SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1906

Nefotorealistične tehnike prikaza korištenjem kolaža

Nadja Roić

Zagreb, rujan 2011.

Sadržaj

Popis slika

1 Uvod

Istraživanja u području računalne grafike tradicionalno su se bavila pokušajima da se postigne fotorealizam, tražen je način da se razne fizikalne pojave što vjernije prikažu. U posljednjem desetljeću računalna grafika počinje se sve više baviti i nefotorealističnim prikazom.

Nefotorealistični prikaz je grana računalne grafike koja, za razliku od tradicionalnih tehnika računalne grafike, nije orijentirana na fotorealizam. Nefotorealistični prikaz inspiriran je umjetničkim tehnikama kao što su slikanje, crtanje, tehničke ilustracije te animirani filmovi.

Ovaj rad bavi se tehnikom prikaza kolaža, koji se ubraja u jednu od tehnika nefotorealističnog prikaza. Digitalni kolaž tvori se na način da se slika prikaže pomoću manjih dijelova. Ovi manji dijelovi mogu biti, ovisno o odabranoj tehnici izrade, jednobojni komadići, te fotorealistične ili nefotorealistične sličice, te mogu biti pravilno ili nepravilno posloženi pravokutnici, ali mogu biti i sličice proizvoljnog oblika.

Tradicionalno, kolaž ili mozaik je ornamentalni ukras izrađen od kvadratnih ili višekutnih raznobojnih kamenčića, glaziranih komadića obojene keramike ili komadića obojenog stakla.

U ovom radu naglasak je na tehnici izrade kolaža koju su prezentirali Kim i Pellacini na SIGGRAPH-u 2002 pod nazivom „Jigsaw“ slikovni mozaik. „Jigsaw“ u ovom slučaju dolazi od riječi „Jigsaw puzzle“ što znači slikovna slagalica. Radi se o kolažu u kojemu se slika prikazuje pomoću brojnih malih sličica. Ove sličice su različitih oblika i boja te su posložene tako da što vjernije predstavljaju originalnu sliku. Pri postavljanju sličica treba paziti da se boja originalne slike i boja sličice što više podudaraju, te da se međusobno preklapanje sličica ili razmak među njima svede na minimum.

U poglavlju 2 bit će ukratko opisane različite tehnike izrade nefotorealističnih prikaza, te kolaža: „Jigsaw“ slikovni mozaik, „Puzzle“ slikovni mozaik, foto-mozaik, te simulirani dekorativni mozaik. U poglavlju 3 detaljnije se obrađuje Voronojev dijagram te njegova primjena za podjelu na površine objekta. Algoritmi za izradu Voronojevog

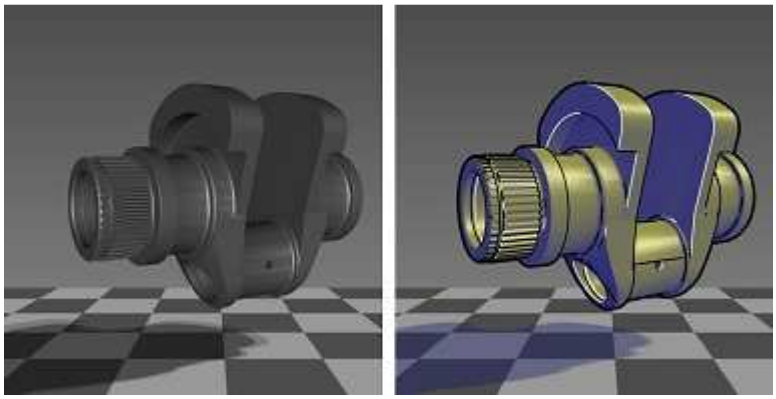
dijagrama dani su u poglavlju 4, a poglavlje 5 govori o algoritmima za izradu centroidalnog Voronojevog dijagrama. Izrada samog centroidalnog Voronojevog dijagrama može se smatrati najsloženijim korakom u izradi „Jigsaw“ kolaža, te se zbog toga njemu posvećuje posebna pažnja u ovom radu. Poglavlje 6 govori o programskom ostvarenju samog „Jigsaw“ kolaža. Rezultati su prikazani u poglavlju 7.

2 Postupci izrade nefotorealističnih prikaza

Kada je u pitanju postizanje raspoznavanja objekta na slici, slika lišena sjena i refleksija može biti puno uspješnija od najnaprednijeg fotorealističnog prikaza. Uklanjanjem suvišnih detalja možemo sliku optimizirati kako bi se objekt bolje raspoznao. U nastavku će biti opisane neke od tehnika izrade nefotorealističnih prikaza.

2.1 Sjenčanje

Fotorealistično sjenčanje donosi puno komplikacija pri uporabi u tehničkim ilustracijama. Ono ponekad adekvatno ne ocrtava osvijetljene objekte dok sjenčanje može zamračiti detalje na površini. Gooch i suradnici razvili su tehniku osvjetljavanja modela simulirajući hladno-do-toplo sjenčanje za tehničke ilustracije.



Slika 2.1. Sjenčanje bazirano na toplim i hladnim bojama

2.2 Prikaz po uzoru na animirani film

Ovaj stil karakteriziraju spojene, uniformno obojane površine. Sastoji se od iscrtavanja linija i bojenja. Jednostavniji prikazi koristit će čvrste boje za različite objekte, dok će kompleksniji prikazi koristiti dvije ili tri boje za svaki materijal. Ovaj pristup često se naziva stepeničastim sjenčanjem. Stepeničasto sjenčanje razlikuje se od fotorealističnih pristupa zbog nedostatka mekanog prijelaza između osjenčanih djelova objekta.

2.3 Siluete i rubovi

Iscrtavanje linija jest jedan od najčešćih i najefektivnijih stilova ilustracije. Primjena ovog stila nalazi se u tehničkim ilustracijama, arhitektonskim nacrtima, znanstvenim dijagramima i raznim umjetničkim tehnikama. Tehnike iscrtavanja linija mogu predočiti informaciju preko slike na sažet i apstraktan način.

Siluetama možemo sa samo nekoliko poteza iscrtati objekte i predočiti činjenice kao što su veličina i oblik objekta. Siluete se mogu izraditi na različite načine kako bi utjecale na percepciju ili estetsku vrijednost oponašajući umjetnost. Iscrtavanje objekata isto se tako može integrirati sa fotorealističnim slikama kako bi se poboljšala razumljivost. Primjer siluete prikazan je na slici 2.2.

Slika 2.2. Silueta

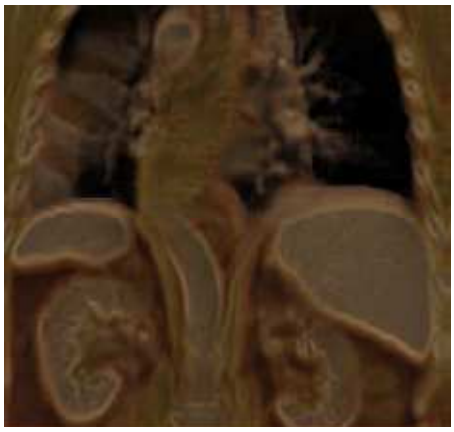
2.4 Volumne ilustracije

Prikazi volumena uobičajeni su u područjima medicinske i znanstvene vizualizacije. Mogućnost da se vizualiziraju kompleksni fenomeni iz stvarnog svijeta našli su svoju primjenu u CT i MRI prikazima mozga i prikazima tokova u dinamici tekućine.

Fotorealističan pristup kreiranju volumena kreira slike koje odgovaraju izgledu materijala u prirodi, ali mogu izostaviti važne strukturalne detalje. Volumne ilustracije kombiniraju fotorealističan pristup s mogućnošću naglašavanja bitnih značajki koristeći nefotorealistične tehnike prikaza. Primjena tehnike volumnih ilustracija zahtjeva manje ručnog podešavanja. Volumne ilustracije pružaju fleksibilan okvir prikaza percepcije volumnih modela kroz naglašavanje značajki i dodavanje efekata osvjetljenja.

Tradicionalno, za prikaz volumena koristi se jedna od dvije tehnike. Prva pokušava načiniti fizikalno točnu simulaciju procesa kao što je osvjetljenje i slabljenje svjetla u plinovitom volumenu ili slabljenje X-zraka kroz tkivo. Ovaj pristup stvara najrealniji pogled na volumni skup podataka, barem što se tiče podataka koji imaju odgovarajuće fizikalno značenje. Drugi pristup je samo djelomično baziran na fizikalnom ponašanju prolaska svjetlosti kroz volumen te koristi proizvoljne prijenosne funkcije koje specificiraju izgled uzorka volumena. Ovaj pristup omogućava dizajneru kreiranje širokog opsega izgleda volumena u vizualizaciji, ali žrtvuje jednostavnost interpretacije fizikalno baziranih pristupa.

Pristup koji kombinira fizikalno bazirani pristup s nefotorealističnim tehnikama inspiriran je umjetničkim tehnikama i tehničkim ilustracijama. Volumna ilustracija spaja prednosti dviju tradicionalnih metoda prikaza volumena na fleksibilan i parametriziran način, omogućuje lakoću interpretacije koja je rezultat fizikalno baziranog osvjetljenja, te omogućuje fleksibilnost metode prijenosnih funkcija. Tehnike volumne ilustracije mogu se upotrijebiti kako bi se kreirale vizualizacije volumenskih podataka koje su efektivnije od tradicionalnih pristupa u prijenosu struktura unutar volumena. Volumne ilustracije primarno su namijenjene prikazu tijela u udžbenicima, znanstvenim člancima, te ostalim obrazovnim materijalima.



Slika 2.3. Primjer volumne ilustracije na slici dobivenoj medicinskim CT-om

2.5 Tehnike mozaika

Mozaici su slike napravljene lijepljenjem malih obojenih pločica na površinu. Vjerojatno su prvi primjer tehnike sinteze slike od diskretnih primitiva. Kreiranje digitalnih mozaika predstavlja jedno od najnovijih istraživanja u području

nefotorealističnog prikaza. Digitalni mozaici su ilustracija sastavljene od skupa malih slika koje se nazivaju pločice. Pločice tvore izvornu sliku u pokušaju imitacije klasičnog mozaika.

