SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 558

Otkrivanje promjena u zaštićenim slikama

Vedran Šuman

Zagreb, svibanj, 2009. godina

Sadržaj

1. Uvod

Cilj završnog rada je pojasniti digitalne vodene žigove, s naglaskom na lomljive digitalne vodene žigove i Stohastičku difuzijsku pretragu (eng. *Stochastic diffusion search*), u daljnjem tekstu SDS, te njihovu moguću zajedničku primjenu za otkrivanje promjena u zaštićenim slikama.

Zaštićivanje sadržaja vodenim žigovima je proces umetanja informacija u digitalni signal, koje je nemoguće poništiti ili izmijeniti.

SDS je algoritam iz obitelji evolucijskih algoritama koji se primarno koristi za probleme optimizacije, a može se svrstati u potkategoriju *swarm inteligence* (inteligencija roja).

U okviru ovog rada napravljen je kratki teorijski uvod u problematiku. Opisane su primjene i problemi primjene vodenih žigova. Također, opisana je jedna tehnika umetanja lomljivih digitalnih vodenih žigova.

Opisan je pobliže sam SDS algoritam i pojašnjen njegov pseudokod. Prikazan je jedan problem na koji se može primijeniti SDS i njegova usporedba s nekoliko drugih algoritama.

U praktičnom dijelu opisano je programsko rješenje za problem pretrage u kojem je korišten SDS, potom primjena tog rješenja na pronalaženje promjena u slikama. Također, opisan je i eksperiment sa programom koji umeće lomljive digitalne vodene žigove, te ih izvlači i daje odluku da li je slika izmijenjena ili ne.

2. Digitalni vodeni žigovi

2.1 Uvod u digitalne vodene žigove

Ubrzanim razvojem Interneta i broadband tehnologija digitalni mediji postaju lako dostupni. Javlja se problem zaštite autorskih prava istih, primjer je ovogodišnja tužba protiv *The Piratbaya*. Jedna od tehnika koja se nudi za navedeni problem su digitalni vodeni žigovi. Prva konferencija na tu temu održana je 1996. godine i otad se stalno povećava broj znanstvenika koji rade na tom području. [4]

Zaštićivanje digitalnih sadržaja digitalnim vodenim žigovima je proces umetanja informacija u digitalni signal. Poželjno je da nakon umetanja bude nemoguće ili jako teško ukloniti te informacije. Digitalni signal koji se zaštićuje može biti audio signal, video signal ili slika. Ukoliko se signal kopira, dodane informacije se također kopiraju.

Postoje vidljivi i nevidljivi digitalni vodeni žigovi. Vidljivi digitalni vodeni žigovi su dodane informacije koje se mogu vidjeti u slikama ili video zapisu. Obično je ta informacija logo ili tekst koji identificira vlasnika sadržaja, npr. logo televizijske kuće u nekoj reportaži.

Nevidljivi digitalni vodeni žigovi su informacije koje se dodaju digitalnim signalima na taj način da krajnji korisnik te informacije ne vidi, iako je te skrivene informacije određenim tehnikama moguće otkriti. Važna primjena nevidljivih digitalnih vodenih žigova je zaštita autorskih prava na način koji onemogućuje ili znatno otežava nedozvoljeno kopiranje digitalnih sadržaja. Proces umetanja nevidljivih digitalnih vodenih žigova se svodi na kodiranje signala i njegovo dekodiranje. Proces kodiranja [4] se može prikazati kao jednadžba (2.1):

$$X' = E_K(X, W) \tag{2.1}$$

Gdje je X originalna slika, W je informacija (vodeni žig) koji se umeće, K je korisnikov ključ i E je funkcija umetanja informacije. Ovisno o funkciji E i naravi algoritma umetanja vodenog žiga, metoda otkrivanja ili izvlačenja mogu poprimiti jako različite i specifične oblike. Glavna razlika tih metoda je da li zahtijevaju ili ne zahtijevaju originalnu sliku za izvlačenje te informacije. Tehnike koje ne zahtijevaju originalnu sliku za izvlačenje informacija se nazivaju zaboravne, javne ili slijepe tehnike. Za izvlačenje informacija tim tehnikama vrijedi:

$$\hat{W} = D_{K'}(\hat{X}') \tag{2.2}$$

gdje je $\hat{X}$ slika s digitalnim vodeni žigom, K ključ za izvlačenje informacija, D funkcija otkrivanja, a $\hat{W}$ je dobivena informacija. Zaboravne tehnike su jako privlačne korisnicima baš ih razloga jer ne zahtijevaju originalnu sliku.

Nevidljivi digitalni vodeni žigovi se, također, mogu podijeliti na robusne i lomljive.

Robusni su dizajnirani da izdrže uobičajene transformacije slika kao što su kompresija, podrezivanje, dimenzioniranje, naglašavanje kontrasta, skeniranje itd. što ih čini dobrim alatom za utvrđivanje vlasništva.

Lomljivi digitalni vodeni žigovi će biti detaljnije pojašnjeni kasnije u tekstu. 2.4 Zbog svega navedenog digitalni vodeni žigovi su postali jako važni, te se posvećuje velika pažnja njihovom razvoju. Njihov naziv potječe od vodenih žigova koji se koriste za zaštitu novčanica.

U nastavku će se pretpostaviti da je digitalni signal koji se koristi statička slika.

2.2 Primjene digitalnih vodenih žigova

Kao što se može zaključiti iz uvoda digitalni vodeni žigovi imaju brojne primjene. Neke najpoznatije i najkorisnije su navedene u daljnjem tekstu.

2.2.1 Dokazivanje vlasništva

Vlasnik dokumenta stavljanjem vodenog žiga potvrđuje svoje vlasništvo nad dokumentom.

U svrhu utvrđivanja vlasništva Alice može generirati digitalni vodeni žig koristeći svoj privatni ključ koji doda u obliku informacije u originalnu sliku. Nakon toga može svoju sliku zaštićenu digitalnim vodenim žigom staviti na Internet gdje će biti javno dostupna. Poslije, kad će se Bob boriti za vlasništvo slike dobivene iz javno dostupne slike, Alice može proizvesti originalnu sliku bez digitalnog vodenog žiga, i dokazati postojanje žiga u Bobovoj slici. Kako je originalna Alicina slika nedostupna Bobu, on ne može učini isto. Kako bi ovaj način dokazivanja vlasništva radio vodeni žig mora 'preživjeti' operacije koje su namijenjene njegovu uklanjanju. Također, vodeni žig bi trebao biti napravljen tako da se ne može krivotvoriti, jer Alice ne želi biti odgovorna za slike koje nisu njezino vlasništvo, odnosno, za slike koje nije ona napravila.

2.2.2 Praćenje kopija sadržaja.

Svaki kupac dokumenta stavljanjem svog vodenog žiga u dokument, pomaže u praćenju izvora nelegalnih kopija dokumenta.

Vlasnici multimedijskih sadržaja koji su elektroničkim putem distribuirani Internetom žele obeshrabriti nedozvoljeno kopiranje i distribuiranje svojih sadržaja. To čine pridruživanjem posebnog vodenog žiga ili otiska prsta u svaku kopiju sadržaja. Ukoliko se poslije pronađe nedozvoljena kopija sadržaja, originalna kopija iz koje je nastala nedozvoljena kopija se može utvrditi izvlačenjem otiska prsta. Ovaj vodeni žig, odnosno otisak prsta, također mora biti nevidljiv i otporan na pokušaje krivotvorenja, uklanjanja ili poništavanja. Dodatno, ovakav vodeni žig mora biti otporan i na tajne ugovore. To znači da ukoliko određeni broj korisnika ima istu sliku, ali s različitim otiscima prstiju, ne smiju biti u mogućnosti napraviti kopiju slike bez svih otisaka prstiju od drugih korisnika.

2.2.3 Kontrola kopiranja i prevencija

Vodeni žigovi mogu sadržavati informacije o pravilima uporabe i kopiranja sadržaja dokumenta. Ta pravila mogu biti oblika ‘zabrani kopiranje’ ili ‘dozvoli stvaranje određenog broja kopija’.

