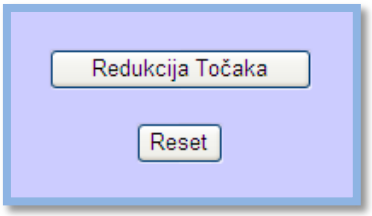
Slika 6.6: Peti dio sučelja – način obavljanja algoritma.

5. Središnji prikaz (slika 6.7) – nalazi se u centralnom dijelu aplikacije i prikazuje slojeve odabrane u legendi.



Slika 6.7: Šesti dio sučelja – središnji prikaz.

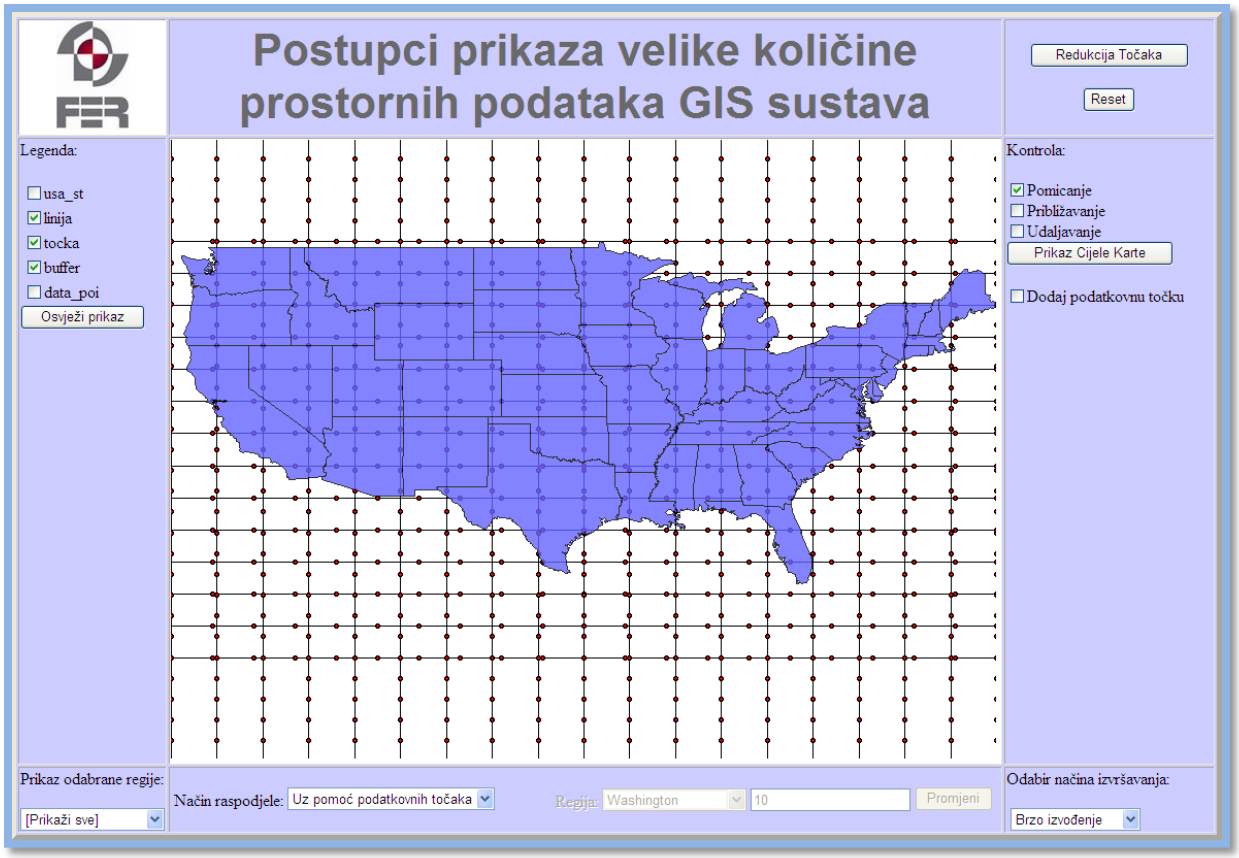
6. Izvršavanje i resetiranje (slika 6.8) – nalazi se u gornjem desnom kutu. Ovdje se pokreće algoritam. Dostupna su dva dugmića: *Redukcija Točaka* i *Reset*. Pritiskom na gumb *Redukcija Točaka* započinje algoritam za izglađivanje poligona i nakon toga se tekst promjeni u *CartoDraw Algoritam*. Ponovnim pritiskom na dugme započinje algoritam za kreiranje kartograma. Pritiskom na tipku *Reset* aplikacija se vrača u početno stanje.



Slika 6.8: Sedmi dio sučelja - izvršavanje i resetiranje.

6.2. Korištenje aplikacije

Nakon pokretanja aplikacije moguće je uključiti sloj na kojem su prikazane linije uzorkovanja postavljene automatski (horizontalne i vertikalne linije) i točke koje određuju područje djelovanja (slika 6.9).



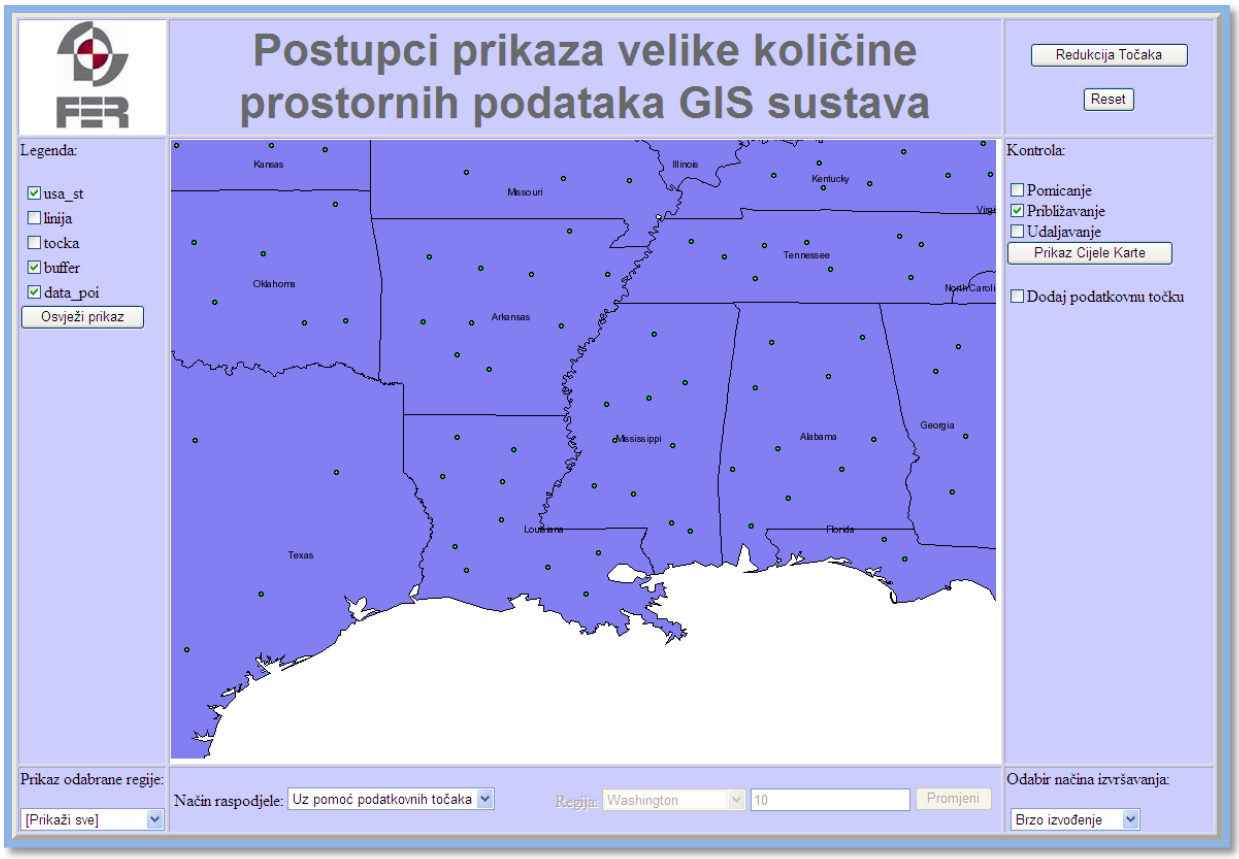
Slika 6.9: Linije uzorkovanja sl i točke područja djelovanja sp korištene u aplikaciji.