Krenuvši od iste izvorne slike moguće je kreirati različite vrste digitalnih mozaika. Krajnji izgled mozaika ovisi o skupu pločica koje možemo koristiti, te o pravilima koja govore na koje pozicije možemo stavljati pločice, te o mogućim deformacijama i rotacijama pločica.

Prvi korak u rješavanju problema kreiranja digitalnog mozaika je formunlacija problema u matematički okvir. Konkretnije, kreiranje mozaika moguće je prikazati kao problem optimizacije:

Za zadano pravokutno područje I^2 u ravnini R^2, skup pločica i skup ograničenja, treba naći N pozicija $P_i(x_i,y_i)$ u I^2 te na ta mjesta postaviti N pločica, po jednu na svaku poziciju P_i na način da su sve pločice disjunktne, površina koju pokrivaju je maksimizirana, a ograničenja se poštuju "koliko god je moguće".

Simulirani dekorativni mozaik

Postoje dvije osnovne vrste klasičnih tehnika izrade mozaika s obzirom na način postavljanja pločica. To su „opus musivum“ i „opus vreticulum“ što bi se slobodno moglo prevesti kao „rad muza“ i „rad u obliku crva“. Ove dvije tehnike prikazane su na slikama 2.4. i 2.5.

Slika 2.4. Antički mozaik „opus musivum“

Slika 2.5. Antički mozaik „opus vremiculatum“

„Opus musivum“ karakterističan je po tome što sve pločice slijede osnovnu liniju, dok se kod „opus vermiculatum“ pojavljuje samo rub oko zadanog oblika.

Obje vrste mozaika poštuju sljedeća pravila:

1. svaka pločica je uniformno obojena
2. pločice se mogu razlikovati u veličini i obliku unutar razumnog opsega te su pretežno konveksnog oblika
3. praznine između pločica se minimiziraju, ali i predstavljaju grafički element koji naglašava rubove i linije

U izradi algoritma za ovu vrstu mozaika ova svojstva se pokušavaju poštovati, s nešto manjim naglaskom na konveksnost pločica.

Pločice moraju biti postavljene u skladu s osnovnim linijama te se nastoji izbjeći njihovo međusobno preklapanje.

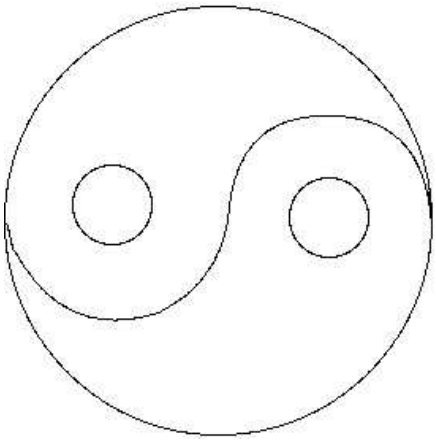
Di Blasi predlaže sljedeći algoritam za izradu ove vrste mozaika: Prvi korak je pronalaženje osnovnih linija slike kao što je prikazano na slici 2.7. Nakon što su osnovne linije detektirane, za svaki slikovni element (piksel) slike traži se minimalna udaljenost od slikovnih elemenata linije te se na temelju toga tvori matrica prikazana na slici 2.8. Iz ove matrice kreiraju se još dvije matrice potrebne da bi se napravio finalni mozaik: gradientna matrica *gM* prikazana na slici 2.9. i matrica poravnanja *llM* prikazana na slici 2.10. Ove matrice računaju se na način:

$$gM(x,y) = \arctan\frac{dtM(x,y+1) - dtM(x,y-1)}{dtM(x+1,y) - dtM(x-1,y)} \tag{1}$$

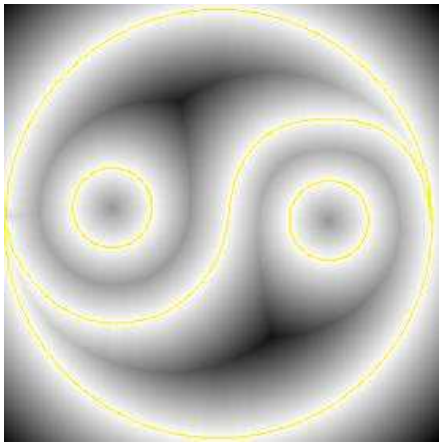
$$llM(x,y) = \begin{cases} 1\ ako\ je\ modulo(dtM(x,y), 2\cdot tSize) = 0 \\ 2\ ako\ je\ modulo(dtM(x,y), 2\cdot tSize) = tSize \\ 0\ inače \end{cases} \tag{2}$$

pri čemu je *tSize* cijeli broj koji označava veličinu pločice.

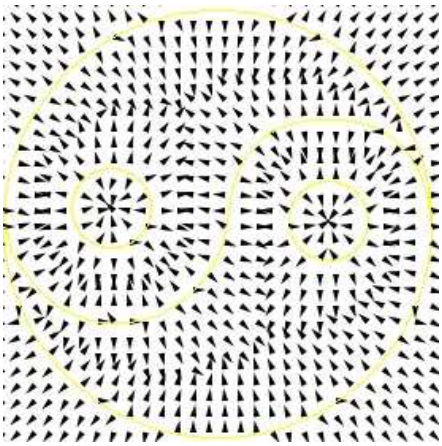
Slika 2.6. Originalna slika



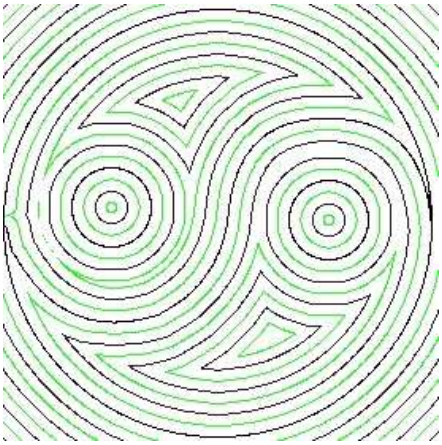
Slika 2.7. Osnovne linije



Slika 2.8. Matrica minimalne udaljenosti



Slika 2.9. Gradijentna matrica



Slika 2.10. Matrica poravnanja

Može se primijetiti sljedeće:

1. nije potrebno da su pločice kvadratne, ali je potrebna dimenzija *tSize* kako bi se izračunala *llM* matrica
2. funkcija korištena za izračun *llM* matrice može se s lakoćom adaptirati kako bi se slika pripremila za prihvat pločica varijabilnih veličina
3. kako je *tSize* cijeli broj i matrica *dtM* ima samo cjelobrojne ulaze, modulo je uvijek dobro definiran

Slijedeći korak je postavljanje pločica na njihova mjesta koristeći slikovne elemente iz *llM* matrice sa vrijednošću 2. Ti slikovni elementi tvore sljedove u obliku lanca.

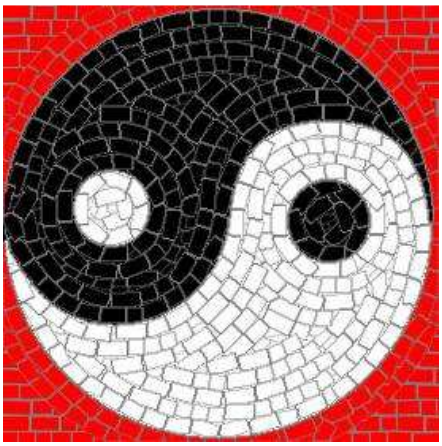
Algoritam glasi

sve dok postoje slikovni elementi

1. odaberi lanac
2. počevši sa slučajnim slikovnim elementom slijedi lanac
3. postavi novu pločicu na propisanu udaljenost putem, orijentacija pločica je određena gradijentnom matricom *gM*

Ukoliko se pločica postavlja samo na temelju do sada prikazane metode javljaju se dva problema, pločice se mogu preklapati ili pločica može prekriti liniju. Preklapanje pločica detektira se pamćenjem prekrivenih slikovnih elemenata. Ukoliko pločica pređe preko linije, reže se.

Nakon ovih koraka potrebno je još provesti akcije koje će poboljšati krajnji estetski dojam pa se svaka pločica smanji kako bi se vidio prostor između pločica. Slika 2.11. prikazuje krajnji rezultat izrade mozaika ovom tehnikom.



Slika 2.11. Simulirani dekorativni mozaik

Fotomozaik

Fotomozaik je tehnika koja transformira sliku u pravokutnu mrežu nalijepljenih sličica. Tipični algoritam za izradu fotomozaika pretražuje veliku bazu sličica kako bi našao onu koja se najviše poklapa s blokom koji treba prekriti. S obzirom da kvaliteta izlazne slike ovisi o veličini baze sličica za prekrivanje, usko grlo svakog algoritma za izradu fotomozaika je proces traženja najbolje sličice. Neki algoritmi predviđaju podatkovne strukture konstruirane „off-line“ kako bi se proces traženja najbolje sličice ubrzao.

Kako bi se fotomozaik izradio ulazna slika prvo se podijeli u mrežu pravokutnika, te se nakon toga svaka ćelija mreže podijeli u podmrežu sa 3x3 ćelija. Za svaku

podćeliju izračuna se srednja vrijednost RGB komponenti. Na ovaj način dobivamo vektor *x* sastavljen od 27 komponenti. Na temelju ovog vektora traži se sličica koja najbolje odgovara ćeliji mreže. Slika 2.12. prikazuje rezultat ove tehnike.

Slika 2.12. Fotomozaik

„Jigsaw“ slikovni mozaik

„Jigsaw“ slikovni mozaik je vrsta mozaika u kojemu se slikovne pločice proizvoljnog oblika koriste kako bi se sastavila slika. Ovu vrstu mozaika prezentirali su Kim i Pellacini na SIGGRAPH-u 2002. Generiranje „Jigsaw“ slikovnog mozaika može se svesti na sljedeći problem:

Za zadani spremnik proizvoljnog oblika i skup slikovnih pločica proizvoljnog oblika, treba popuniti spremnik što je moguće kompaktnije sa pločicama iz skupa u bojama sličnim spremniku pri čemu je moguća blaga deformacija oblika pločica kako bi se postigao bolji vizualni efekt.