U zatvorenim sustavima gdje je za gledanje mutimedije potreban posebni sklop za gledanje ili kopiranje sadržaja, može se umetnuti digitalni vodeni žig koji će sadržavati informaciju o tome koliko puta je neki sadržaj kopiran. Svaki put nakon kopiranja sadržaja sklop promijeni vodeni žig. Nakon što se dosegne postavljeni broj, sklop će odbiti daljnje umnožavanje sadržaja.

2.2.4 Autentifikacija

Korištenjem lomljivih digitalnih vodenih žigova utvrđuje se autentičnost sadržaja dokumenta. Lomljivi vodeni žigovi imaju svojstvo da svaka promjena na dokumentu uzrokuje njihov lom, odnosno kasnije je nemoguće izdvojiti iz dokumenta originalni žig. Ako se izdvojeni žig poklapa sa sadržajem dokumenta, tada je to znak da sadržaj dokumenta nije mijenjan.

Ukoliko se multimedijalni sadržaji koriste u legalne svrhe, npr. medicinske aplikacije ili priopćavanje vijesti važno je osigurati da originalni sadržaj medija nije manipuluran na način da je

izmijenjen ili krivotvoren. Ovo se postiže umetanjem vodenog žiga u sadržaj. Kada se provjerava autentičnost sadržaja, vodeni žig se izvlači iz sadržaja, a cjelovitost sadržaja se dokazuje cjelovitošću žiga. Vodeni žig koji se koristi za autentifikaciju ne smije utjecati na kvalitetu sadržaja i mora biti izrazito otporan na pokušaj uklanjanja.

Neke države koriste ovaj oblik za dodatnu *zaštitu osobnih iskaznica*. Informacije koje se nalaze na osobnoj iskaznici se dodaju kao digitalni vodeni žig na samu sliku. Poslije se izlvačenjem tih informacija može ustvrditi da li slika odgovara osobnoj iskaznici, odnosno dokazati da li je ista krivotvorena.

2.2.5 Steganografija

Odnosno tajna komunikacija. Umetnuti signal vodenog žiga može se iskoristiti i kao nositelj informacije koja se želi sakriti od treće stranke. Skrivanje jedne informacije unutar druge je tipican primjer steganografije. Dok neki digitalni sadržaji sadrže zaglavlja s tzv. metadatom, vodeni žigovi su posebni po tome što je informacija koju želimo sakriti u samom signalu, a ne u zaglavlju ili slično. Treba naglasiti da steganografija nije isto što i kriptografija. Iako obije discipline kriju informacije, steganografija ih pokušava učiniti nevidljivima, dok ih kriptografija pokušava zamaskirati u drugi oblik raznim kodiranjima.

2.3 Problemi primjene vodenih žigova na slike

U analogno doba fotografija je bila prihvaćena kao dokaz događaja. Danas, u digitalnom dobu s brojnim tehnikama kojima se lako manipulira digitalnim fotografija više nije tako. Kako danas za fotografiju, odnosno sliku, bilo u analognom ili digitalnom obliku više nitko ne može garantirati njezinu autentičnost, pojavila se potreba za tehnikama autentifikacije slika.

Tehnike autentifikacije su proučavane u sklopu kriptografije nekoliko destljeća, i one osiguravaju cjelovitost poruke. Na prvi pogled se čini da potreba za tehnikama autentifikacije slika ne predstavlja problem s obzirom da su brojne tehnike poznate u sklopu kriptografije.

Nažalost, situacije je potpuno drugačija. Postoji ogromna količina redudantnih podataka u slikama. Također, postoji i veliki broj različitih prezentacija istog vizualnog sadržaja. Zbog navedenih razloga tehnike autentifikacije slika se suočavaju s jedinstvenim problemima koji nisu obuhvaćeni kriptografijom. Neki od problema su:

- Tehnikama autentifikacije se želi osigurati cjelovitost sadržaja, a ne samo prezentacije istoga. Pretvaranje formata slike iz bmp formata u jpeg format je primjer promjene prezentacije. Želja je osigurati cjelovitost sadržaja bez obzira na oblik prezentacije, odnosno format slike. Uobičajene tehnike kriptografije osiguravaju cjelovitost prezentacije.
- Prilikom autentifikacije sadržaja slike poželjno je osigurati da autentikator bude uklopljen u sliku. Prednosti ovakvog pristupa su da nije potrebno modificirati formate slike za umetanje autentikatora, npr. gif format nema mehanizma za zaglavlje autentifikatora. Druga prednost je ta što bi autentikator preživio transformacije slike iz jednog formata u drugi.
- Prilikom autentifikacije sadržaja slike, uz otkrivanje informacije da li je slika izmijenjena ili ne, poželjno je otkriti i gdje je slika izmijenjena.
- Unatoč velikoj količini podataka unutar jedne slike, svaka tehnike autentifikacije mora biti izvediva programski i sklopovski u realnom vremenu.

Navedeni problemi se mogu riješiti korištenjem lomljivih digitalnih vodenih žigova.

2.4 Lomljivi digitalni vodeni žigovi

Lomljivi digitalni vodeni žig je dizajniran tako da preda binarnu informaciju je li slika mijenjana ili ne, te ako je mijenjana na kojoj lokaciji. Za ilustraciju načela rada opisati će se jedna od mogućih tehnika[5].

Nevidljivi vodeni žig W se umeće u originalnu sliku X veličine m x n. Slika X se dijeli na blokove, kojih je k x l, tako da X_r predstavlja r_{th} blok slike. Vodeni žig W je dvoslojni i dijeli se istim principom, tako da W_r predstavlja r_{th} blok vodenog žiga. Za svaki blok slike X_r, stvara se odgovarajući blok $\tilde{X}_r$, koji je identičan bloku X_r, osim iznimke da je najmanje značajni bit svakog bloka $\tilde{X}_r$ postavljen na nulu.

Za svaki blok X_r se izračunava kriptografska hash funkcija H (K ;m ;n; $\tilde{X}_r$), gdje je K korisnikov ključ. Nad prvih kl bitova koji su izlaz hash funkcije, a koji se tretiraju kao k x l pravokutno polje, je izvršena logička operacija XOR s trenutnim blokom vodenog žiga W_r, te se stvara novi binarni blok C_r.

Svaki element bloka C_r se umeće u najmanje značajan bit odgovarajućeg $\tilde{X}_r$ elementa, čime se stvara izlazni blok X'_0.

Autentifikacija slike se obavlja izvlačenjem bloka C_r iz svakog bloka X'_0 slike zaštićene vodenim žigom. Potom se nad tim blokom obavi logička operacija XOR sa kriptografskom hash funkcijom H(K; m; n; $\tilde{X}'_0$). Tim postupkom, koji je sličan prethodno opisanom, se dobiva izvučeni blok vodenog žiga.

Promjene na slici zaštićenoj digitalnim vodeni žigom odgovaraju promjenama na odgovarajućim blokovima vodenog žiga čime se može odrediti koja lokacija slike je izmijenjena.

Glavni problemi pri dizajniranju digitalnih vodenih lomljivih žigova su:

- *Lokalnost* Koliko dobro tehnika implementacije žiga može identificirati piksel koji je modificiran? Primjer prethodno opisane tehnike radi blokove od 12 x 12 piksela, i svaka manja promjena unutar bloka za rezultat daje samo blok u kojem je napravljena izmjena, ali ne točnu lokaciju samog izmijenjenog piksela.
- *Transparentnost* Umetanje vodenog žiga uzrokuje narušavanje kvalitete same slike. Digitalni žig bi trebao što manje narušiti izgled slike.
- *Sigurnost* Koliko je teško nekome tko ne zna samo korisnikov ključ K u implementaciji vodenog žiga, izmijeniti sliku bez mijenjanja žiga? Odnosno, koliko je teško unijeti novi, ali ispravan vodeni žig?