Prije pokretanja samog CartoDraw algoritma je potrebno smanjiti broj točaka kako bi se algoritam izvršavao brže. Rezultat algoritma redukcije točaka prikazan je slikom 6.10 (karta prije redukcije) i slikom 6.11 (karta nakon redukcije). Na slikama je uvećana linija s najvećim nazubljenjem (Mississippi) kako bi se što bolje primijetio učinak algoritma.

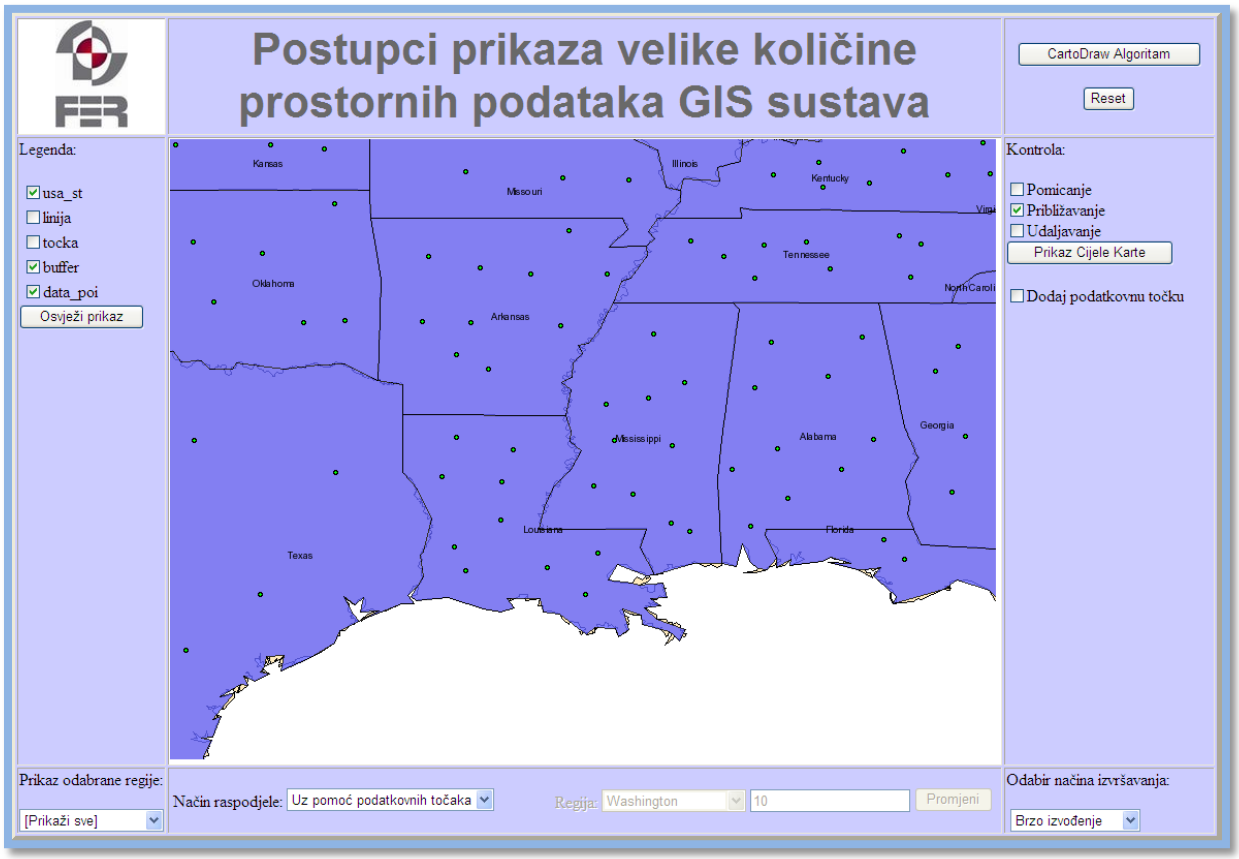
Isto tako, prije pokretanja CartoDraw algoritma, treba pripremiti ulazne podatke preko kojih se želi obaviti raspodjela površina kartograma.

Podatke je moguće pripremiti preko podatkovnih točaka ili ručno – unošenjem vrijednosti za pojedine regije. Pri pokretanju aplikacije neke podatkovne točke su već inicijalno postavljene. Svaka regija sadrži deset podatkovnih točaka čime algoritam (bez dodatnih promjena) teži uniformnoj raspodijeli površina svake regije.

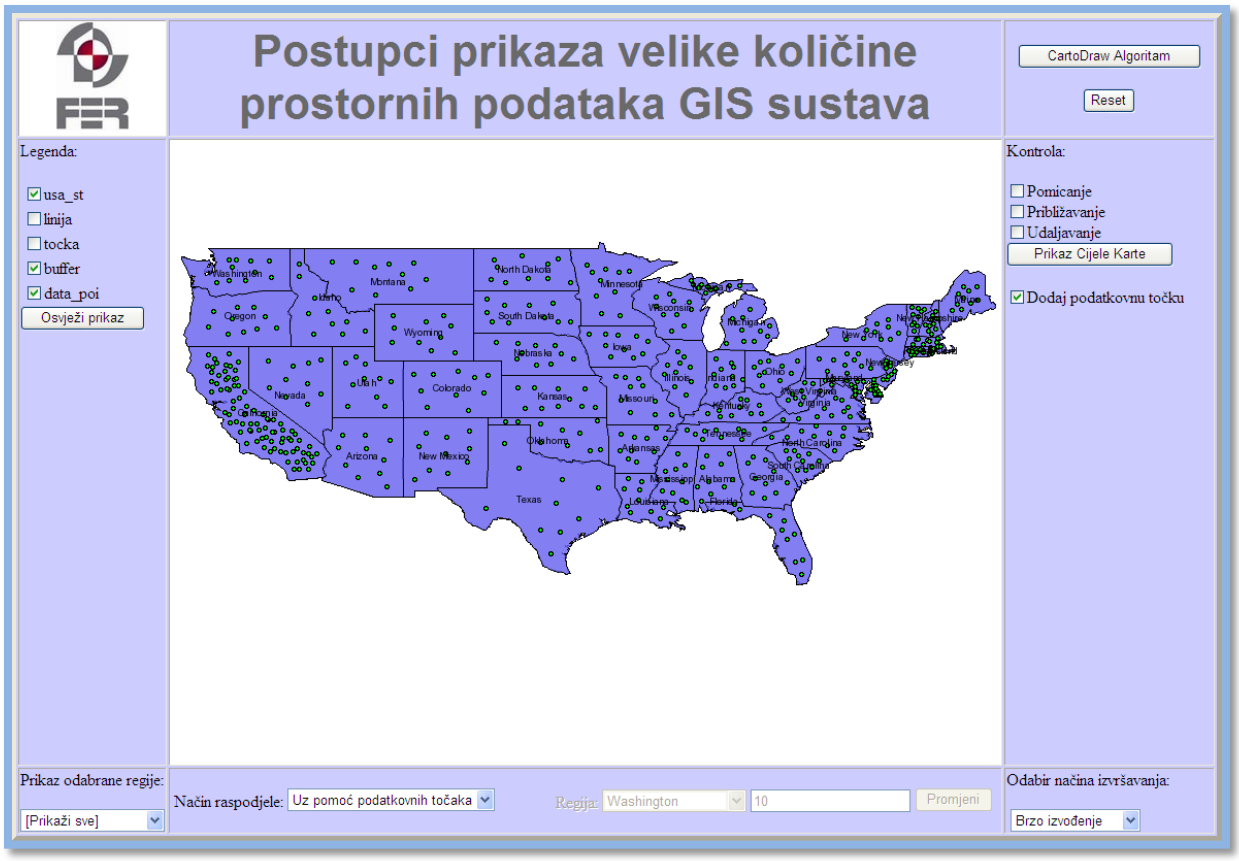
Za potrebe testiranja pripremljene su tri različite verzije ulaznih podataka na različite načine: uniformna razdioba, unošenje podatkovnih točaka (slika 6.12) te ručno – upisivanjem željene razdiobe za pojedino područje (slika 6.13).