Problemu se pristupa tako da se mozaik definira kao konfiguracija pločica koja minimizira funkciju energije mozaika. Mijenjanjem težinskih faktora u funkciji energije mogu se kreirati različiti tipovi mozaika. Kažemo da je konfiguracija pločica „Jigsaw“ slikovni mozaik kada minimizira energiju *E* definiranu kao:

$$E = w_C \cdot E_C + w_G \cdot E_G + w_O \cdot E_O + w_D \cdot E_D \tag{3}$$

E_C je energija boje koja kažnjava konfiguracije koje se ne poklapaju sa bojom ulazne slike. E_G je energija praznine koja kažnjava konfiguracije koje imaju previše praznina između pločica. E_O je energija preklapanja koja kažnjava međusobno preklapanje pločica. E_D je energija deformacije koja kažnjava konfiguracije s izuzetno deformiranim pločicama. Male deformacije se dozvoljavaju s obzirom da u mnogo

slučajeva nije moguće pronaći konfiguraciju u kojoj su preklapanja i praznine dovoljno male da bi se postigao odgovarajući vizualni efekt. Ovo se prije svega događa u slučajevima s malim bazama pločica.

Ovim algoritmom možemo sastaviti ranije spomenuti fotomozaik tako da ograničimo skup ulaznih pločica na pravokutne pločice, te postavimo težinske faktore preklapanja, praznina i deformacija na beskonačno. Da bi se ovim algoritmom izradio ranije spomenuti simulirani dekorativni mozaik potrebno je ograničiti ulazni skup pločica na jednobojne pločice, pri čemu su boje pločica odabrane iz palete ulazne slike. Težinski faktori preklapanja i deformacije postavljaju se na veliki iznos. Težinski faktor praznine umjereno je velik.

Algoritam za izradu „Jigsaw“ slikovnog mozaika podijeljen je u tri faze. U prvoj fazi izabire se mjesto na koje ćemo postaviti pločicu, pri čemu se ignoriraju deformacije. U drugoj fazi, vrše se male popravke pozicija na koje će se postaviti pločice. U trećoj, završnoj fazi svaka pločica se postavlja na svoju poziciju.

Prva faza algoritma pronalazi približno dobru konfiguraciju ignorirajući deformacije, što znači da se traži konfiguracija koja minimizira energiju boje, preklapanja i praznine.

$$E = w_C \cdot E_C + w_G \cdot E_G + w_O \cdot E_O \tag{4}$$

Da bi se pronašla ova konfiguracija koristi se algoritam pretraživanja najboljim prvim. Algoritam postavlja jednu po jednu pločicu. Za svaku novu pločicu traži se okvirno prikladna pozicija u spremniku. Nakon toga pretražuje se baza kako bi se odredilo koju pločicu koristiti, te se određuje točna pozicija i orijentacija pločice. Nakon što se postavila pločica računa se novi spremnik koji se sastoji od starog spremnika iz kojeg se oduzela površina na koju je postavljena pločica u prethodnom koraku. Sa postavljanjem pločica se nastavlja sve dok se ne popuni cijeli spremnik ili se ne može pronaći odgovarajuća pločica za popunjavanje spremnika. Ukoliko se ovo dogodi, konfiguracija se vraća za nekoliko koraka do konfiguracije s minimalnom energijom.

Nakon pronalaženja najboljeg razmještaja još uvijek može postojati previše preklapanja i praznina koji estetski kvare sliku. Ovo se pretežno događa u slučajevima s malim skupom pločica. Ponekad se estetski bolji rezultati mogu postići

blagom deformacijom pločica kako bi se reducirala pojava preklapanja i praznina, no deformacije ne smiju biti prevelike kako ne bi previše promijenile originalnu pločicu.

S obzirom da bi jednostavna implementacija gore predloženog algoritma zahtijevala prevelike računalne resurse, predloženo je nekoliko tehnika optimizacije.

Prilikom postavljanja pločice u spremnik isprobavanje svake moguće lokacije bilo bi prezahtjevno. Zbog toga se isprobavaju samo one lokacije koje će spremnik učiniti lakšim za popunjavanje u sljedećoj iteraciji. S obzirom da ovo ovisi i o konkretnoj pločici koja će biti zalijepljena ne možemo unaprijed odrediti kakav spremnik ćemo dobiti nakon lijepljenja pločice, no možemo predvidjeti kako će spremnik izgledati nakon lijepljenja pločice prosječnog oblika. Što je spremnik konveksniji to će ga biti lakše popuniti. Prije postavljanja nove pločice konstruira se centroidalni Voronojev dijagram u kojem svaka površina ima veličinu površinu otprilike jednaku veličini prosječne pločice. Nakon što se načini centroidalni Voronojev dijagram izabire se područje s najmanjim brojem susjeda kako bi se u sljedećoj iteraciji spremnik mogao lakše popuniti.

Svaki puta kada ne možemo pronaći odgovarajuću pločicu za ispunu spremnika, vraćamo se na konfiguraciju koja je do tog trenutka imala najmanju energiju. Kako bi se ovo izbjeglo pokušava se gledati unaprijed na način da se prilikom postavljanja nove pločice kažnjavaju one pločice koje će otežati punjenje spremnika u sljedećoj iteraciji. Kako bi se to postiglo funkcija energije uzima u obzir kako će spremnik izgledati nakon postavljanja pločice.

$$E = w_C \cdot E_C + w_G \cdot E_G + w_O \cdot E_O + w_{LA} \cdot E_{LA} \tag{5}$$

$$E_{LA} = w_A \cdot površina + (1 - w_A) \cdot duljina^2 \tag{6}$$

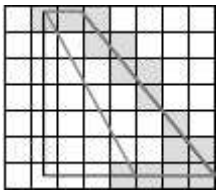
pri čemu je *površina* površina spremnika, *duljina* je duljina granice, a w_A težinski faktor.

Dodavanje oblika spremnika u funkciju energije sprečava postavljanje pločica koje će u sljedećoj iteraciji pridonijeti težem ispunjavanju spremnika.

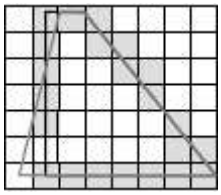
Za zadani spremnik i lokaciju unutar spremnika trebali bismo isprobati sve moguće pločice iz skupa, a isto tako i njihove pozicije i orijentaciju. Kako je broj pločica uvijek visok, linearno pretraživanje je zabranjeno. Koristi se tehnika geometrijskog hash-

iranja da bi se odabralo nekoliko pločica koje odgovaraju određenoj poziciji u spremniku. Nakon toga za te pločice evoluira se funkcija energije i odabire najbolja.

Kako bi se koristila tehnika geometrijskog hash-iranja, kreira se pravokutna mreža u fazi pretprocesiranja. Svaki kvadrat ove mreže predstavlja ulaz u hash tablici. Ukoliko granica oblika prolazi kvadratom mreže, identifikacijski broj pločice i njena orijentacija zabilježit će se u hash tablici. U fazi pretprocesiranja za sve pločice i njihove orjentacije stvaraju se zapisi u hash tablici. Svaki put kada moramo postaviti pločicu na određenu poziciju unutar spremnika, tu poziciju uspoređujemo sa zapisima u hash tablici. Pločice čiji se zapisi iz hash tablice najbolje poklapaju sa rubovima pozicija iz spremnika smatraju se potencijalnim kandidatima. Na slici 2.13. vidimo primjer pločice sa lošim preklapanjem, a na slici 2.14. primjer pločice sa dobrim preklapanjem.



Slika 2.13. Loše preklapanje – preklapanje sa 15 bodova



Slika 2.14. Dobro preklapanje – preklapanje sa 22 boda

Tehnika hash-iranja smanjuje kompleksnost algoritma sa $O(N_{pločica})$ na $O(h)$ pri čemu je h granulacija mreže.

Na slikama 2.15. i 2.16 dani su rezultati ovog algoritma koje su izradili Kim i Pellacini.

Slika 2.15. Mozaik sastavljen od 1367 pločica

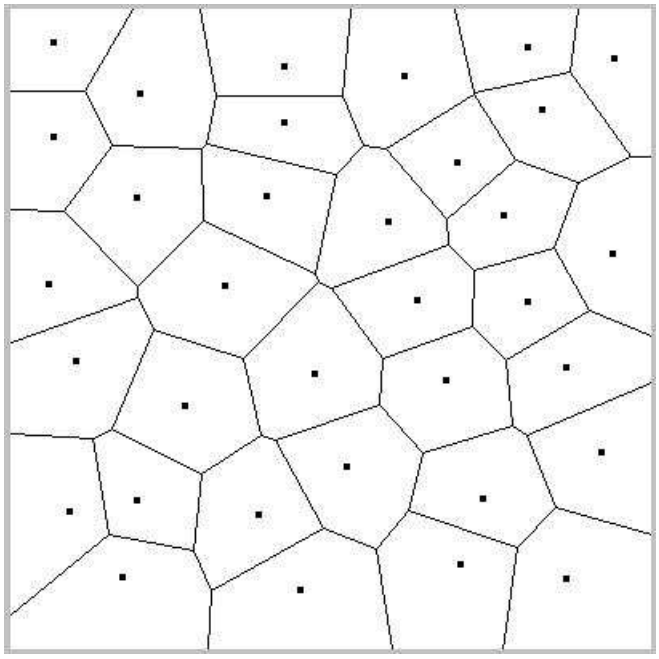
Slika 2.16. Mozaik sastavljen od 1812 pločica

3 Voronojev dijagram

Voronojev dijagram je posebna vrsta dekompozicije na diskretne skupove objekata u prostoru, npr. diskretne skupove točaka metričkog prostora određena udaljenošću. Ime je dobio po Georgy-ju Voronojevu. Za Voronojev dijagram upotrebljavaju se i nazivi Voronojeva popločenja, Voronojeva dekompozicija ili Dirchlet popločenja po Lejeune-u Dirichlet-u.

U najjednostavnijem slučaju, zadan je skup točaka *S* u ravnini koje su Voronojeva područja. Svaka točka *s* ima Voronojevu ćeliju *V(s)*, koja se naziva još Dirichletovom ćelijom. Ova ćelija sastoji se od točaka koje su bliže *s* nego bilo kojoj drugoj točci. Segmenti Voronojevog dijagrama su sve točke u ravnini koje su jednako udaljene od dvije susjedne točke. Voronojevi čvorovi su točke jednako odaljene od dviju ili više točaka.