2.4.1 Slabo lomljivi digitalni vodeni žigovi

Obični digitalni vodeni žigovi su dizajnirani tako da se pri svakoj promjeni slike slome. Odnosno, nisu otporni ni na pokušaje prilagođavanja slike za potrebe korisnika bez napada na autentičnost. Npr. prilagođavanje veličine slike stranici novina. Zbog navedenog razloga se razvijaju slabo lomljivi digitalni vodeni žigovi koji bi bili otporni na 'dobroćudne' napade, kao što su prilagođavanje slike potrebi, ali bi svejedno bili otporni na napade na autentikaciju slike, odnosno, na 'zloćudne' napade.

Prvi problem s kojim se suočava je točno odrediti što su to 'dobroćudni' napadi i gdje je granica. Spomenuto prilagođavanje slike za ispis u novinama može promijeniti sliku iz boje u crno bijelu, povećati kontrast, izoštriti je, a sve bez mijenjanja samog sadržaja iste slike. Pristup razradi

funkcije za dodjeljivanje digitalnog vodenog žiga je takav, da bi ona trebala pamtiti položaje ključnih objekata slike, odnosno svaka ključna točka slike bi dobila svoj otisak prsta.

Iznimno je teško napraviti funkciju koja će znati točnu granicu sadržaja slike i dozvoljenih operacija koje iako ne mijenjaju sadržaj slike, mijenjaju vodeni žig. Slabo lomljivi digitalni vodeni žigovi imaju potencijalnu puno veću primjenu.

3. Stohastička difuzijska pretraga

3.1 Uvod

SDS je prvi puta opisao Dr. John Mark Bishop 1989. godine. SDS je efektivna generička metoda pretraživanja, originalno razvijena kao populacijski orijentiran algoritam koji traži odgovarajuće uzorke. Algoritam je dio Inteligencije Roja (eng. Swarm Intelligence) i algoritama optimizacije i pretraživanja, inspiriranih socijalnim insektima (pčele i mravi), kao što su optimizacija kolonijom mrava *(eng. Ant Colony Optimization,* u daljnjem tekstu ACO), optimizacija partikularnim rojem (eng. *Parcticle Swarm Optimization,* u daljnjem tekstu PSO) i genetskim algoritmima. Zadnjih godina je iskazana velika zainteresiranost za distribuiranim izračunavanjem korištenjem interakcije između agenata.

ACO je zasnovan na komunikaciji koja se zasniva na izmjenama fizičkih osobina simulirane okoline (mravi koji koriste feromone u svojoj okolini). Za takav oblik indirektne komunikacije rabi se izraz stigmetrička komunikacija (eng. *Stigmetric Communication*). SDS koristi oblik direktne (jedan na jedan) komunikacije između agenata, slične mehanizmu izravne komunikacije prisutne kod jedne vrste mrava, *Leptothorax acervorum.*

SDS koristi populaciju agenata gdje svaki ima hipotezu o mogućem rješenju i evaluira je parcijalno. Agenti izvode jeftinu, djelomičnu procjenu hipoteze (kandidata za rješenja problema pretrage). Potom dijele informacije o hipotezi (difuzija informacija) direktnom jedan na jedan komunikacijom. Kao rezultat difuzijskog mehanizma, veoma kvalitetna rješenja mogu se identificirati pomoću skupine agenata s istom hipotezom. Rad SDS-a je najjednostavnije prikazati Igrom restorana.

SDS je najučinkovitiji za probleme optimizacije gdje se objektna funkcija može rastaviti na dijelove koji se mogu ocijeniti samostalno. Da bi pronašao optimum dane objektne funkcije SDS upošljava roj od n agenata, gdje svaki održava hipotezu o optimumu. SDS algoritam nameće ponavljanje faza TEST (ispitaj) i DIFFUSION (podijeli) dok roj agenata ne konvergira optimalnoj hipotezi.

3.1.1 Igra restorana

Grupa izaslanika je nazočna na dugoj konferenciju u njima nepoznatom gradu. Svaku večer moraju pronaći gdje će večerati. Postoji veliki odabir restorana, svaki s velikim odabirom jela. Problem s kojim se grupa suočava je pronalaženje najboljeg restorana gdje će najveći broj delegata uživati u večeri. Paralelna pretraga restorana i kombinacija jela bi bila iscrpljujuća te, najvažnije, preduga. U rješavanju problema delegati odlučuju primijeniti SDS. [1]

Svaki izaslanik se ponaša kao agent održavajući hipotezu najboljeg restorana u gradu. Svake noći svaki izaslanik testira svoju hipotezu večerajući u svom, trenutno najboljem restoranu neko nasumce odabrano jelo. Iduće jutro, za doručkom svaki izaslanik koji nije uživao u jelu prethodne večeri, pita jednog odabranog kolegu kakva je bila njegova večera. Ukoliko je iskustvo pozitivno i on prihvaća taj restoran za svoj omiljeni, odnosno najbolji. U suprotnom jednostavno izabere nasumce restoran iz telefonskog imenika. Korištenjem ove strategije uočeno je da se vrlo brzo značajan broj izaslanika skupi oko najboljeg restorana u gradu. Taj proces se može opisati idućim algoritmom prikazanim na slici Slika 3-1.

FAZA INICIJALIZACIJE

GDJE SVI AGENTI (IZASLANICI) GENERIRAJU

INICIJALNU HIPOTEZU (RESTORAN)

PETLJA

FAZA ISPITIVANJA

SVAKI AGENAT EVAULIRA DOKAZE SVOJE HIPOTEZE (DEGUSTACIJA OBROKA). AGENTI SE DIJELE U AKTIVNE (ONI KOJI SU ZADOVOLJNI VEČEROM) I NEAKTIVNE (ONI KOJI NISU).

FAZA DIFUZIJE

NEAKTIVNI AGENTI USVAJAJU NOVE HIPOTEZE KOMUNIKACIJOM S DRUGIM AGENTOM, ILI AKO JE AGENT KOJEM SU SE OBRATILI ZA POMOĆ TAKOĐER NEAKTIVAN, NEMA RAZMJENE INFORMACIJA, PA NASUMICE USVAJAJU NOVU HIPOTEZU

Slika 3-1 Pseudokod SDS-a

Iteraciom kroz faze testiranja i difuzije agenti stohastički istražuju cijeli prostor rješenja. Agenti provode više vremena oko dobrih rješenja. Pošto svaki agent provodi vrijeme istražujući dobro rješenje, zapošljavajući nove agente, koji zapošljavaju još više novih, prostor s lošim rješenjima se jako malo istražuje. Kandidati za rješenje se identificiraju po velikom broju agenata koji će se okupiti oko njih.

Jaka strana SDS je njegova mogućnost da pobjegne iz lokalnih optimuma. To je postignuto vjerojatnosnim ishodom parcijalne evaluacije hipoteze u kombinaciji s relokacijom agenata putem stohastičkih mehanizama aktivacije, odnosno novačenja. Parcijalna evaluacija hipoteze dozvoljava agentu brzo formiranje svog mišljenja o hipotezi koju trenutno istražuje bez dugotrajnog testiranja (npr. u igri restorana, agenti mogu pronaći najbolji restoran u gradu bez da probaju jelo u svakom restoranu).

3.1.2 Terminologija

U originalnoj definiciji SDS-a populacija agenata traži najbolje rješenje za problem optimizacije. Skup svih mogućih rješenja problema formiraju prostor rješenja *S*. Svaka točka prostora *S* ima pridruženu objektnu vrijednost. Objektne vrijednosti iz cijelog prostora rješenja S formiraju objektnu funkciju *f*. Zbog pojednostavljenja, pretpostavlja se da je problem minimalizirati sumu od $n\{0,1\}$ - vrijednosti komponente funkcije f_i, koja može biti deterministička:

$$\min_{\forall s \in S} f(s) = \min_{\forall s \in S} \sum_{i=1}^{n} f_i(s) \qquad f_i : S \to \{0,1\} \tag{3.1}$$

Veliki broj problema optimizacije nakon transformacije se rješavaju jednadžbom (3.1), unatoč činjenici da ona postavlja veliko ograničenje. Primjer takvog ograničenja biti će prikazan u poglavlju Primjeri primjene 3.3. Tijekom rješavanja problema, svaki agent održava hipotezu o najboljem rješenju, odnosno hipotezu koja je kandidat za rješenje, ili označava vrijednost, odnosno točku u prostoru rješenja.