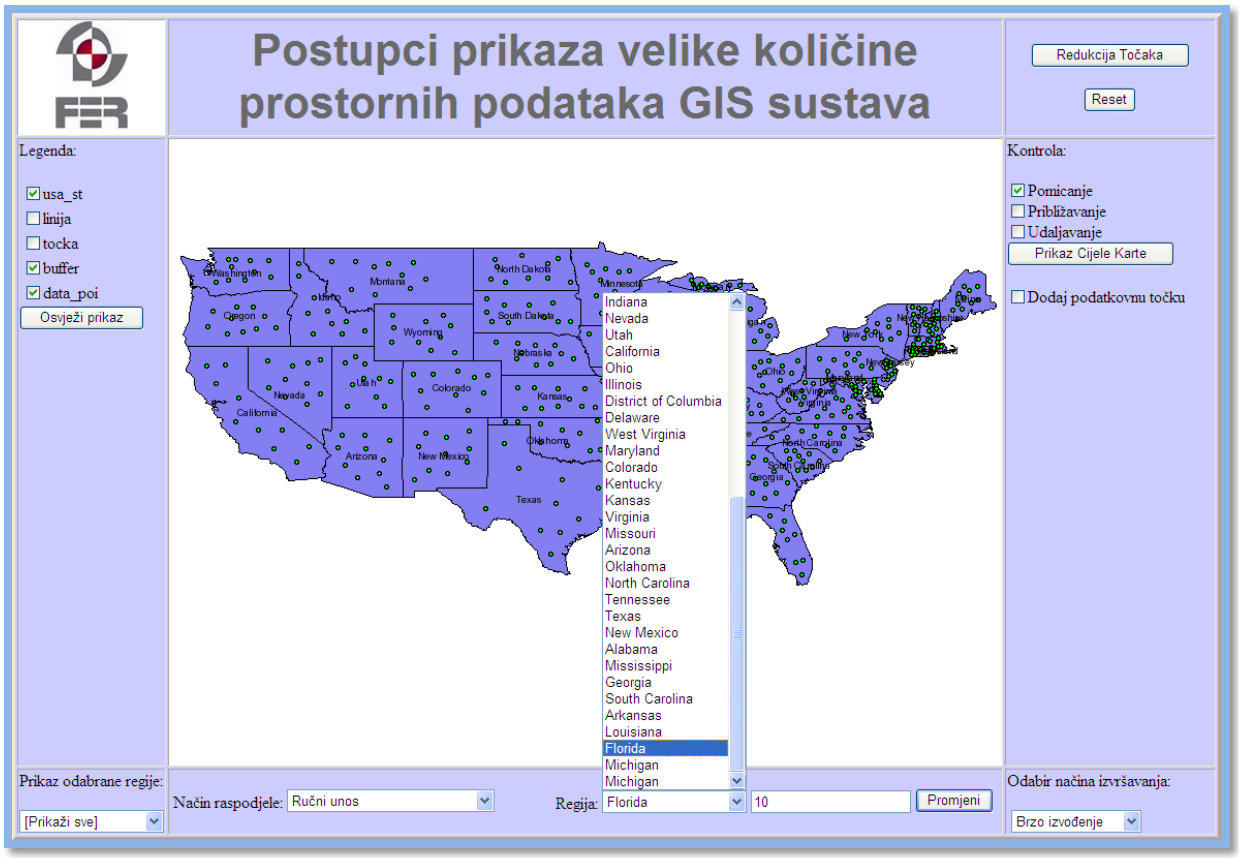
Slika 6.10: Prije izvođenja algoritma za redukciju broja točaka.



Slika 6.11: Nakon izvođenja algoritma za redukciju broja točaka.



Slika 6.12: Dodavanje točaka preko sučelja – dodano 75 je podatkovnih točaka u Kaliforniju.



Slika 6.13: Primjer ručnog namještanja raspodjele.

6.3. Rezultati

Slijede prikazi rezultata iscrtavanja kartograma. Obavljeno je testiranje brzog i kvalitetnog izvođenja algoritma na tri različita primjera. U prvom primjeru su sve dodatne opcije kojima se može pristupiti preko sučelja netaknute tako da površine poligona teže uniformnoj raspodjeli. Za drugi primjer je ubačeno sedamdeset i pet dodatnih podatkovnih točaka u Kaliforniju (kako je prikazano slikom 6.12). U trećem primjeru su pripremljene raspodjele tako da je za Floridu ručno upisan broj sto (kako je prikazano slikom 6.13).

Za sve su primjere korišteni isti parametri. Popis parametara prikazan je tablicom 6.1.

Tablica 6.1: Korišteni parametri u implementaciji.

	$MaxAreaDiff$	r	$const$	ε	ε_S	ε_A
Brzi algoritam	0.013	0.0001	0.01	0.009	2.57	0.0007
Kvalitetni alg.	0.013	0.0001	0.01	0.009	2.7095	0.0007

MaxAreaDiff iz tablice 6.1 se odnosi na parametar pri redukciji broja točaka spomenut u algoritmu 5.1. Vrijednost r je radijus kružnog segmenta spomenut u poglavlju 5.3.2.2.. Parametri ε, ε_S i ε_A su spomenuti glavnom CartoDraw algoritmu (algoritam 5.3).