Na slici 3.1. prikazan je primjer Voronojevog dijagrama.



Slika 3.1. Primjer Voronojevog dijagrama

3.1 Primjene Voronojevog dijagrama

Voronojev dijagram ima široku mogućnost primjene, pogotovo pri rješavanju geometrijskih problema. Udaljenost je implicitno uključena u gotovo sve primjene Voronojevog dijagrama.

Jedna od ranijih primjena Voronojevog dijagrama bila je na području epidemiologije kada je John Snow proučavao ulično širenje kolere 1854 u Soho-u u Engleskoj. On je pokazao korelaciju između područja na karti Londona i područja s najvišom smrtnošću od zaraze.

Voronojev dijagram može se koristiti kako bi se riješio problem najbližeg susjeda. Za dani skup od *n* položaja u ravnini treba, za slučajno odabranu točku *x*, pronaći područje najbliže *x*. Voronojev dijagram dijeli ravninu na regije čije točke *x* imaju zajedničko najbliže područje.

Uz dani Voronojev dijagram može se pronaći najveća prazna kružnica između skupa točaka u zatvorenom poligonu, npr, izgraditi prodavaonicu što je moguće dalje od postojećih, a da se još uvijek nalazi u određenom mjestu.

Voronojev dijagram koristi se u polimernoj fizici kako bi izrazio slobodan volumen polimera. Polimeri su velike molekule složene od strukturnih jedinica koje se ponavljaju. U klimatologiji, Voronojev dijagram koristi se kako bi se izračunala količina oborine na danom području. Voronojev dijagram koristi se i za proučavanje rasta šuma, a može biti koristan i kod razvijanja modela za predviđanje šumskih požara.

Na području računalne grafike Voronojev dijagram se koristi za generiranje tekstura organskog izgleda.

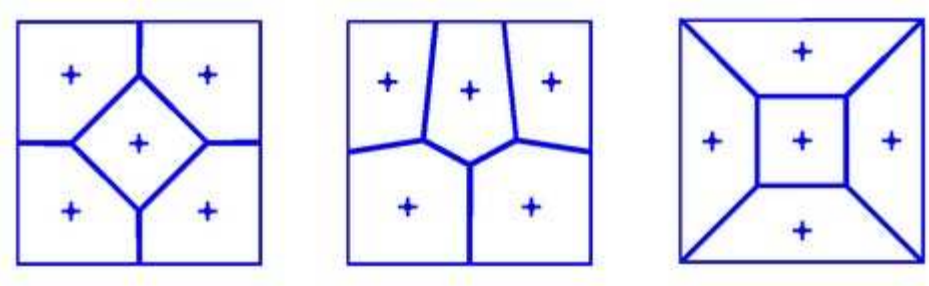
Pri robotskoj navigaciji Voronojev dijagram služi pronalasku čistih ruta. Ukoliko je točka prepreka, rubovi grafa biti će rute koje su najudaljenije od prepreka.

3.2 Centroidalni Voronojev dijagram

Centroidalni Voronojev dijagram je poseban slučaj Voronojevog dijagrama. Voronojev dijagram smatra se cetroidalnim ako je točka koja generira Voronojevu ćeliju ujedno i centar mase te ćelije. On se može smatrati optimalnom podjelom koja odgovara optimalnoj distribuciji generatora. Postoje brojni algoritmi kojima se može generirati centroidalni Voronojev dijagram. Najpoznatiji su Lloyd-ov algoritam i algoritam K-srednjih vrijednosti.

Centroidalni Voronojevi dijagrami koriste se kod kompresije podataka, te generiranja optimalnih površina, kvantizacija te grupiranja. Puno uzoraka iz prirode mogu se dobro aproksimirati Centroidalnim Voronojevim dijagramom.

Slika 3.3. prikazuje primjere centroidalnih Voronojevih dijagrama.



Slika 3.3. Centroidalni Voronojev dijagram

4 Algoritmi za izradu Voronojevog dijagrama

Formalna definicija Voronojevog skupa V_i *je sljedeća:*

$$V_i = \{w, z_i, z_j \in \Omega | d(w, z_i) < d(w, z_j), i = 1,2, \dots, K; i \neq j\} \quad (7)$$

pri čemu je Ω *otvoreni skup u* R^n*, a d funkcija udaljenosti.*

Za izradu Voronojevog dijagrama koriste se Fortune-ov algoritam i algoritam Podijeli-pa-vladaj.

4.1. Fortune-ov algoritam

Fortune-ov dijagram generira Voronojev dijagram iz skupa točaka u ravnini sa složenošću O(n log n). Ovaj algoritam objavio je Steven Fortune 1986.

Algoritam održava dvije linije koje se pomiču preko ravnine kako algoritam napreduje. Prva linija je ravna okomita linija koja se pomiče s lijeve strane na desnu preko ravnine. U bilo kojem trenutku tijekom trajanja algoritma ulazne točke koje se nalaze lijevo od ove linije ugrađuju se u Voronojev dijagram, dok se točke desno od linije ignoriraju. Druga linija je kompleksna krivulja koja se sastoji od parabola. Ova linija dijeli dio ravnine na kojoj je Voronojev dijagram poznat od ostatka ravnine. Za svaku točku koja se nalazi lijevo od prve linije može se definirati parabola od točaka koje su jednako udaljene od te točke i od linije. Druga linija je unija ovih parabola. Kako se prva linija pomiče, točke druge linije ocrtavaju rubove Voronojevog dijagrama.

Algoritam održava binarno stablo pretraživanja kao strukturu podataka te njima opisuje strukturu druge linije, te održava red prioriteta u kojem su navedeni potencijalni budući događaji koji mogu promijeniti strukturu druge linije. U ove događaje ubraja se dodavanje nove parabole drugoj liniji i uklanjanje krivulje iz druge linije. Ovo se događa kada prva linija postane tangenta na kružnicu koja se proteže preko tri ulazne točke čije parabole tvore segment na drugoj liniji. Svaki ovakav događaj može se prioritizirati po x-koordinati prve linije na točci u kojoj se pojavio događaj. Algoritam se sastoji od neprestanog uklanjanja događaja iz reda prioriteta usput pronalazeći promjene koje je događaj uzrokovao i ažuriranjem podatkovnih struktura. S obzirom da ima O(n) događaja koje treba procesirati, a vrijeme procesiranja pojedinog događaja je O(log n) ukupno vrijeme je O(n log n).

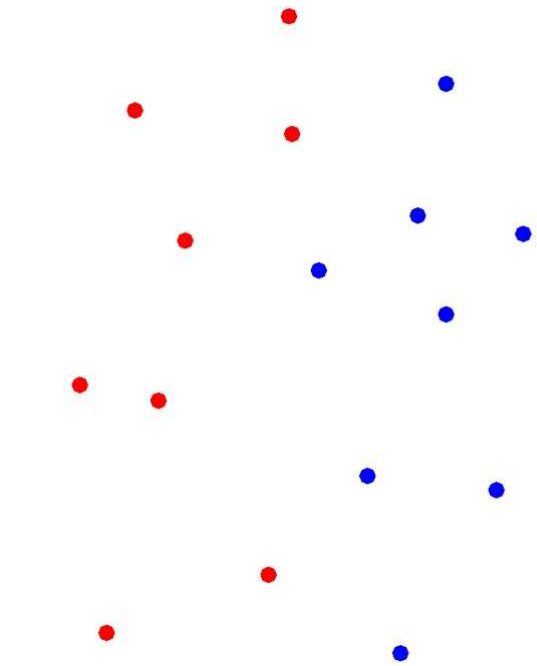
4.2 Algoritam Podijeli-pa-vladaj

Shamos i Hoey predstavili su prvi deterministički algoritam za izračun Voronojevog dijagrama u ravnini koji je optimalni u najgorem slučaju. Paradigma „Podijeli-pa vladaj“ jedna je od osnovnih paradigmi oblikovanja efikasnih algoritama. Paradigma rekurzivno dijeli originalni problem u nekoliko jednostavnijih podproblema koji su svi otprilike jednaki veličinom. Rješenje problema dobiva se spajanjem rješenja podproblema. U Podijeli-pa-vladaj algoritmu Shamosa i Hoey-a skup točaka S podijeljen je na dva podskupa S_L i S_R koji su otprilike jednake veličine. Nakon toga se Voronojev dijagram $Vor(S_L)$ od podskupa S_L i Voronojev dijagram $Vor(S_R)$ podskupa S_R računaju rekurzivno.

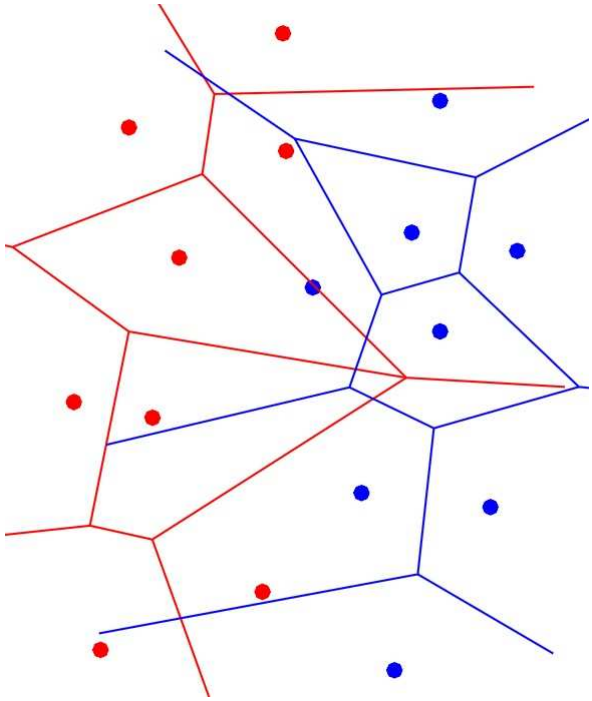
Na slikama 4.1 do 4.4 prikazana je jedna iteracija generiranja Voronojevog dijagrama algoritmom Podijeli-pa-vladaj.