3.2 Algoritam

Originalno agenti u SDS-u rade paralelno i sinkronizirani su. Prolaze kroz razne faze, koje se mogu sažeti u idući pseudokod prikazan slikom Slika 3-2:

```
INITIALISE (agents);

REPEAT

        TEST (agents);
        DIFFUSE (agents);

UNTIL (halting criterion);
```

Slika 3-2 Pseudokod agenata SDS-a

INITIALISE: (Inicijaliziraj) Tipično se početna hipoteza svakog agenta bira podjednako nasumce kroz prostor pretrage. Dakako, svaka informacija o vjerojatnom rješenju problema dostupna apriori može biti korištena u postavljanju inicijalne hipoteze. Postoji puno metoda postavljanja hipoteza, ali njihove detaljno poznavanje nije potrebno za razumijevanje samog algoritma.

TEST: (Ispitaj) Svaki agent nasumice izabire jednu komponentu funkcije f_i, $i \in \{1,...,n\}$, i evaluira partikularnu hipotezu $s_h \in S$. Ovisno o ishodu hipoteze, agenti se dijele u dvije skupine: aktivne i neaktivne. Za aktivne agente vrijedi $f_i(s_h) = 0$, a za neaktivne agente vrijedi $f_i(s_h) = 1$. Fazu ispitivanja moguće je opisati idućim pseudokodom prikazanim na slici Slika 3-3

```
FOR AGENT = 1 TO (ALL AGENTS)
        KOMPONETNTA FUNCKIJE = ODABERI-NASUMICE-KOMPONENTU-FUNKCIJE()
        IF( KF( AGENT.HIPOTEZA) == 0) AGENT.AKTIVNOST = ISTINA;
                ELSE
                AGENT.AKTIVNOST = NEISTINA;
                END
END
```

Slika 3-3 Pseudokod faze ispitivanja

Booleanova testna funkcija vraća ISTINA ako je nasumce odabrana djelomična procjena objektne funkcije pokazatelj dobre hipoteze.

DIFFUSE: (Difuzija, Razlaganje) Tijekom ove faze, svaki neaktivni agent nasumice odabire agenta za komunikaciju. Ukoliko je odabrani agenta aktivan, onda prvi agent preuzima njegovu hipotezu: difuzija informacija. Ukoliko je odabrani agenta neaktivan, nema hipoteze, nema razmjene podataka, pa prvi agent nasumice odabire neku hipotezu. U standardnom SDS-u aktivni agenti ne započinju komunikaciju, već samo neaktivni. Ova faza se može opisati idućim pseudokodom prikazanim na slici Slika 3-4:

```
FOR AGENT 1 TO (ALL AGENTS)
        IF (AGENT.AKTIVNOST == NEISTINA)
                AGENT2 = ODABERI-NASUMICE-AGENTA(AGENTI);
                IF (AGENT2.AKTIVNOST == ISTINA)
                        AGENT.HIPOTEZA = AGENT2.HIPOTEZA
                        ELSE
                        AGENT.HIPOTEZA = ODABERI-NASUMICE-HIPOTEZU();
                        END
                END
END
```

Slika 3-4 Pseudokod faze difuzije

KRITERIJ ZAUSTAVLJANJA (Halting criterion) Postoji puno kriterija zaustavljanja. Njihovo detaljno poznavanje također nije potrebno za razumijevanje algoritma. Za otkrivanje hipoteze koriste se dva kriterija zaustavljanja: a) jaki kriterij zaustavljanja (*Strong Halting Criterion)* koji, nakon što 'shvati' da je najveća nakupina agenata premašila minimalan prag, provjerava (stohastički) da li je veličina nakupine stabilna u određenom broju ponavljanja, te b) slabi kriterij zaustavljanja (*Weak Haltering Criterion)* koji provjerava stabilnost i minimalan broj aktivnih agenata (ukupna aktivnost uvelike ovisi o najboljem rješenju pronađenom dosad).

3.2.1 Od operacije agenta do ponašanja populacije

Algoritamski opis operacija agenata nije dovoljan da se u potpunosti razumije kako SDS rješava probleme optimizacije. Potrebno je razmotriti što se dešava sa populacijom kao cjelinom. Iteraciom faza ispitivanja i razlaganja agenti kao jedinke istražuju cijeli prostor rješenja. Kako faza testiranja češće uspijeva u prostoru s dobrim objektnim vrijednostima, agenti će u prosjeku više vremena provoditi istražujući kvalitetna rješenja, istodobno privlačeći druge agente, koji će pak povlačiti još agenata. To je mehanizam koji uzrokuje formiranje dinamičkih, ali stabilnih nakupina agenata u određenom prostoru rješenja. Ograničenost broja agenata osigurava da samo najbolje rješenje otkriveno do tog trenutka održi stabilan broj agenata u nakupini. Upravo ova neproporcijalna uporaba resursa (broja agenata) na kraju dozvoljava da optimalno rješenje bude pronađeno najvećom nakupinom agenata, bez da svaki agent pretraži cijeli prostor rješenja.

3.2.2 Manipuliranje procesima distribucije resursa

Naglasak na lokalno istraživanje

Standardni SDS nema mehanizme za istraživanje sličnosti u samoj objektnoj funkciji. To znači da prostor rješenja često može imati jako slične objektne vrijednosti. Mehanizam uvođenja malih varijacija na različitosti hipoteza se lako može uvesti u postojeći algoritam, što je već učinjeno u rojevima. Jedna mogućnost je narušavanje kopiranja parametara hipoteze uvođenjem nasumičnog pomaka prilikom kopiranja hipoteze za vrijeme faze difuzije. Nešto poput mutacije kod evolucijskih algoritama. Učinak je raspršenje agenata po susjednim lokacijama, umjesto da se samo okupljaju u nakupine oko jedne vrijednosti. To omogućava poboljšano vrijeme konvergencije u prostoru sa sličnostima objektne funkcije i praćenje pokretnih maksimuma dinamičkih problema.

Naglasak na globalno istraživanje

Spor zahtjeva za pretraživanjem većine prostora rješenja, pogotovo u dinamičkim okruženjima, i potrebe za stabilnim nakupinama koje istražuju trenutno najbolje rješenje ne daje uvijek optimalna rješenja prilikom rabljenja standardnog SDS-a. Njegova distribucija resursa je pohlepna. Nakon što je dobro rješenje otkriveno veliki dio roja će se distribuirati u njegovo istraživanje, što će omogućiti tim agentima da istražuju dalje. Potreban je mehanizam koji će osloboditi neke od tih agenata, a da neće uvelike narušiti stabilnost agenata u nakupini dobrog rješenja. Takav mehanizam bi povećao učinak SDS-a u velikom broju problema, posebno prilikom dinamičke optimizacije. Jedan takav mehanizam je kontekstno-osjetljiv SDS. Njegova razlika od standardnog SDS-a je u fazi razlaganja za aktivne agente. Aktivni agenti standardnog SDS-a cijelo vrijeme održavaju svoju trenutnu hipotezu. Kontekstno-ovisan SDS svakom aktivnom agentu odabire drugog aktivnog agenta za komunikaciju. Ukoliko oba podupiru isto hipotezu, onda prvi agent postaje neaktivan i odabire nasumice novu hipotezu iz prostora rješenja. Mehanizam regulira samog sebe protiv stvaranja ogromnih nakupina agenata, jer što je više agenata s istim hipotezama, veća je vjerojatnost da će upravo oni međusobno komunicirati. Time je uveden mehanizam

negativne selekcije u originalni algoritam koji omogućuje SDS-u da održi relativno velike nakupine na višestruko sličnim, skoro optimalnim rješenjima.

3.2.3 Paralelnost SDS-a

SDS je sam po sebi paralelan, no praktično paralelno izvođenje SDS-a je problematično zbog a) potrebe svakog agenta da komunicira s drugima i b) količine komunikacije između agenata. Moguće rješenje je organizacija agenata u rešetkastu strukturu s direktnom komunikacijom sa samo *k* najbližih agenata. Drugo rješenje je podjela roja agenata u manje rojeve, gdje se svaki roj izvodi na zasebnom procesoru ali sa potpunom povezanošću uz nisko frekventnu komunikaciju među rojevima.