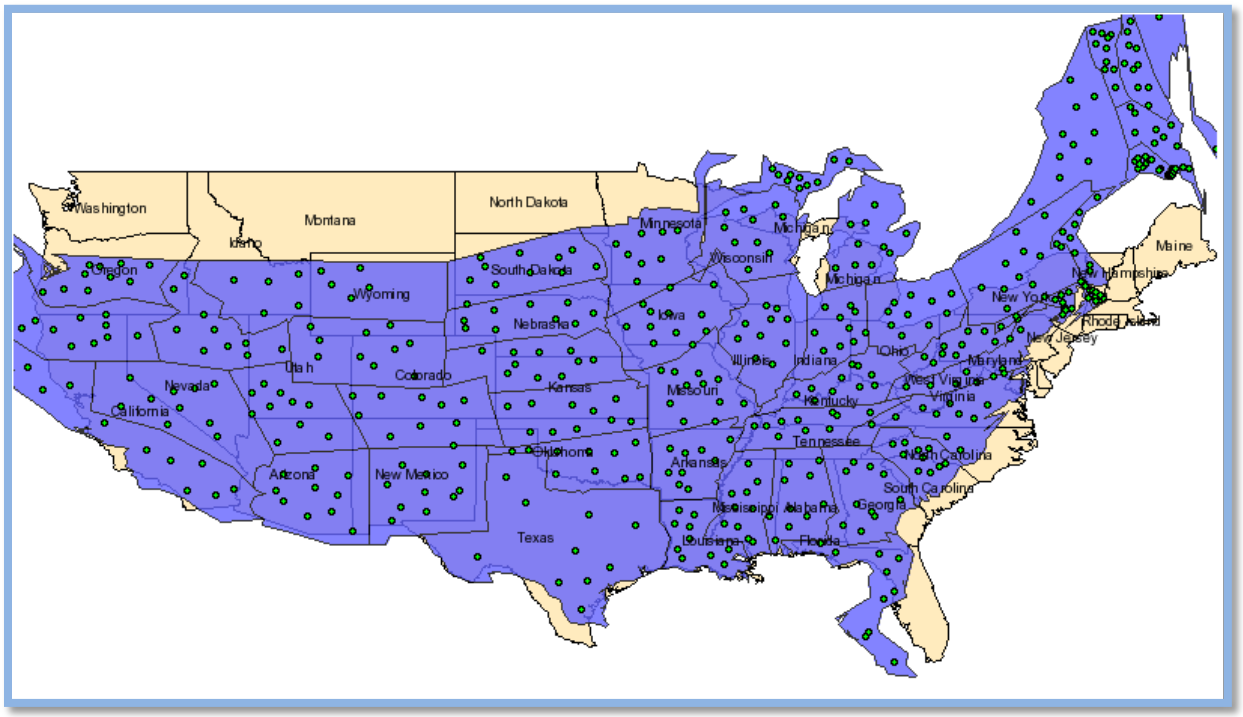
Tablica 6.2 prikazuje vremena potrošena za izvođenje probnih primjera te broj iteracija brzog i kvalitetnog algoritma. Testiranje je izvršeno na računalu: Intel Core Duo CPU T2350 @ 1.86GHz 1.86GHz, 2 GB RAM-a.

Tablica 6.2: Brojevi iteracija i vremena izvođenja probnih primjera.

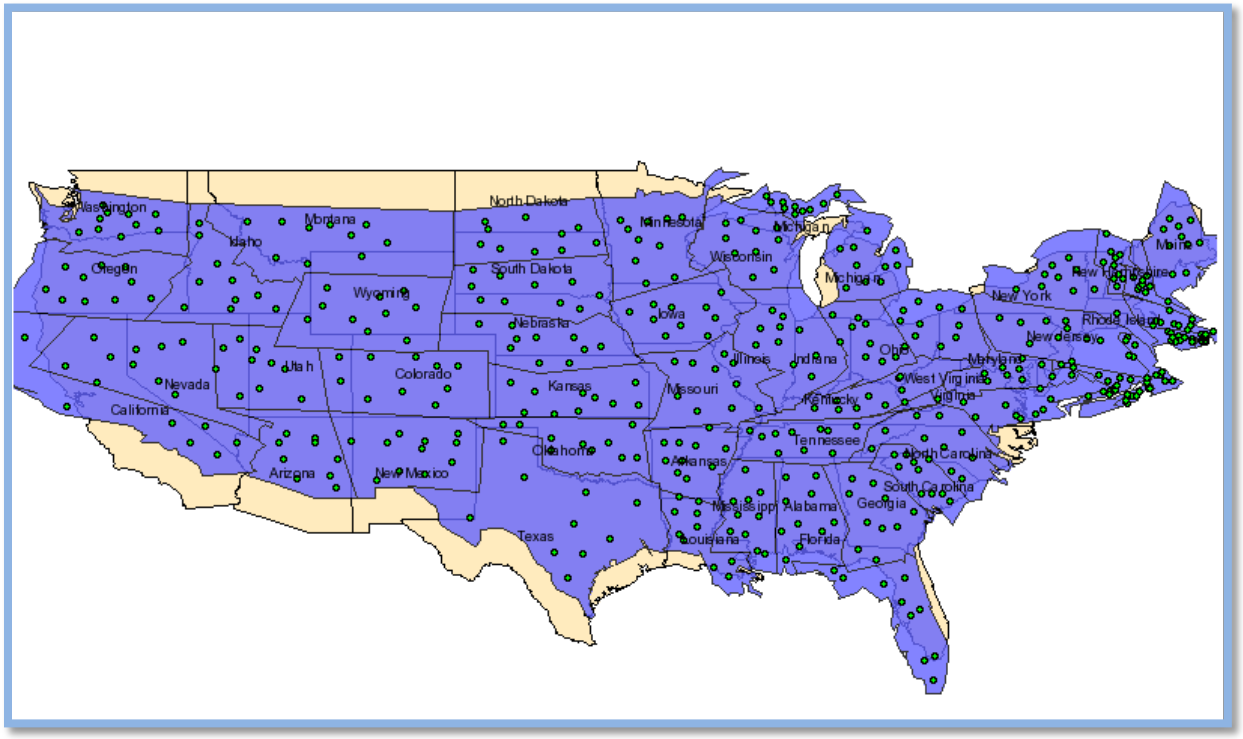
Broj primjera	Način izvođenja	Broj iteracija	Vrijeme izvođenja algoritma [min]	Prosječno vrijeme jednog koraka [min]
Prvi	**Brzi**	199	04:26,2	00:01,3
	Kvalitetni	175	18:23,2	00:06,3
Drugi	**Brzi**	207	04:49,8	00:01,4
	Kvalitetni	213	21:18,0	00:06,0
Treći	**Brzi**	392	07:50,4	00:01,2
	Kvalitetni	356	37:58,4	00:06,4

Slijede slike primjera korištenja aplikacije, njihova ocjena i zapažanja.