Slika 4.1 prikazuje cijelu površinu s točkama koje generiraju Voronojev dijagram. Točke su podijeljene u dva podskupa, lijevi S_L označen crvenom bojom i desni S_R označen plavom bojom. Na slici 4.2 napravljeni su Voronojevi dijagrami lijevog i desnog podskupa. Slika 4.3 prikazuje spajanje lijevog i desnog podskupa Na slici 4.4. dan je krajnji rezultat spajanja lijevog i desnog podskupa.

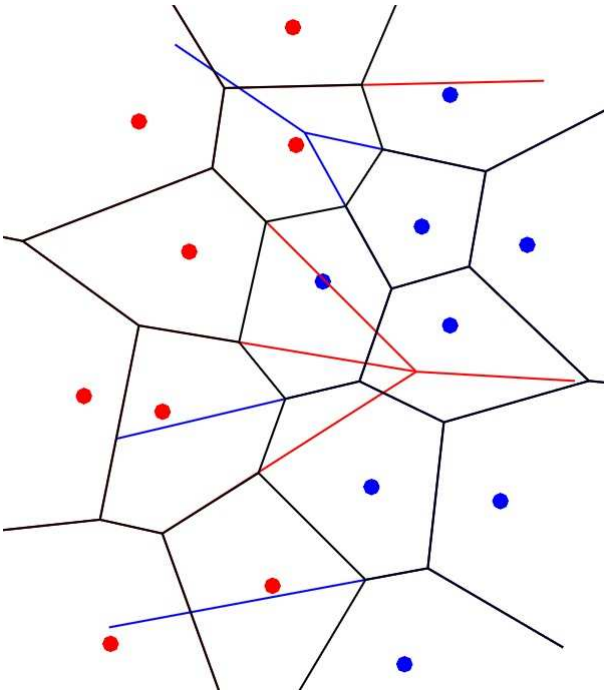
Postupak dijeljenja skupova na lijevi i desni podskup je rekurzivan. Dijeljenje se nastavlja sve dok se površine ne svedu na površine koje nemaju više od 3 točke.



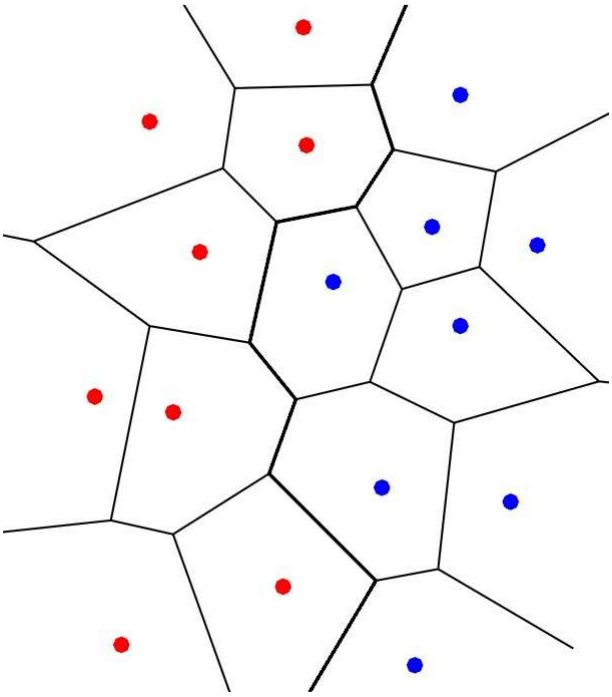
Slika 4.1. Podijela točaka na lijevi i desni skup



Slika 4.2. Izrada Voronojevog dijagrama za lijevi i desni skup točaka



Slika 4.3. Spajanje lijevog i desno Voronojevog dijagrama



Slika 4.4. Voronojev dijagram

5 Algoritmi za izradu centroidalnog Voronojevog dijagrama

Formalna definicija Voronojevog skupa V_i *je sljedeća:*

$$V_i = \{w, z_i, z_j \in \Omega | d(w, z_i) < d(w, z_j), i = 1,2, \dots, K; i \neq j\} \quad (8)$$

pri čemu je Ω *otvoreni skup u* R^n*, a d funkcija udaljenosti.*

Centroidalni Voronojev dijagram je posebni slučaj Voronojevog dijagrama u kojemu je z_i ujedno i centar mase Voronojeve regije. Centar mase ci regije Vi računa se kao:

$$c_i = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx} \quad (9)$$

pri čemu je $\rho(x)$ gustoća funkcije V_i.

Postoje brojni algoritmi za izradu centralnog Voronojevog dijagrama, no najčešće se koriste Lloyd-ov algoritam i algoritam K-srednjih vrijednosti

5.1 Algoritam K-srednjih vrijednosti

Za zadanu regiju Ω i gustoću funkcije $\rho(x)$,

1. Kreira se razdioba vjerojatnosti $P(x)$ za funkciju gustoće $\rho(x)$.
2. Koristeći metodu Monte Carlo odabire se početni skup od K točaka, $\{z_i\}_{i=1}^{K} \in \Omega$, koje odgovaraju $P(x)$.
3. Inicijalizira se težinska varijabla; J_i=1.
4. Odabire se $w \in \Omega$ u skladu sa $P(x)$.
5. Traži se z_i najbliži w_i te se označava sa z_i^*.
6. Postavlja se $z_i = \frac{J_i * z_i^* + w}{J_i + 1}$ i $J_i = J_i + 1$.
7. Koraci 4 do 5 se ponavljaju dok se ne postigne odgovarajuća preciznost.

5.1.1 Prednosti i nedostaci algoritma

Količina potrebnog računanja za generiranje točaka koje će konvergirati pravom centroidalnom Voronojevom dijagramu gotovo je jednaka za sve dimenzije. Broj

potrebnih iteracija je prilično velik s obzirom da algoritam ovisi o slučajno odabranim točkama.

Snaga ovog algoritma leži u njegovom korištenju slučajno odabranih točaka kako bi potakla konvergenciju, umjesto korištenja centara mase regija. Računalo vrlo lako generira slučajne točke, dok bi za računanje centara masa bile potrebne numeričke integracije koje su procesoru puno zahtjevnije. Usprkos velikom broju iteracija potrebnom da bi se kreirao centroidalni Voronojev dijagram, algoritam K-srednjih vrijednosti smatra se najbržim algoritmom za veći broj dimenzija.

5.2 Lloyd-ov algoritam

Za zadanu regiju *Ω* i gustoću funkcije *ρ(x)*,

1. Koristeći metodu Monte Carlo odabire se početni skup od K točaka, $\{z_i\}_{i=1}^{K} \in \Omega$.
2. Konstruira se Voronojev dijagram $\{V_i\}_{i=1}^{K}$ od *Ω* za točke$\{z_i\}_{i=1}^{K}$.
3. Računa se centar mase Voronojevih regija $\{V_i\}_{i=1}^{K}$, ovi centri mase su novi skupovi točaka $\{z_i\}_{i=1}^{K}$.
4. Koraci 2 i 3 se ponavljaju dok se ne postigne odgovarajuća preciznost.

5.2.1 Prednosti i nedostaci algoritma

Lloyd-ov algoritam puno je direktniji od algoritma K-srednjih vrijednosti pa je tako i manji broj iteracija potrebnih da bi se kreirao centroidalni Voronojev dijagram.

Za veće dimenzije je Lloyd-ov algoritam sporiji od algoritma K-srednjih vrijednosti zbog kompliciranog procesa numeričke integracije. No, Lloyd-ov algoritam omogućava laku manipulaciju oblikom, veličinom i pozicijom Voronojevih regija. Ova manipulacija se postiže odabirom odgovarajuće funkcije gustoće.

5.2.2 Performanse algoritma u dvodimenzionalnom prostoru

U Lloyd-ovom algoritmu generirane točke zamjenjuju se centrima masa odgovarajućih regija. S ažuriranim točkama računaju se nove Voronojeve regije. U dvodimenzionalnom prostoru postavljanje novih centara masa nije linearno

ograničeno te novi centri masa često skreću s predviđene putanje. Ako su ove devijacije male performanse su bolje nego kod velikih devijacija.

6 Programsko ostvarenje „Jigsaw“ kolaža

6.1 Uvod

Zadatak implementacije „Jigsaw“ slikovnog mozaika je da zadani spremnik (originalnu sliku) proizvoljnog oblika što kompaktnije napunimo sa sličicama koje po boji najbolje odgovaraju djeliću spremnika na koji ćemo smjestiti tu sličicu. Na mozaik gledamo kao na razmještaj sličica u kojemu je minimizirana funkcija energije.

Problem se definira na slijedeći način:

Za dani spremnik proizvoljnog oblika i za skup sličica proizvoljnog oblika T, treba naći skup oblika S takav da vrijedi:

Unija skupa S što vjernije predstavlja spremnik

Svaki član skupa S je translatirana i rotirana kopija nekog člana skupa T

Da bismo izračunali „Jigsaw“ slikovni mozaik uvodimo okosnicu temeljenu na funkciji energije gdje je mozaik definiran kao razmještaj sličica koji minimizira težinsku funkciju energije. Mijenjanjem težinskih faktora za pojedine energije, možemo dobiti različite tipove mozaika.