3.3 Primjeri primjene

3.3.1 Usklađivanje poravnavanjem manje fotografije u veću

Problem je uskladiti poravnavanjem manju fotografiju, koja je slikana iz blago drugačijeg kuta gledanja, sa velikom fotografijom. Treba napomenuti da je primjer odabran da pokaže princip rada SDS, a ne kao rješenje specifičnog problema. Primjer je prikazan na slici Slika 3-5.

Slika 3-5 Fotografija za razradu problema

Problem se sastoji od prepoznavanja manje fotografije unutar veće, tako što će se naći koordinate (x,y) koje će dati najbolju podudarnost male fotografije i dijela velike fotografije. Veličina velike slike je 300 sa 860 piksela, dok je veličina male slike 30 sa 40 piksela. Slika je u RGB formatu, odnosno svaki piksel ima tri vrijednosti intenziteta boje. Prostor pretrage je diskretan.

Za utvrđivanje podudaranja između dvaju fotografija koristi se matematička funkcija za mjerenje udaljenosti između dvaju točaka, samo što funkcija uspoređuje vrijednosti intenziteta crvene, zelene i plave boje:

$$f(x,y)=\sum_{k,l}(|r_{kl}-R_{kl}(x,y)|+|g_{kl}-G_{kl}(x,y)|+|b_{kl}-B_{kl}(x,y)|) \tag{3.2}$$

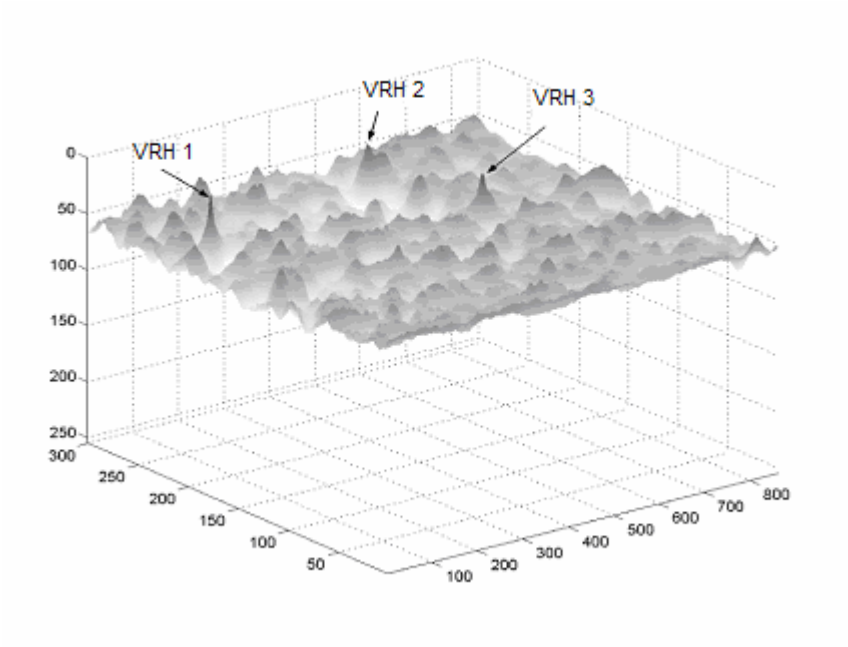
gdje r_{kl} označava vrijednost intenziteta crvene boje piksela (k,l) male fotografije. R_{kl} označava vrijednost intenziteta crvene boje piksela (x+k, y+l) u velikoj fotografiji. Problem podudaranja se svodi na rješavanja problema:

$$\min_{x,y} f(x,y) \tag{3.3}$$

Treba uočiti da se objektna funkcija (3.2) može razložiti u više komponentnih funkcija, koje mogu neovisno odrađivati svoj dio procjene (npr. samo za zelenu boju). Prostor rješenja *S* je dvodimenzionalan i diskretan gdje su $x\in\{1,2,3,...,860\}$, a $y\in\{1,2,3,...,300\}$. Veličina prostora rješenja je $860*300=258000$. Broj komponentnih funkcija f_i je utvrđen uvjetima zbrajanja jednadžbe (3.2) i veličinom male fotografije te različitim vrijednostima intenziteta boja $30*40*3=3600$. Komponente funkcije f_i su diskretne s cjelim brojem u vrijednosti [0,255]. Minimizacijski problem (3.3) se svodi na (3.1). Za komponentu *i* i hipotezu rješenja (x,y), testna procedura računa kvantitetu:

$$t_i(x,y)=\frac{f_i(x,y)}{255} \tag{3.4}$$

Ispitna procedura daje vrijednost 0 s vjerojatnošću $t_i(x,y)$, dok daje vrijednost 1 sa vjerojatnošću $1-t_i(x,y)$. Ova procedura osigurava da transformacija objektnih vrijednosti osigurava očuvanje redoslijeda tih vrijednosti, što je dovoljan uvjet za točnu optimizaciju objektne funkcije. Rješenje za dani primjer je prikazano na slici Slika 3-6.



Slika 3-6 Prostor rješenja problema poravnavanja manje fotografije u veću

Vrh 1 je rješenje, dok su vrhovi 2 i 3 bili kandidati za rješenje, no zbog niže kvalitete od vrha 1 nisu izabrani.

3.3.2 Usporedba s drugim algoritmima

Iako postoje dobro definirani problemi pretrage za uspoređivanje algoritama kao što je DeJongov test, takav eksperiment za prikaz efikasnosti za korištenje SDS-a nije prikazan zbog nedovoljno definiranih uvjeta težine same pretrage. U ovom odjeljku biti će prikazana empirijska ocjena težine pretrage uspoređujući SDS s nekoliko drugih čestih optimizacijskih algoritama: nasumična pretraga (*random search*), metoda penjanja na brdo (*hill climber)*, i optimizacija rojem čestica (*particle swarm optimization*, u daljnjem tekstu PSO). Algoritmi se uspoređuju na problemu opisanom u prethodnom odjeljku 3.3.1.

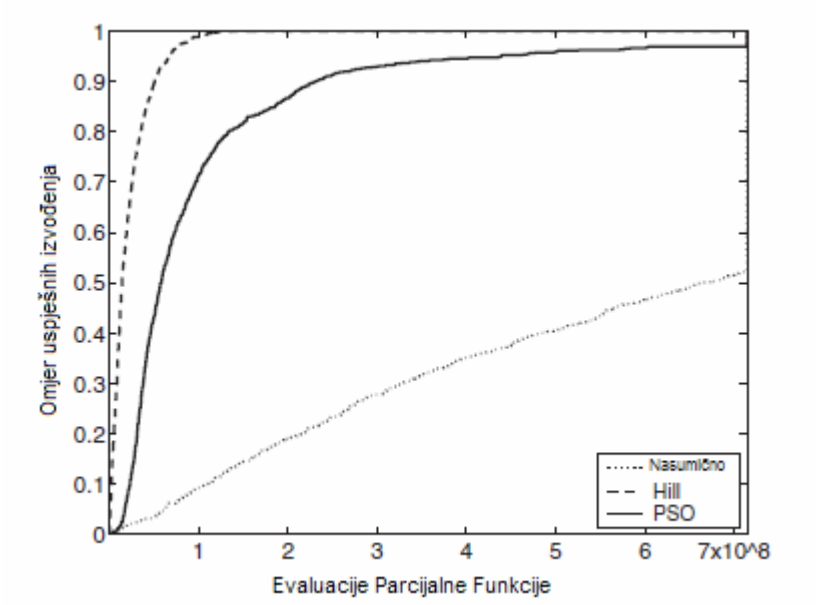
Nasumična pretraga odabire nasumično rješenje i evaluira ga dok ne pronađe Vrh 1 na slici Slika 3-6.

Metoda penjanja na brdo odabire rješenje nasumično i evaluira ga. U idućim iteracijama 8 susjednih rješenja je evaluirano. Pretraga se pomiče prema najboljem poboljšanju u objektivnoj vrijednosti. Ukoliko takav pomak nije moguć, pretraga je pronašla lokalni optimum, te se ponovno pokreće u novom, nasumično odabranom području. Proces se ponavlja dok se ne otkrije optimalno rješenje, odnosno Vrh 1.