6.3.1. Prvi primjer – uniformna raspodjela

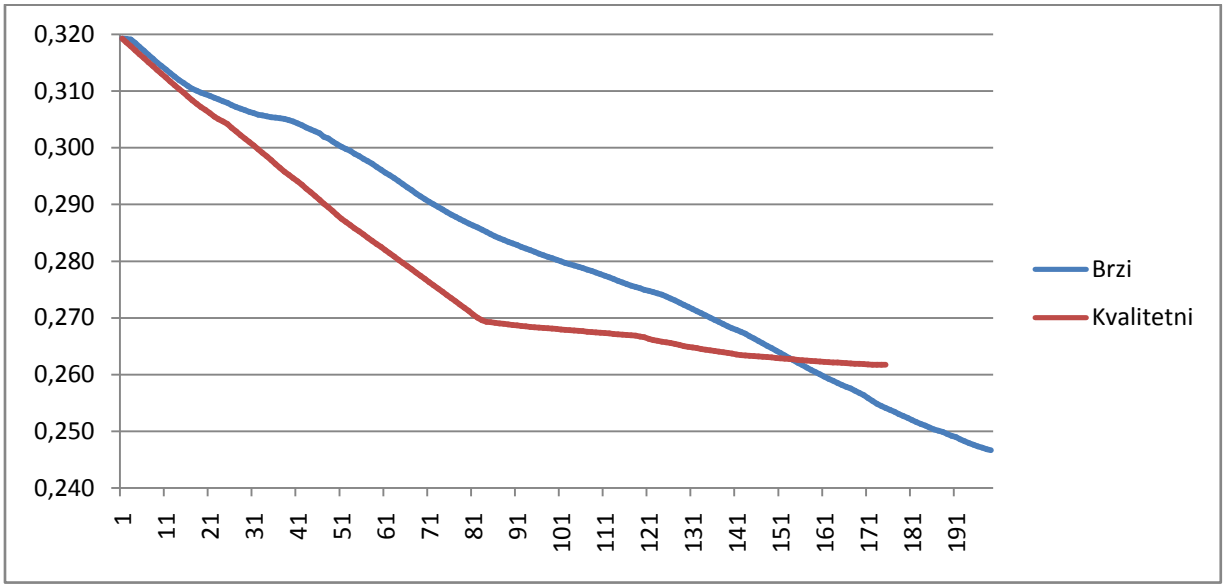


Slika 6.14: Brzi algoritam – uniformna raspodjela.

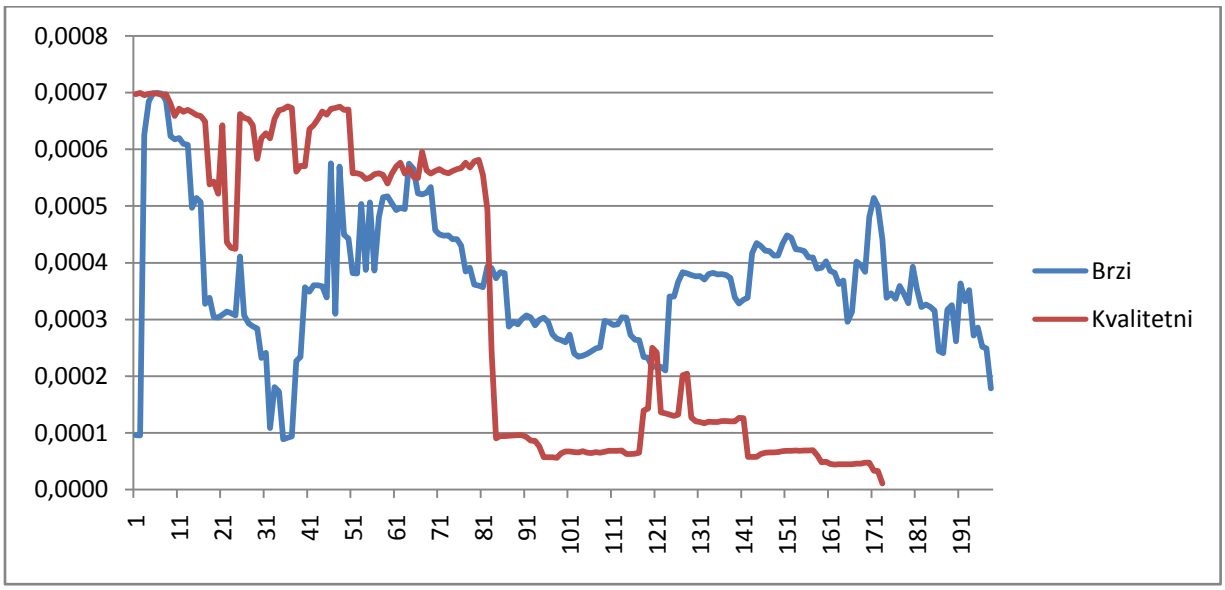


Slika 6.15: Kvalitetni algoritam – uniformna raspodjela.

Istočni dio karte SAD-a ima prostranije države nego što to ima zapadni dio tako da je lako zaključiti kako bi se (radi uniformne razdiobe) lijevi dio karte trebao stisnuti, a desni nabujati. Taj se učinak bolje izražava kod brze verzije algoritma, ali je i distorzija oblika na tom dijelu značajnija. Kako se može vidjeti (slika 6.16) brzi algoritam ne daje puno manju površinsku pogrešku, a znatno izobliči originalnu kartu.

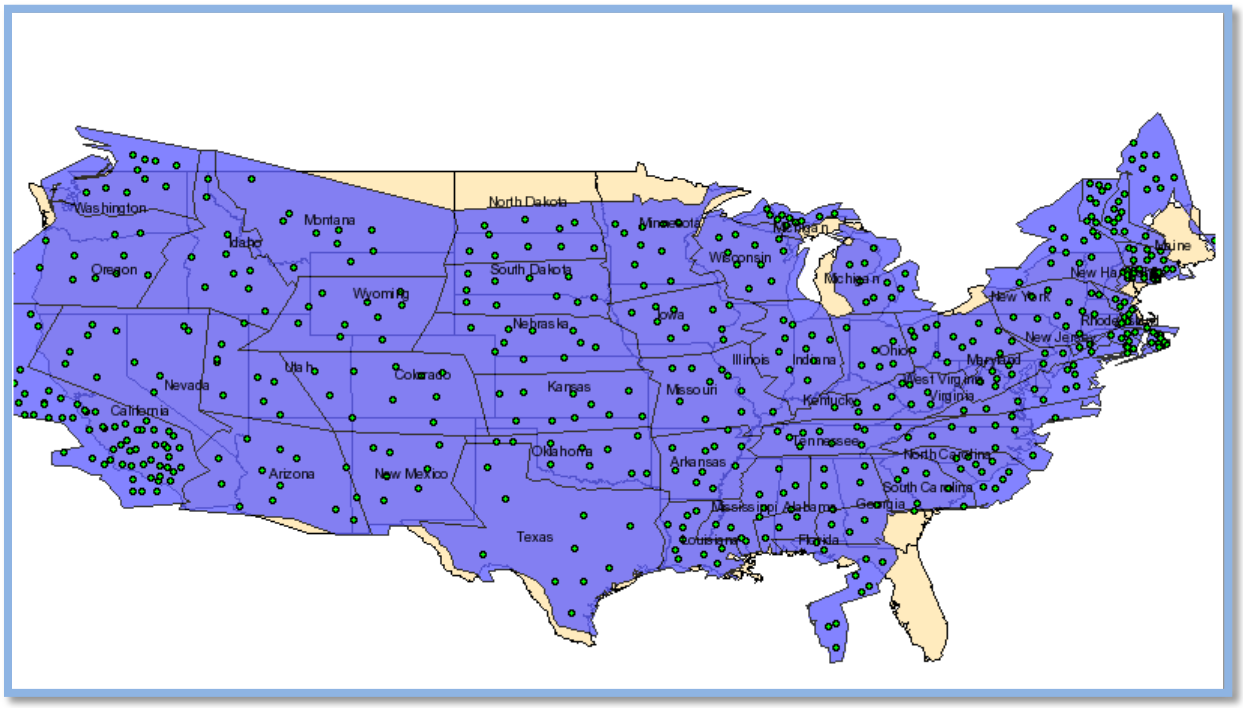


Slika 6.16: Graf površinske pogreške po iteracijama
CartoDraw algoritma za prvi primjer.