Kažemo da je razmještaj „Jigsaw“ slikovni mozaik kada minimizira energiju E definiranu kao:

$$E = w_c \cdot E_c + w_g \cdot E_g + w_o \cdot E_o + w_l \cdot E_l \tag{10}$$

pri čemu je:

E_c – energija boje; razlika između boje sličice i boje spremnika

E_g – energija praznine; veličina razmaka između sličica u razmještaju

E_o – energija preklapanja; veličina prostora na kojem se sličice međusobno prekrivaju

E_l – energija odnosa opsega i površine slike; kažnjava sličice koje bi nam mogle stvarati problema u daljnjem popunjavanju

6.2 Koraci algoritma

Ovaj algoritam rađen je na temelju algoritma koji su Kim i Pellacini predstavili na Siggraph-u 2002 [1], no u nekim dijelovima je izmijenjen. U daljnjem tekstu bit će predstavljeni koraci algoritma pomoću kojega je implementiran „Jigsaw“ slikovni mozaik. Koraci ovog algoritma su slijedeći:

1. Priprema ulaznih podataka
2. Računanje prosječnog oblika sličica pomoću kojih ćemo napuniti spremnik
3. Izrada Centroidalnog Voronojev dijagrama (CVD)
4. Traženje površine unutar CVD-a s najmanjim brojem susjeda na koju ćemo postaviti slijedeću sličicu
5. Traženje najbolje moguće sličice
6. Postavljanje sličice u spremnik i priprema za postavljanje nove sličice
7. Optimizacija
8. Implementacija korisničkog sučelja

6.3 Priprema ulaznih podataka

Prije započinjanja s radom, program za izradu „Jigsaw“ slikovnog mozaika mora imati odgovarajuće ulazne podatke potrebne za rad. Ulazni podaci su originalna slika proizvoljnog oblika i što veći skup sličica kojima popunjavamo originalnu sliku. Originalna slika može biti proizvoljnog oblika. Crni dio slike, odnosno dio slike s RGB vrijednostima (0,0,0) ne popunjava se. Sličice se prije upotrebe podese tako, da su im okviri jednake veličine i pravokutnog su oblika. I kod njih, kao i kod originalne slike, područja s RGB vrijednostima (0,0,0) se ignoriraju. Svaka sličica se rotira pa tako svaku sličicu imamo zarotiranu za 0°, 90°, 180° i 270° stupnj eva.

Kao ulazni podatak koristimo i tablicu s informacijom o rubnim područjima sličice. Nju radimo tako da pravokutnik u kojem se nalazi sličica razdijelimo na kvadratiće jednakih veličina. Ukoliko se u nekom kvadratiću nalazi rub (granica između boje i crnog područja) naše sličice, kvadratić će biti označen. Tablica se radi tako da za svaki redak kvadratića svake sličice oformimo jedan redak, dok nam stupci predstavljaju stupce kvadratića. Ukoliko neki redak kvadratića određene sličice ima

rub u nekom stupcu, taj stupac bit će označen. Ova tablica poslužit će u izboru najbolje moguće sličice. Sličice čiji se rub neće dovoljno dobro poklapati sa rubom površine koju želimo popuniti bit će odmah odbačene.

Na slici 6.1. prikazana je podjela sličice na mrežu od 8 redaka i 8 stupaca. Područja u kojima se nalazi rub sličice obilježena su znakom x. Ispod sličice nalazi se isječak iz tablice informacija o rubnim područjima. Tablica je pohranjena u bazu podataka za što se koristi program MS Access. U koloni „fileName“ zapisano je ime datoteke iz koje smo uzeli sličicu. Kolona „row“ podrazumijeva retke sličice, ostali stupci u tablici označavaju stupce sličice. Ukoliko se u nekom stupcu za određeni redak određene sličice nalazi rub sličice, stupac je označen s „yes“, u obrnutom slučaju, imamo oznaku „no“.

Slika 6.1. Podjela sličice na mrežu

fileName	row	0	1	2	3	4	5	6	7
bu12-0.bmp	0	No	Yes	Yes	No	No	Yes	Yes	No
bu12-0.bmp	1	Yes	Yes	No	No	No	No	Yes	Yes
bu12-0.bmp	2	Yes	No	No	No	No	No	No	Yes
bu12-0.bmp	3	No	No	No	No	No	No	No	No
bu12-0.bmp	4	No	No	No	No	No	No	No	No

fileName	row	0	1	2	3	4	5	6	7
bu12-0.bmp	5	Yes	No	No	No	No	No	No	Yes
bu12-0.bmp	6	Yes	Yes	No	No	No	No	Yes	Yes
bu12-0.bmp	7	No	Yes	Yes	No	No	Yes	Yes	No

Slika 6.1. Uvećan prikaz sličice i njeni podaci u tablici informacija o rubnim područjima

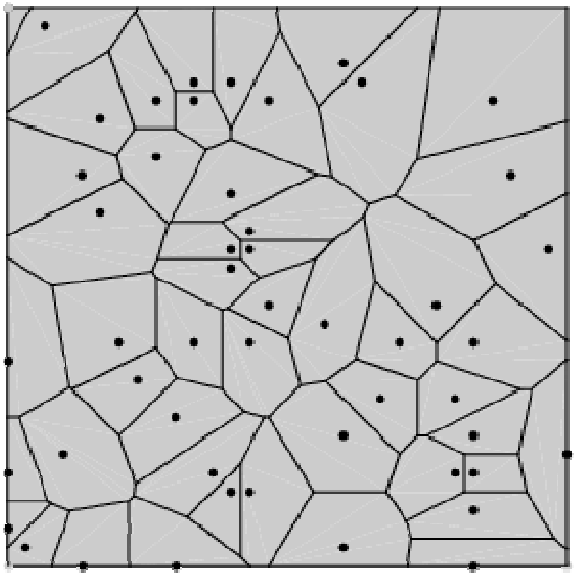
6.4 Računanje prosječnog oblika sličica pomoću kojih ćemo napuniti spremnik

Kad bi se cijeli spremnik gledao kao jedna cjelina traženje najboljeg razmještaja bilo bi vremenski vrlo zahtjevno jer bismo trebali isprobati sve moguće kombinacije sličica da dođemo do najboljeg. Umjesto toga, spremnik se razdjeljuje na površine koje su približno jednake prosječnoj površini sličica, a popunjavaju se jedna po jedna površina.

Prosječan oblik sličice računa se tako da se prođe svaki slikovni element unutar okvira sličice i za svaki slikovni element se gleda da li više od polovice sličica ima boju na tom slikovnom elementu. Ukoliko ima, kvadratić u kojem se iscrtava prosječan oblik oboji se u bijelu, ako ne, on ostaje crn.

6.5 Izrada Centroidalnog Voronojev dijagrama (CVD)

Voronojev dijagram je površina s *n* točaka podijeljena na *n* površina u kojoj svaku površinu sačinjavaju one točke koje su jednoj određenoj točci bliže nego bilo kojoj drugoj od *n*-1 točaka. Voronojev dijagram ima vrlo široko područje primjene. Koristi ga se u raznim znanostima među kojima se nalaze i antropologija, astronomija, biologija, geografija, geologija, marketing, zoologija itd. Voronojev dijagram prikazan je na slici 6.2.



Slika 6.2. Voronojev dijagram – točke i njima pripadajuće površine

Centroidalni Voronojev dijagram je Voronojev dijagram gdje su točke pomoću kojih se tvore površine ujedno i centroidi tih površina. Iz Voronojev dijagrama Centroidalni Voronojev dijagram možemo dobiti iterativnim postupkom tako što se nakon svake nove podjele na površine uzmu nove točke koje su centroidi tih površina i na temelju njih opet grade nove površine. Na slici 6.3. prikazan je Voronojev dijagram nakon prvog, slučajnog razmještaja točaka, te 3. i zadnja 5. iteracija ovog postupka.

Slika 6.3. Iterativni postupak dobivanja CVD-a

Izrada CVD-a može se smatrati centralnim dijelom algoritma. CVD nam služi da bismo spremnik podijelili na površine približno jednake površinama prosječnog oblika sličica.

U ovom dijelu koristi se prethodno izračunat prosječni oblik sličica, odnosno njegova veličina. Veličinu spremnika podijelimo s veličinom sličica da bismo dobili

broj točaka za izradu Voronojev dijagrama. Točke se nasumično raspodjeljuju po površini. Pravilno raspodjeljivanje točaka po površini moglo bi skratiti rad ovog, vremenski najzahtjevnijeg, dijela algoritma, ali rezultati dobiveni na taj način nisu bili zadovoljavajući jer daju površine kvadratnog oblika.

Postoje dvije opcije izrade Voronojevog dijagrama: izrada jednostavnim ili Fortune-ovim algoritmom. Jednostavni algoritam radi na način da prolazi svim točkama slike te provjerava kojoj od točaka za izradu Voronojevog dijagrama pripada svaka točka.

U 5 iteracija dobijemo Voronojev dijagram približno jednak Cetroidalnom Voronojev dijagramu. Svaka od ovih iteracija vremenski je zahtjevna zato što se za svaku točku mora ispitati njena udaljenost od ostalih točaka. Uzima se 5 iteracija zato što je eksperimentalnim putem zaključeno da je to minimalni broj iteracija kod kojeg je Voronojev dijagram dovoljno sličan Centralnom Voronojev dijagramu.

Da bi bilo jednoznačno određeno koja površina pripada pojedinoj točci, površine se boje bojama čija se vrijednost računa pomoću x i y koordinate točke. Na taj način izbjegava se i to da se dvije različite površine oboje istom bojom što bi u kasnijem radu programa moglo dovesti do grešaka. Boja se računa na sljedeći način:

R-komponenta = x-koordinata MOD 100

G-komponenta = y-koordinata MOD 100

B-komponenta = broj stotica x-koordinate *10 + broj stotica y-koordinata

npr., za točku (355,266) – R=55; G=66; B=32 (3 je broje stotica u x-koordinati, a 2 je broj stotica u y-koordinati)

Nedostatak ovog proračuna je u tome što smo okvir spremnika ograničili na veličinu 999x999 slikovnih elemenata, a iskorištava se samo 99x99x99 od 255x255x255 mogućih kombinacija za boje. Na slici 6.4. vidi se CVD obojan na ovaj način.

Slika 6.4. CVD u bojama koje odgovaraju centroidima površina

6.6 Traženje površine unutar CVD-a sa najmanjim brojem susjeda na koju ćemo postaviti slijedeću sličicu

Nakon što je gotov CVD, potrebno je odabrati površinu na koju će se postaviti slijedeća sličica. Odabire se površina s najmanjim brojem susjednih površina pa je zato u ovom koraku potrebno odrediti broj susjednih površina svake površine. To se radi tako da se pravcima spoje centroidi svih površina. Ukoliko između spojenih centroida postoji još neka boja, boja koje ne pripada površini niti jednog centroida, zaključuje se da površine oko centroida nisu susjedi. U obrnutom slučaju, oni se proglašavaju susjedima, a u datoteku se zapisuju susjedi svake površine koji se na kraju prebroje. Površina s najmanjim brojem susjeda bit će ispunjena sličicom.