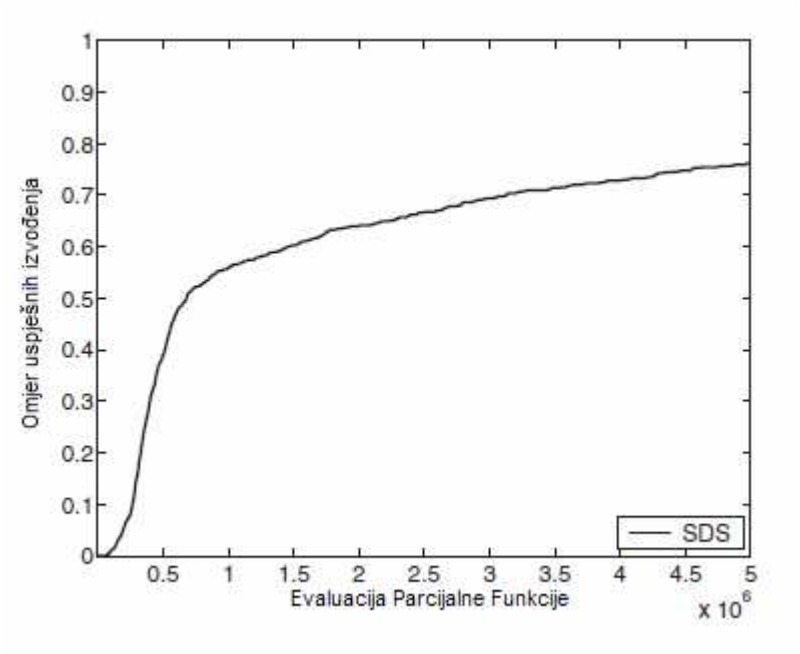
PSO Ovaj algoritam slijedi inačicu PSO ograničenog na lokalne optimume [3]. Algoritam se izvodi dok se optimalno rješenje Vrh 1 ne pronađe s barem jednom česticom, odnosno agentom. Korišteni su sljedeći parametri: koeficijent ograničenja $X = 0.179$, kognitivni i socijalni parametri $c_1 = c_2 = 2.05$, 200 čestica s radijusom susjedstva 1, $\max V_x = 100$ i $\max V_y = 200$. Parametri su odabrani u svrhu davanja optimalnog PSO algoritma pretrage. Za detalje implementacije detalje potražiti u [3].

SDS Korišten je standardni SDS algoritam s 1000 agenata. Mali mutacijski mehanizam opisan u poglavlju 3.2.2 je korišten koji dodaje male pomake (x,y) koordinatama gdje je vjerojatnost mutacije 8%. Pretraga konvergira optimalnom rješenju, tj. Vrhu 1 kad se skupi trećina svih agenata oko njega.

Rezultati izvođenja pokusa su prikazani idućim slikama.



Slika 3-7 Usporedba izvođenja Nasumične pretrage, Metode penjanja uz brdo i PSO. Prikaz za 1000 izvođenja



Slika 3-8 Prikaz rezultata 1000 izvođenja algoritma SDS

Slike *Slika 3-7* i Slika 3-8 prikazuju ponašanje četiri gore navedena algoritma prilikom pretrage. Slike prikazuju distribuirano gomilanje ukupnog broja evaluacija parcijalnih funkcija navedenih algoritama. Učinkovitost parcijalne evaluacije može se prikazati usporedbom skupoće, odnosno uloženih resursa, SDS-a i ostalih algoritama.

Nasumična pretraga treba oko 191 000 evaluacija za uspješnost od 50 % u pronalasku globalnog optimuma. To znači da je potrebno 191 000 * 3 600 = 687,6 miliona evaluacija komponente funkcije f_i. SDS će napraviti oko 683 000 komponentnih evaluacija. PSO zahtijeva 16 000

potpunih evaluacija funkcija, a Metoda penjanja uz brdo 4 000.

Zanimljivo je pogledati, ne uspoređujući brojeve, sa koliko komponentnih funkcija f_i SDS nadalje nadmašuje druge algoritme. Vjerojatnostni parcijalni evaluacijski mehanizam SDS-a pretvara pretragu u stohastički dinamički proces neovisan o broju komponentnih funkcija u objektnoj funkciji. One ovise samo o obliku prostora pretrage. Drugim riječima, bez obzira da li za prostor pretrage treba 100, 1000 ili 100 tisuća komponentnih funkcija, prosječno ponašanje pretrage SDS-a je uvijek isto. Konkretno za prostor pretrage prikazan na slici Slika 3-5, SDS bi nadmašio nasumičnu pretragu za objektne funkcije koje se sastoje od 683 000 / 191 000 ≈ 4 ili više komponentnih funkcija. Za PSO se dobije 683 000 / 16 000 ≈ 43, a za Metodu penjanja uz brdo 683 000 / 4 000 ≈ 171. Relativno loši rezultati PSO-a naspram Metode penjanja uz brdo mogu se objasniti da se roj sastoji od 200 jedinki, odnosno čestica gdje svaka izvodi potpunu parcijalnu evaluaciju. Vjerojatno bi se učinkovitost PSO-a poboljšala ukoliko bi se povećale dimenzije problema.

3.4 SDS i evolucijski algoritmi

SDS algoritam opisan perspektivom agenata koji iteriraju faze ispitivanja i razlaganja može se učiniti da je daleko, ili da uopće nije evolucijski algoritam. Iako nije originalno potekao iz metafora biološke evolucije, nego promatrajući neuronske mreže, SDS se uklapa u algoritme inspirirane Darwinovom evolucijom izvedenih osnovnim procesima varijacije, selekcije i replikacije. SDS se uklapa iz perspektive hipoteze. SDS čini nasumične odabire i male preinake hipoteza prilikom njihovog kopiranje, kao što evolucijski algoritmi rade uvođenje nasumičnih imigranata i mutacije. Odbacivanje hipoteze je njezina 'smrt', dok je kopiranje hipoteze svojevrsna reprodukcija. Dobre hipoteze imaju veću šansu preživljavanja faze testiranja, te istovremeno se šire na veći broj agenata. Zbog ograničenog broja agenata, samo određeni broj agenata može održavati hipotezu, te kroz međusobnu komunikaciju agenata ostvaruje se selekcija, iako nema klasične selekcije. Zbog kontinuirane i indirektne evaluacije svake hipoteze, SDS simulira evolucijske procese u nešto drugačijem vremenski definiranom prostoru, za razliku klasične linearne evolucije.

3.5 Usporedba SDS i algoritama socijalnih insekta

Nasuprot stigmetričnoj komunikaciji korištenoj u većini algoritama mrava, SDS koristi jedan na jedan sistem novačenja po uzoru na ponašanje određenih vrsta mrava koji ne ostavljaju feromone, nego idu jedan za drugim. Iz veze s SDS-om se tvrdi da se efikasna i globalna odluka može postići samo interakcijom i međusobnom komunikacijom među populacijom, gdje svaka jedinka održava svoju hipotezu i parcijalno je evaluira. Procesi novačenja novih agenata su mnogo kompleksniji, s obzirom na to da agent u SDS-u ne mora fizički otići do drugog agenta (mrava), te izvoditi rutine ponašanja, odnosno 'plesa' kako bi mu prenjeo svoju hipotezu i unovačio ga. Nigdje u prirodi ne postoji sistem novačenju orijentiran na aktivne i neaktivne agente. Sistem je preuzet od određenih vrsta mrava i njihovog ponašanja prilikom osnivanja novog mravinjaka. Promatrajući ih, došlo se do zaključka da ti mravi trebaju veću razinu individualnih kognitivnih sposobnosti, kao što je usporedba dva pogodna mjesta za podizanje novog mravinjaka, da bi došli do optimalnog rješenja, što je suprotno stigmetričnoj komunikaciji većine mrava. Unatoč tome, osnovne sličnosti SDS-a i socijalnih insekta sugerira da globalna i čvrsta odluka u obadva sistema proizlazi iz kooperacije povezanih agenata, gdje svaki od njih individualno ne bi bio sposoban riješiti problem u približno istom vremenu.