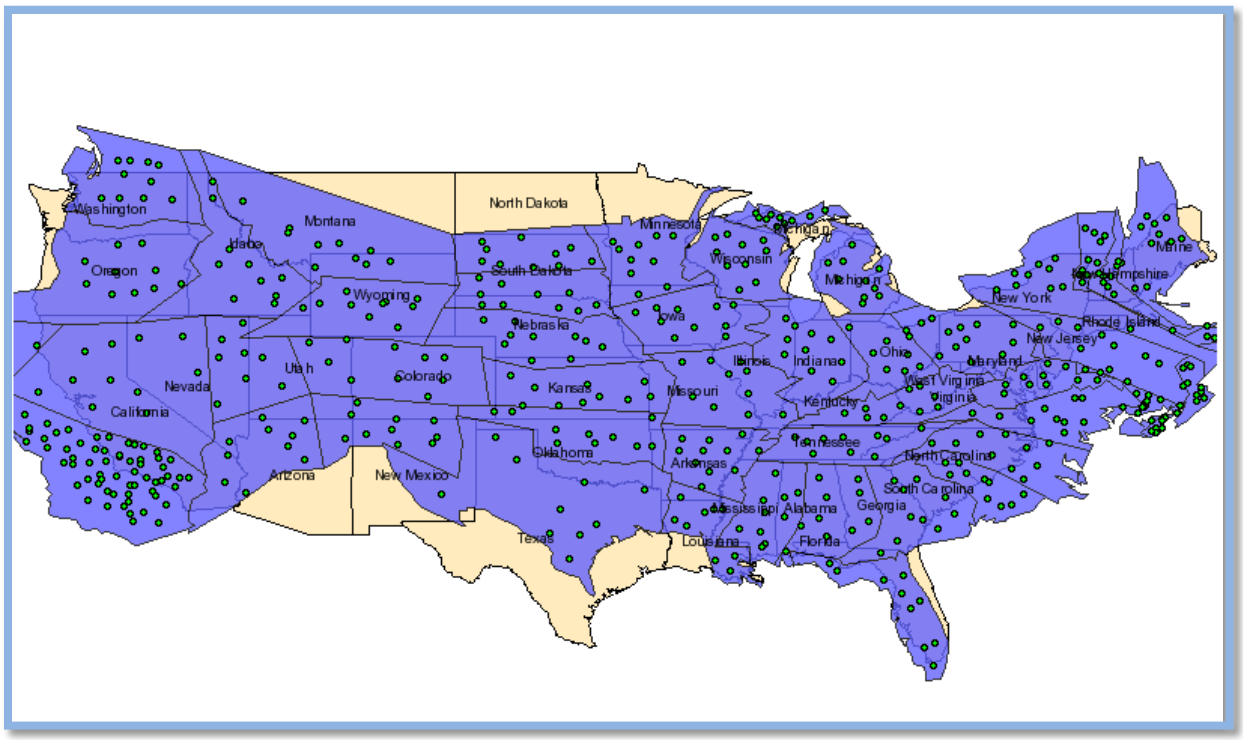


Slika 6.17: Graf poboljšanja nakon svake iteracije
CartoDraw algoritma u odnosu na prethodnu iteraciju za prvi primjer.

6.3.2. Drugi primjer – Kalifornija

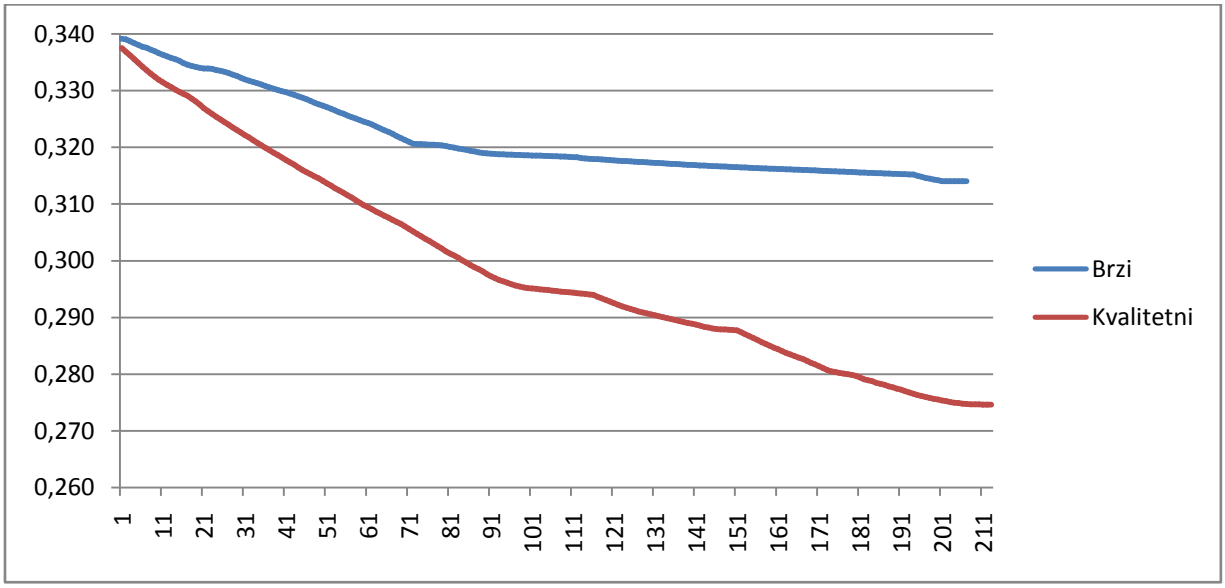


Slika 6.18: Brzi algoritam – Kalifornija sadrži 85 točaka,
a ostale države po 10.

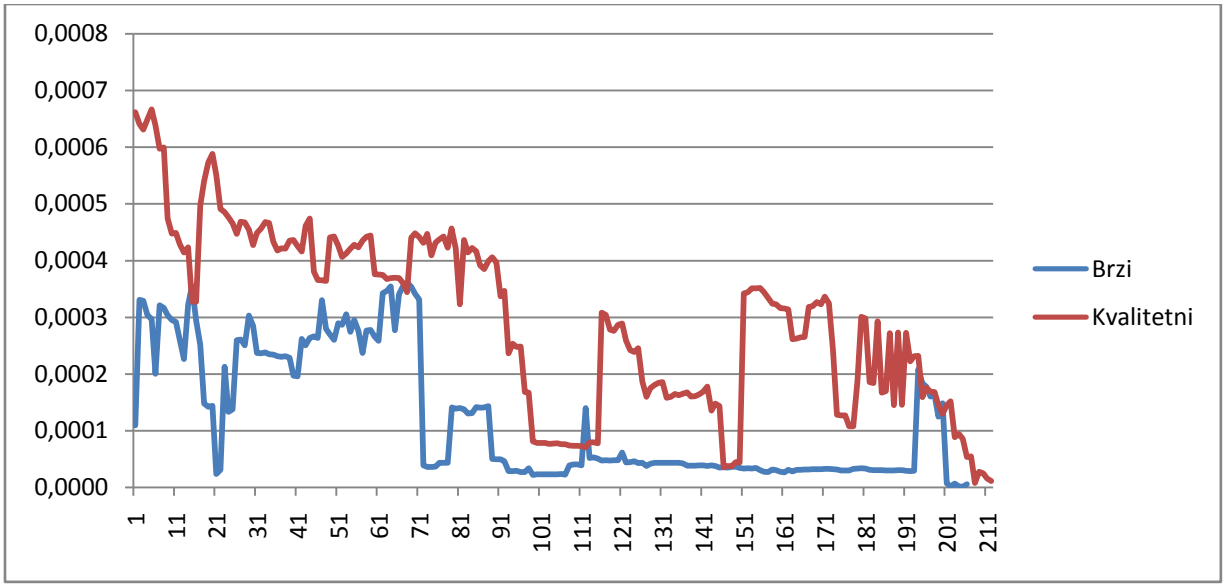


Slika 6.19: Kvalitetni algoritam – Kalifornija sadrži 85 točaka,
a ostale po 10.