6.7 Traženje najbolje moguće sličice

Na slici 6.5. prikazane su neke od preko 5084 sličica pomoću kojih popunjavamo spremnik. U izboru najbolje moguće sličice koristi se tablica s informacijom o rubnim područjima sličice. Brojimo kvadratiće rubnih područja koji su zajednički kod površine za popunjavanje i kod sličice. Samo sličice čija se rubna područja preklapaju s rubnim područjima odabrane površine u dovoljnoj mjeri doći će u obzir za daljnju analizu. Samo u slučaju da ne postoji niti jedna sličica koja zadovoljava minimalne uvjete sve sličice bit će analizirane.

Slika 6.5. Primjeri sličica pomoću kojih se popunjava spremnik

U daljnjoj analizi računaju se pojedine komponente energije za svaku sličicu.

E_c – energija boje

Za svaku točku u sličici gleda se kvadratna udaljenost RGB komponenti sličice i spremnika

E_g – energija praznine

Zbrajaju se sve točke koje postoje u odabranoj površini, a crne su u sličici

E_o – energija preklapanja

Zbrajaju se sve točke koje postoje u sličici, a izlaze iz područja odabrane površine

E_l – energija odnosa opsega i površine

Računa se po formuli:

$$E_l = w_a \cdot površina + (1-w_a) \cdot opseg^2 \quad (11)$$

Težinski faktori pojedinih energija u izrazu određeni su eksperimentalno.

6.8 Postavljanje sličice u spremnik i priprema za postavljanje nove sličice

Poznate su gornja-lijeva i donja-desna koordinata površine unutar originalne slike koju treba prekriti odabranom sličicom. Sličica se kopira slikovni element po slikovni element u spremnik. Crna boja u sličici se ne kopira. Površina na koju je zalijepljena sličica vadi se iz spremnika. Iz spremnika se vadi i sva površina koja je pripadala površini koju smo popunjavali, a koja se nalazila između postavljene sličice i ruba spremnika. Površina koju nije prekrila sličica, a koja graniči sa drugim površinama ostaje u spremniku. Cijeli postupak se ponavlja s ostatkom spremnika. Postupak je gotov kada u spremniku nema više površina.

6.9 Optimizacija

Optimizacija gotovog razmještaja izvršava se nakon što je razmještaj završen. Pri optimizaciji svaka slika rotira se u položaj u kojem ima minimalnu energiju. Rotacija započinje od slike koja je zadnja postavljena, a završava sa zadnje postavljenom slikom. Za to nam služi datoteka u koju smo za vrijeme postavljanja sličica upisivali njihova imena. Prije početka optimizacije ovu datoteku potrebno je preokrenuti, tako da dobijemo naziv zadnje sličice na početku, a naziv prve sličice na kraju datoteke. Da bismo za svaku sliku izračunali njen najbolji položaj trebamo znati kako bi razmještaj izgledao bez samo te slike. Za to nam služe dvije pomoćne slike, u jednu stavljamo sličice u već optimiziranom razmještaju dok je na drugoj slici još neoptimiziran razmještaj, u kojem se ne nalaze sličice iz već optimiziranog razmještaja, niti sličica čiji se položaj trenutno podešava.

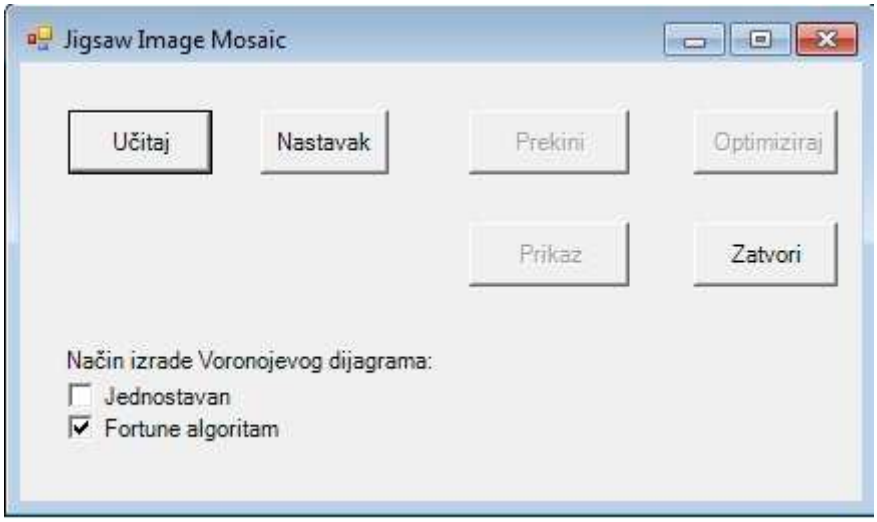
6.10 Implementacija korisničkog sučelja

Preko korisničkog sučelja korisniku se daje mogućnost učitavanje originalne slike, pokretanje i zaustavljanje izrade neoptimiziranog dijela mozaika, optimizacija, pregled rezultata i zatvaranje izbornika.

S obzirom da se radi o algoritmu čije izvršavanje traje i do nekoliko sati, bilo je potrebno omogućiti korisniku prekid i kasnije nastavljanje s radom. Program može nastaviti s radom i nakon gašenja zato što su sve informacije bitne za nastavak rada pohranjene u datotekama. Program se može prekinuti tijekom izrade neoptimiziranog razmještaja pritiskom na dugme „Prekini“. Pritisak na to dugme bit će omogućen nakon što program završi s izračunom prosječne površine. Ukoliko program pronađe već izračunatu površinu neće je računati ponovno. S obzirom da se postojeće sličice ne mijenjaju i površina uvijek ostaje ista. Za dio programa koji radi neoptimiziranu konfiguraciju pokrenuta je nova dretva. Ona šalje poruke glavnom programu da li se promijenilo stanje o tome može li se ili ne prekinuti njen rad. Njen rad ne smije biti prekinut u trenutku pisanja u datoteke da se podaci iz datoteke ne bi izgubili. Ukoliko se izrada neoptimiziranog razmještaja pokuša prekinuti u trenutku kada prekid nije dozvoljen, korisniku se javlja poruka da mora pričekati završetak trenutne faze rada. Nakon toga, izvršavanje se prekine, a korisniku se nudi dugme za nastavak „Nastavak“. Ako korisnik potpuno izađe iz programa, program će na temelju datoteka

prepoznati da je u tijeku rada neoptimiziranog razmještaja i ponudit će mu tipku „Nastavak“. Bez obzira na to, korisnik se može odlučiti za izradu mozaika neke druge slike. Ukoliko pritiskom na dugme „Učitaj“ odabere neku drugu sliku neće više moći nastaviti s radom na prvoj slici.

Nakon što je neoptimizirani razmještaj napravljen, korisniku se javlja odgovarajuća poruka. Korisnik može odmah prijeći na optimizaciju. Ukoliko korisnik ne optimizira razmještaj, to može napraviti i prilikom sljedećeg otvaranja programa. Program će znati da je pohranjen gotov neoptimiziran razmještaj spreman za optimizaciju.



Slika 6.6. Izgled korisničkog sučelja

Na slici 6.6 vidi se izgled korisničkog sučelja prilikom pokretanja programa. Dugme „Učitaj“ je aktivno jer u svakom trenutku možemo odustati od dosadašnjeg slike i započeti rad na novoj slici. Dugme „Nastavak“je aktivno jer je program pronašao već započeti rad i daje mogućnost nastavka tog rada. Dugme „Zatvori“ je aktivno jer u ovom trenutku možemo odustati od bilo kakvih daljnjih akcija. U trenutku kad pritisnemo dugme „Nastavak“ sva ostala dugmad bit će onemogućena do trenutka kad program dođe u fazu gdje će prekid biti moguć. U tom trenutku dugme „Prekini“ bit će omogućeno.

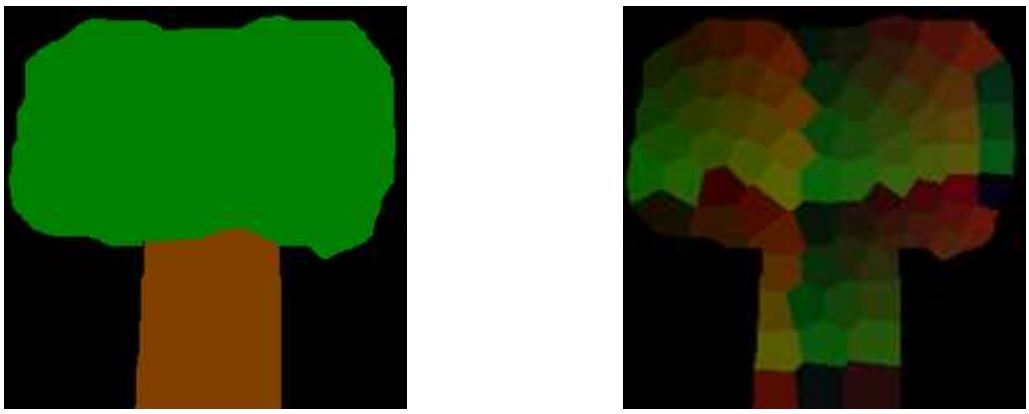
Prije početka stvaranja novog „Jigsaw“ mozaika imamo mogućnost izrade Voronojevog dijagrama jednostavnim ili Fortune-ovim algoritmom. Jednostavni algoritam podrazumijeva usporedbu svake točke slike sa svim ostalim kako bismo vidjeli kojoj površini dijagrama pripada određena točka. Fortune algoritam radi

Voronojev dijagram ranije opisanim Fortune-ovim algoritmom. Ukoliko prekidamo rad programa pamti se odabrani algoritam te se njime nastavlja izvođenje programa.

6.11 Slikovni prikaz faza rada programa

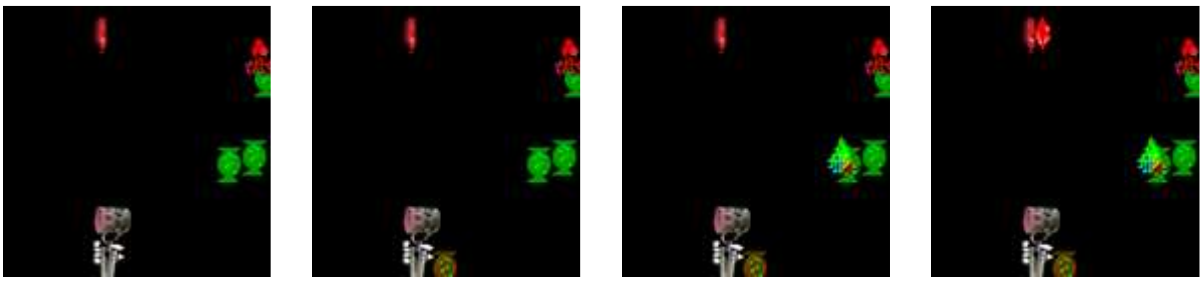
U ovom odjeljku, pomoću slika prikazane su faze rada programa. Na slici 6.8 prikazane su originalna slika i njezin CVD.