4. Projektiranje SDS sustava

4.1 Program za uklapanje slika

4.1.1 Ideja

Osnovni cilj ostvarenog programa je pretraga, i uklapanje male slike u veliku, kao što je to opisano u poglavlju 3.3.1. Treba naglasiti da je poglavlju 3.3.1 predviđeno da je mala slika slikana iz blago različitog kuta. U ostvarenju ovog programa je predviđeno da je mala slika dio velike, odnosno njezina potslika. Ulaz programa su dvije slike u bmp formatu. Izlaz je u tekstualnom obliku. Ime programa je Stochastic Diffusion Search.

4.1.2 Ulazi i izlazi programa

Kao ulazi programa zamišljene su dvije slike u bmp formatu. Predviđeno je da prva slika bude veličine 500*300 piksela, a mala slika 50*30 piksela. Program radi i za slike drugih veličina, koje moraju biti manje od zadanih dimenzija, iako je izvorno predviđen za zadane dimenzije.

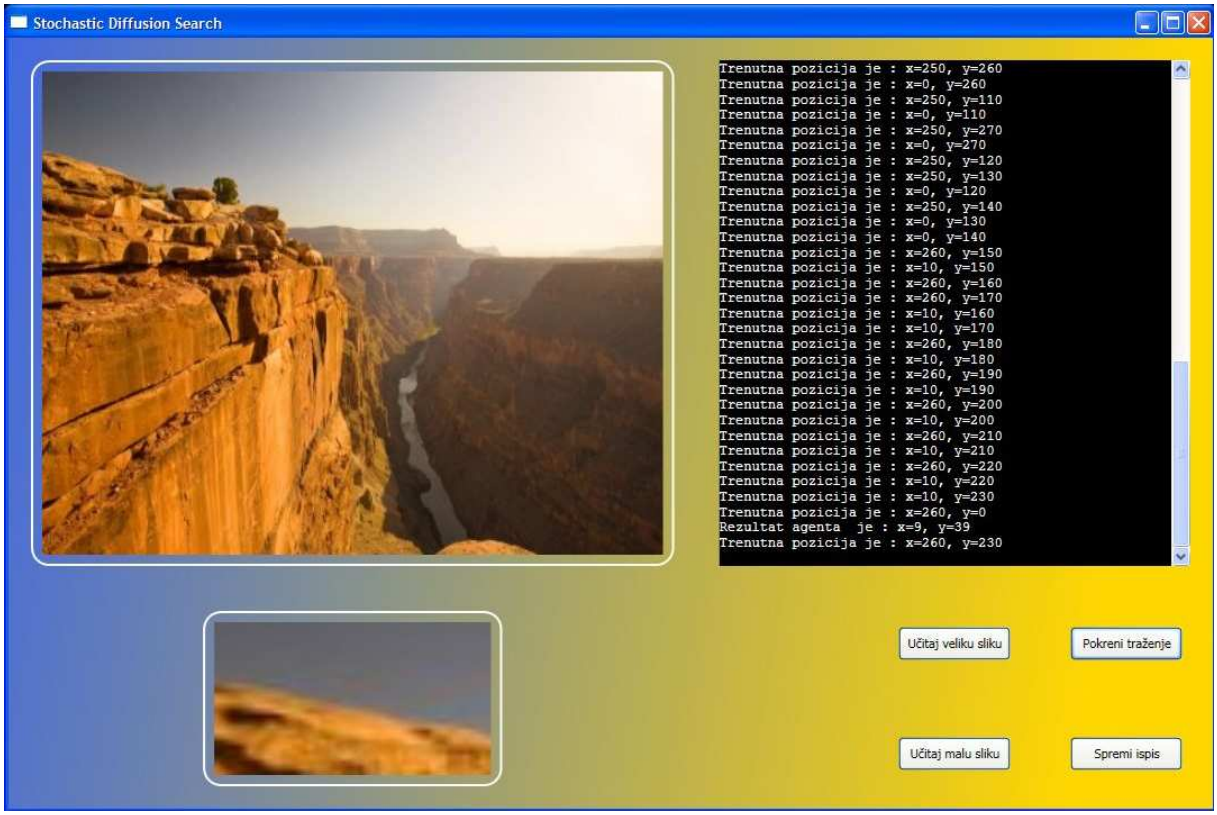
Izlazi programa su na desnoj strani glavnog prozora programa i ispisuju se u obliku koordinata na kojima se agenti trenutno nalaze. Rezultate je moguće spremiti u obliku txt datoteke.

4.1.3 Realizacija

Program je ostvaren programskim jezikom C# u razvojnom okruženju Microsoft Visual C# 2008 Express Edition. Za pokretanje programa nisu potrebni nikakvi dodatci, iako računalo mora imati instaliran .NET framework 3.0 ili 3.5.

Agenti SDS-a kao algoritma su programski ostvareni kao četiri dretve gdje svaka pretražuje jedan kvadrant slike.

Program se pokreće kao posebna Win32 aplikacija.



Slika 4-1: Aplikacijski prozor programa Stochastic Diffusion Search

Korisnik učita slike u program i nakon toga glavni program pokreće 4 neovisne dretve, od kojih svaka pretražuje svoj kvadrant (kao kvadranti u koordinacijskom sustavu).

Agenti rade tako da kreću od početka svog kvadranta, odnosno u gornjem lijevom kutu svog kvadranta. Te uzimaju vrijednost boje tog piksela koju dobivaju kao cjelobrojnu vrijednost. Bitno je napomenuti da ovdje program ne razlaže vrijednosti boja u RGB format kao što je to bio slučaj u poglavlju 3.3.1, već gledaju boju u jednom formatu.

Ukoliko se prva vrijednost boja piksela slažu, agent prelazi na drugi piksel male i velike slike, sve dok ne nađe potpuno preklapanje. Ukoliko nađe poklapanje ispiše rezultat.

Dretve, odnosno agenti, će svakih deset piksela ispisati gdje se nalaze i da li su pronašle rješenje.

4.1.4 Upute za korištenje

Nakon što korisnik pokrene aplikaciju otvara se prozor kao što je to prikazano na slici Slika 4-1. Lijevi dio prozora je rezerviran za prikaz velike i male slike, dok se desno ispisuju rezultati. Na dnu desnog kuta se nalaze četiri gumba za kontroliranje programa.

Prvi gumb je 'Učitaj veliku sliku'. Nakon njegovog pritiskanja korisniku se otvara novi prozor u kojem treba učitati veliku sliku. Slika mora biti bmp formata (inače program neće raditi) i korisnik mora pripaziti na dimenzije kako je već opisano.

Drugi gumb je 'Učitaj malu sliku' i procedura je ista kao za veliku sliku, no ovaj put korisnik treba učitati malu sliku.

Treći gumb je 'Pokreni traženje' čijim se pritiskom pokreće pretraga, odnosno stvaraju se i aktiviraju agenti. Program će dojaviti korisniku novim prozorom kad je gotov s pretragom.

Četvrti gumb je 'Spremi ispis' kojim se svi rezultati koji su prikazani u prozoru rezultata mogu spremiti u obliku txt datoteke. Korisniku će se otvoriti novi prozor gdje može odabrati ime datoteke i njezinu lokaciju. Ukoliko prilikom čitanja dobivene txt datoteke ispis bude nečitak, ili slova budu gusto naslagana, preporuča se otvaranje dobivene txt s Microsoft Wordom koji će od korisnika zatražiti da odabere dekodiranje. Potrebno je odabrati *Other encoding, Unicode* (UTF 8). Ukoliko otvorite dobivenu txt datoteku s programom Notepad++ on će automatski prepoznati format, i pravilno prikazati tekst.

Za ponovnu pretragu, program se mora ugasiti i ponovno pokrenuti. Nije dovoljno samo učitati nove slike i stisnuti gumb 'Pokreni traženje'.