U ovom primjeru oba algoritma pokazuju dosta slične rezultate, ali se i ovdje može zamijetiti distorzija na području Floride. Kvalitetni algoritam u ovom primjeru pokazuje znatno bolje rezultate sa strane površinske pogreške (slika 6.20). Zapadni, gušći dio karte se bolje raširio, a istočni (odnosno sredina karte) se znatnije suzio u odnosu na brzi algoritam.

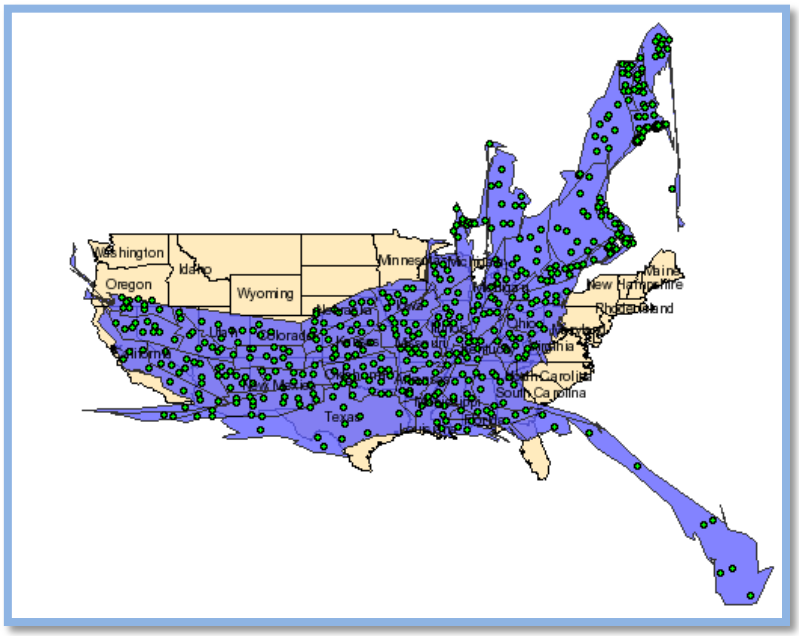


Slika 6.20: Graf površinske pogreške po iteracijama
CartoDraw algoritma za drugi primjer.

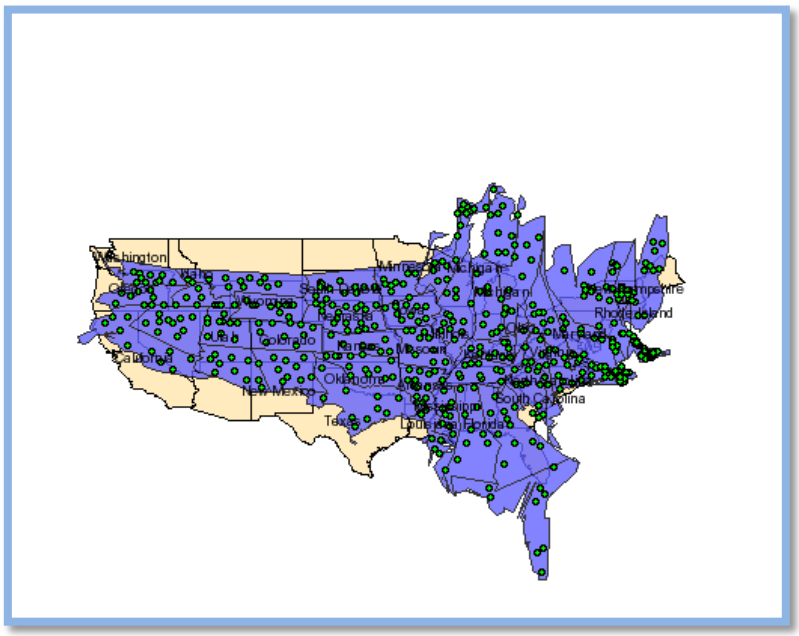


Slika 6.21: Graf poboljšanja nakon svake iteracije
CartoDraw algoritma u odnosu na prethodnu iteraciju za drugi primjer.

6.3.3. Treći primjer – Florida



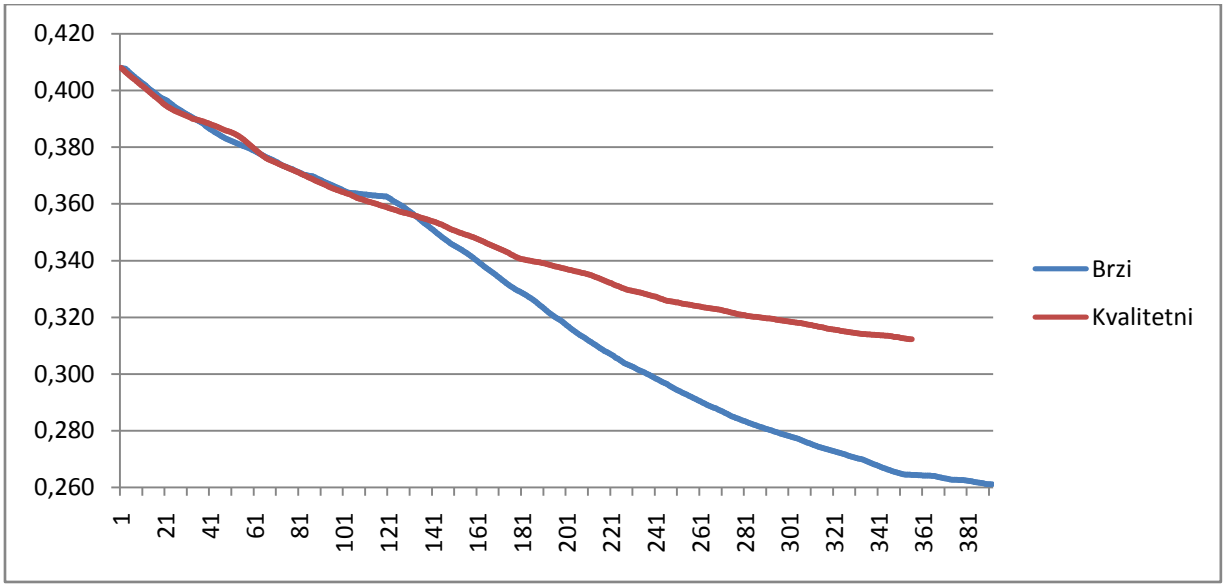
Slika 6.22: Brzi algoritam – Floridi je ručno unesena brojka 100, a ostalim državama po 10.