ORIGINALNA SLIKA **CVD DIAGRAM ORIGINALNE SLIKE**



Slika 6.8. Originalna slika i njezin centroidalni Voronojev dijagram

Na slici 6.9 prikazano je postavljanje sličica u spremnik. U prvoj od prikazanih slika, već se nalazi 8 sličica, u sljedećem koraku, u donji dio slike dodana je sličica smeđeg bombona. U sljedeća dva koraka dodani su i bor, a zatim i usne. Ovdje se vidi da se spremnik počinje popunjavati od rubova.

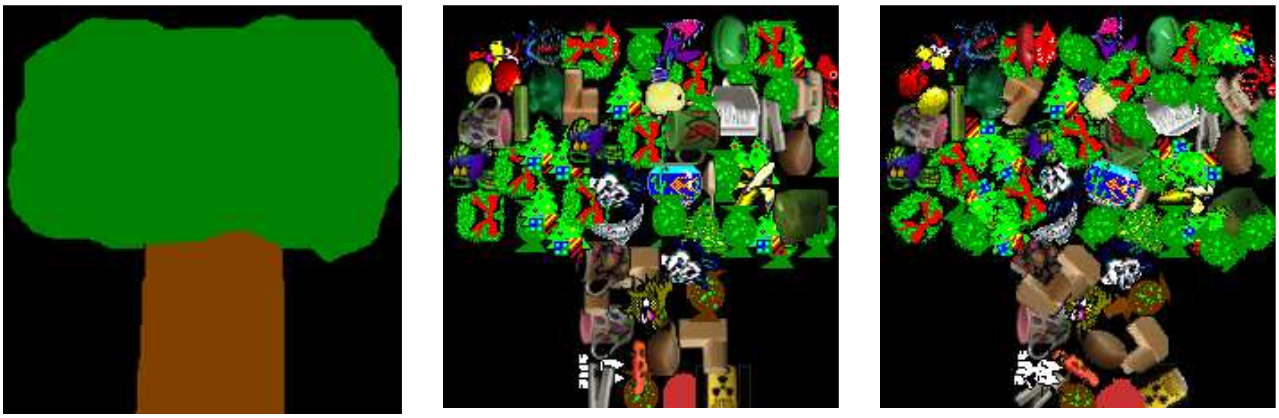


Slika 6.9. Postavljanje sličica u spremnik

Na slici 6.10 prikazano je uklanjanje popunjenih površina iz spremnika. Novi spremnik sastoji se od starog spremnika iz kojeg je izuzeta zadnja zalijepljena sličica zajedno sa dijelom površine CVD-a na koju je zalijepljena. Svaki od dolje prikazanih spremnika stvoren je nakon što su postavljene sličice koje se nalaze na gornjoj slici.

Slika 6.10. Uklanjanje popunjenih površina iz spremnika

Na slici 6.11 prikazani su redom originalna slika, te rezultati implementirane aplikacije: neoptimizirani te optimizirani „Jigsaw mozaik“.



Slika 6.11. Originalna slika, neoptimizirani „Jigsaw“ slikovni mozaik, te optimizirani „Jigsaw“ slikovni mozaik

7 Podaci o implementaciji i ograničenjima

Program je napisan u Microsoft Visual Studio 2010, C# jeziku.

Veličina ulazne slike ograničena je na 512x512 slikovnih elemenata. Razlog ograničenja je način izračunavanje boje iz koordinata točke što se koristi kod CVD-a. Ovo ograničenje omogućilo bi rad sa slikama do 999x999 slikovnih elemenata. Zbog efikasnijeg korištenja memorije ovo ograničenje pomaknuto je na 512x512 slikovnih elemenata.

7.1 Vremenska mjerenja

Sva vremenska mjerenja obavljena su na računalu slijedećih karakteristika:

Intel Core2 Duo CPU, 4 GB RAM, 64-bitni operacijski sustav

Okviri svih sličica kojima se ispunjava mozaik veličine je 32x32 slikovnih elemenata. Sličica ima 5084, odnosno 1271 različitih sličica u rotacijama od po 90°. Sve sličice ulaze u natječaj za izbor najbolje sličice.

Originalna slika veličine: 295 x 215 slikovnih elemenata – 69 sličica

Broj ne-crnih slikovnih elemenata: 22307

Opcija izrade mozaika s jednostavnim algoritmom generiranja Voronojevog dijagrama

Vrijeme potrebno za izvršavanje neoptimiziranog mozaika: 45 min

Vrijeme potrebno za izvršavanje optimizacije: 30 min

Vrijeme potrebno za dobivanje CVD-a: 2 min

Vrijeme potrebno za pronalaženje najbolje sličice: 36 s

Opcija izrade mozaika sa Fortune-ovim algoritmom generiranja Voronojevog dijagrama

Vrijeme potrebno za izvršavanje neoptimiziranog mozaika: 20 min

Vrijeme potrebno za izvršavanje optimizacije: 30 min

Vrijeme potrebno za dobivanje CVD-a: 8 s

Vrijeme potrebno za pronalaženje najbolje sličice: 36 s

Vremenska mjerenja iščitana su iz vremena kreiranja datoteka stvorenih u određenim fazama rada programa i iz poruka koje ispisuje program u „Debug“ načinu rada. Poruke se šalju kada se završi neka faza rada, a svaka poruka ispisuje i vrijeme kada je kreirana. Osim okvira originalne slike za vremensko trajanje rada programa bitno je i koliko ima ne-crnih slikovnih elemenata u originalnoj slici (slikovnih elemenata na temelju kojih se radi mozaik). U tu svrhu napravljen je jednostavni program koji je prebrojio sve ne-crne slikovne elemente.

8 Zaključak

Cilj ovog rada bilo je proučiti tehnike prikaza kolaža, koje se ubrajaju u jednu od tehnika nefotorealističnog prikaza, s nalgaskom na „Jigsaw“ kolaž čija izrada je i implementirana. Kako je izrada Voronojevog dijagrama jedan od važnijih koraka ove implementaciji, njegova izrada je detaljno opisana u radu.

Implementacija „Jigsaw“ kolaža postigla je zadovoljavajuće vizualne rezultate. Vizualni rezultati bili bi još bolji da su se koristile veće originalne slike uz istu veličinu sličica, međutim, zbog vremena potrebnog da se obrade veće originalne slike, obrađene su samo slike manjeg formata. Bolji vizualni učinak može se postići i korištenjem većeg broja sličica za popunjavanje.

Uvođenjem opcije izrade Voronojevog dijagrama Fortune-ovim algoritmom znatno je ubrzan inače najdugotrajniji korak izrade mozaika, a to je izrada Voronojevog dijagrama.

Algoritam koji su predstavili Kim i Pellacini na SIGGRAPH-u 2002 [1] slijedi se u zamisli da se prvo izradi CVD, te nakon toga popuni površina s najmanjim brojem susjeda, i kod izbora najbolje sličice koji se temelji na računanju minimalne energije i na tablici informacija o rubnim područjima. Za razliku od originalnog algoritma, u ovoj implementaciji se nakon postavljanja sličice u spremnik iz spremnika ne uklanja samo prostor na koji je postavljena sličica, nego i dijelovi koji bi mogli smetati u nastavku rada. Razlika je i u tome što se rotacija ne izvršava prilikom postavljanja pojedine slike. To bi moglo biti vremenski previše zahtjevno kod potrage za najboljom sličicom. Rotacija slika u ovom slučaju događa se tek u fazi optimizacije, nakon što je neoptimiziran razmještaj već postavljen. U originalnom algoritmu izvršavaju se i transformacije slika koje se u ovoj implementaciji ne rade.

9 Literatura

1. Kim, J. , Pellacini F.: *Jigsaw Image Mosaics*, SIGGRAPH 2002, str. 657 – 654

2. Gianpiero Di Blasi: *Fast Techniques for Non Photorealistic Rendering* 2006

3. *Wikipedia,* http://www.wikipedia.com

4. Kostrenčić, M. , Protega, M.: Enciklopedija Leksikografskog Zavoda, Mozaik 1961, Vol 5. , str. 285

5. M. McDerby, L. Lever: *State of the Art Non-Photorealistic Rendering (NPR)*

Techniques 2006

6. David Ebert, Penny Rheingans: *Volume Illustration: Non-Photorealistic Rendering of Volume Models*

9. Franz Aurenhammer, Rolf Klein: Voronoi Diagrams

10. Brauwerman, Zoll, Farmer, Gunzburger: *Centroidal Voronoi Tessellations Are Not Good Jigsaw Puzles*

11. Marko Čupić: Skripta iz računalne grafike

12. Vera Sacristán: *Algorithms for constructing Voronoi diagrams*

10 Sažetak

Rad se bavi nefotorealističnim tehnikama prikaza korištenjem kolaža, te se detaljno razrađuje postupak izrade nefotorealističnih tehnika prikaza uz upotrebu Voronojevog dijagrama za određivanje pozicija pločica kolaža. Implementirana je izrada „Jigsaw“ kolaža, pri čemu se Voronojev dijagram izrađuje pomoću dva algoritma.

Ključne riječi: nefotorealistični prikaz, mozaik, Voronojev dijagram, centroidalni Voronojev dijagram, Fortune-ov algoritam

11 Abstract

This paper describes non-photorealistic rendering using mosaic. Non-photorealistic rendering technique where Voronoi diagram is used for tile placement is shown in details. „Jigsaw“ image mosaic is implemented and the choice between two algorithms for Voronoi diagram calculation is available.

Keywords: non-photorealistic rendering, mosaic Voronoi diagram, centroidal Voronoi diagram, Fortune algorithm.