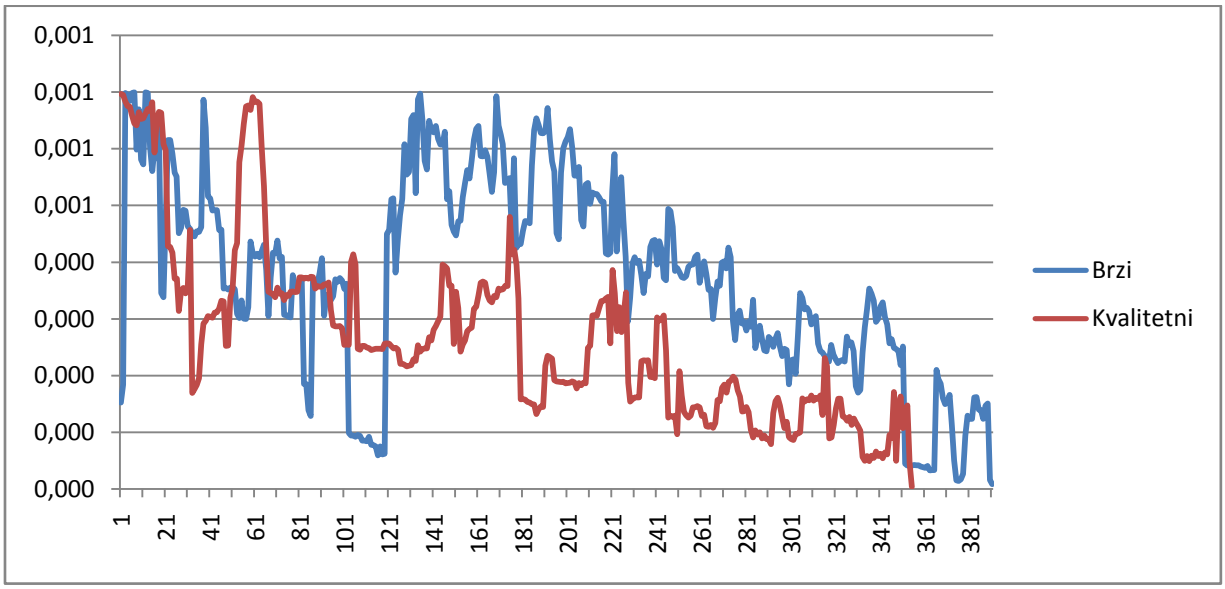
Slika 6.23: Kvalitetni algoritam – Floridi je ručno unesena brojka 100, a ostalim državama po 10.

Na ovom primjeru se može zamijetiti kako se kod brzog algoritma pojavljuje intenzivna distorzija oblika u području Floride i sjeverno od nje. Područje Teksas je isto tako previše raširen te se na jugo-istočnom dijelu pojavio šiljak koji ukazuje na problem provjere – da li je točka u poligonu. Koso širenje Floride u smjeru jugo-zapad znatno povećava granični okvir te regije tako da sve veći broj nepotrebnih točaka djelovanja linije uzorkovanja sp dobije mogućnost rastezati kartu.

Kvalitetni algoritam pokazuje znatno bolje rezultate, iako brzi ima (prema slici 6.24) manju površinsku pogrešku.



Slika 6.24: Graf površinske pogreške po iteracijama
CartoDraw algoritma za treći primjer.



Slika 6.25: Graf poboljšanja nakon svake iteracije
CartoDraw algoritma u odnosu na prethodnu iteraciju za drugi primjer.

7. Zaključak

U ovom radu predstavljena je metoda prikaza velike količine prostornih podataka GIS sustava uz pomoć kartograma. Korišten je brzi CartoDraw algoritam implementiran na dva načina: brzi i kvalitetni algoritam. Kvalitetni algoritam pokazuje bolje rezultate u svim isprobanim primjerima u vidu očuvanosti oblika regija što se pokazuje dosta važnim za lakše snalaženje na karti.

Brzina izračunavanja kartograma za namjenu u SCADA sustavima nije toliko važna jer je dovoljno da se algoritam za izračun obavi jednom nakon unošenja novih podatkovnih točaka te se deformacija karte sačuva kako bi bila uvijek dostupna za korištenja. No glavna namjena korištenog algoritma je kreiranje kartograma na osnovi podataka koji pristižu u stvarnom vremenu pa su u ovom radu, uz kvalitetu algoritma, isprobane i mogućnosti brze generacije kartograma. Prema rezultatima je vidljivo da podaci dobiveni u stvarnom vremenu ne bi smjeli činiti velike razlike u vizualizaciji unutar deset minuta kako bi slika bila konzistentna sa sustavom iz kojeg podaci pristižu.

Može se primijetiti kako sama deformacija mape ne odstranjuje problem međusobnog precrtavanja točaka, već samo smanjuje potrebu da one budu zgusnute. Jedno od mogućih unaprjeđenja je korištenje metoda lokalnog razmještaja točaka kao što je to PixelMaps[3].

8. Literatura

1. Keim, Daniel A.; North, Stephen C.; Panse, Christian: „CartoDraw: A Fast Algorithm for Generating Contiguous Cartograms“, IEEE Transactions on visualization and computer graphics, Siječanj 2004.

2. Panse, Christian; Sips, Mike; Keim, Daniel A.; North, Stephen C.: „Visualization of Geo-spatial Point Sets via Global Shape Transformation and Local Pixel Placement“ IEEE Transactions on visualization and computer graphics, Rujan 2006.

3. Keim Daniel A.; Panse, Christian; Sips, Mike; North Stephen C.: „Pixel Based Visual Mining of Geo-Spatial Data“, Rujan 2003.

4. ESRI Work Group: „ESRI Shapefile Tehnical Description“, Lipanj 1998.

5. Grupa autora: „Scalable Vector Graphics (SVG) 1.1 Specification“, W3C Recommendation 14. Siječanj 2003., http://www.w3.org/TR/SVG11, Listopad 2007.

6. Grupa autora: „Extensible Markup Language“, http://www.w3.org/XML, Listopad 2007.

7. Grupa autora: „MapServer New Users“, http://mapserver.gis.umn.edu/new_users, Listopad 2007.

9. Sažetak

(Postupci prikaza velike količine prostornih podataka GIS sustava)

U ovom radu su kratko opisani SCADA sustavi i potreba za njihovom integracijom s GIS sustavom. Predstavljena je metoda prikaza velike količine prostornih podataka GIS sustava uz pomoć kartograma. Za realizaciju kartograma korišten je brzi CartoDraw algoritam pomoću kojeg je moguće postići podjednaku gustoću podatkovnih točaka unutar područja kako bi se smanjila potreba za međusobnim precrtavanjem točaka. Ostvarene su dvije implementacije algoritma koje se razlikuju u načinu na koji se ispituje da li točka pripada poligonu. Implementacija u kojoj se gleda da li točka pripada graničnom okviru regije daje brži odziv, a implementacija u kojoj se točno određuje pripadnost točke poligonu nudi kvalitetnije rezultate u pogledu očuvanja oblika.

Ključne riječi: SCADA i GIS, CartoDraw, prikaz velike količine prostornih podataka

10. Abstract

(Methods for visualization of large data sets within GIS system)

This study has a short description of SCADA systems and the need for their integration with GIS system. The method for visualization of large data sets in GIS system is presented by using cartograms. CartoDraw algorithm is used for drawing cartograms to achieve nearly equal density of data points within the regions to reduce the need for overplotting. There were created two implementations of algorithm with difference to each other in a way they perform test that determines if point is a part of polygon. Implementation that checks if bounding box of polygon contains the point has quicker response and the one that accurately determines relation between point and polygon offers better shape preservation and therefore more quality result.

Key words: SCADA and GIS, CartoDraw, visualization of large data sets