4.2 Program za detekciju promjene slika

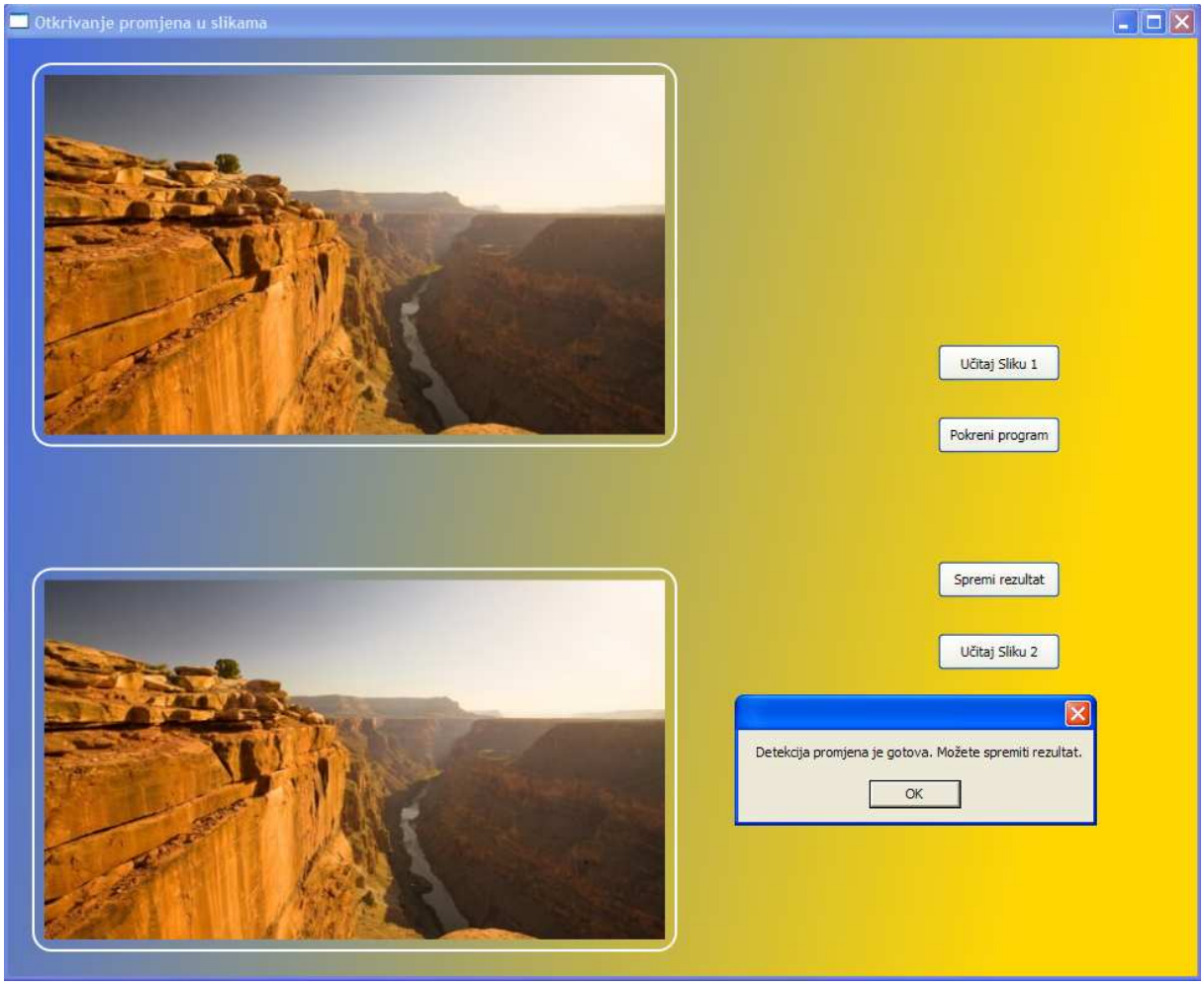
4.2.1 Ideja

Program je ostvaren na osnovu SDS programa uz određene modifikacije. Program sad uspoređuje dvije slike i daje za rezultat lokacije promjena.

4.2.2 Ulazi i izlazi iz programa

Kao ulazi programa zamišljene su dvije slike u bmp formatu. Predviđeno je da slike budu veličine 500*300 piksela. Program radi i za slike drugih veličina, koje moraju biti manje od zadanih dimenzija.

Izlaz programa je nova slika, također u bmp formatu na kojoj su uočljivom zelenom bojom naznačeni pikseli koji se razlikuju za dvije zadane ulazne slike.



Slika 4-2 Aplikacijski prozor programa za otkrivanje promjena u slikama

4.2.3 Upute za korištenje

Nakon što korisnik pokrene aplikaciju otvara se prozor kao što je to prikazano na slici Slika 4-2. Lijevi dio prozora je rezerviran za prikaz slika, dok se desno nalaze četiri gumba za kontroliranje programa.

Prvi gumb je 'Učitaj Sliku 1'. Nakon njegovog pritiskanja korisniku se otvara novi prozor u kojem treba učitati prvu sliku. Slika mora biti bmp formata (inače program neće raditi) i korisnik mora pripaziti na dimenzije kako je već opisano.

Drugi gumb je 'Učitaj Sliku 2' i procedura je ista kao za Sliku 1.

Treći gumb je 'Pokreni traženje' čijim se pritiskom pokreće pretraga, odnosno stvaraju se i aktiviraju agenti. Program će dojaviti korisniku novim prozorom kad je gotov s pretragom.

Četvrti gumb je 'Spremi rezultat' kojim se, nakon što program dojavi da je pretraga završena, može spremiti dobiveni rezultat u obliku nove bmp slike. Korisniku se otvara prozor u kojem može odabrati lokacije gdje će spremiti sliku rezultat i odabrati njezino ime.

5. Eksperimenti

5.1 Ispitivanje SDS programa

Program je testiran s proizvoljnim slikama koje su preporučenih dimenzija iz prethodnog poglavlja.

Velika slika za testiranje je prikazana na slici Slika 5-1. Dok je mala slika zbog lakšeg uočavanja istaknuta unutar veće.

Slika 5-1 Velika slika za testiranje programa

Nakon izvođenja programa, i spremanja ispisa dobiva se sljedeći rezultat:

```
Inicijalizacija agenata (dretve
pretrazivaci).
Rezultat početni, rješenje još ne
postoji : x=-1, y=-1
Agent je aktiviran.
Agent je aktiviran.
Agent je aktiviran.
Agent je aktiviran.
Trenutna pozicija je : x=250, y=150
Trenutna pozicija je : x=250, y=160
Trenutna pozicija je : x=0, y=0
Trenutna pozicija je : x=250, y=0
Trenutna pozicija je : x=0, y=150
Trenutna pozicija je : x=250, y=170
Trenutna pozicija je : x=0, y=10
Trenutna pozicija je : x=250, y=10
Trenutna pozicija je : x=0, y=160
Trenutna pozicija je : x=250, y=180
Trenutna pozicija je : x=0, y=20
Trenutna pozicija je : x=250, y=20
Trenutna pozicija je : x=0, y=170
Trenutna pozicija je : x=250, y=190
Trenutna pozicija je : x=0, y=30
Trenutna pozicija je : x=250, y=30
```

```
Trenutna pozicija je : x=0, y=180
Trenutna pozicija je : x=250, y=200
Trenutna pozicija je : x=0, y=40
Trenutna pozicija je : x=250, y=40
Trenutna pozicija je : x=0, y=190
Trenutna pozicija je : x=250, y=210
Trenutna pozicija je : x=0, y=50
Trenutna pozicija je : x=250, y=50
Trenutna pozicija je : x=0, y=200
Trenutna pozicija je : x=250, y=220
Trenutna pozicija je : x=0, y=60
Trenutna pozicija je : x=250, y=60
Trenutna pozicija je : x=0, y=210
Trenutna pozicija je : x=0, y=70
Trenutna pozicija je : x=250, y=70
Trenutna pozicija je : x=0, y=220
Trenutna pozicija je : x=250, y=230
Trenutna pozicija je : x=0, y=80
Trenutna pozicija je : x=250, y=80
Trenutna pozicija je : x=0, y=230
Trenutna pozicija je : x=250, y=240
Trenutna pozicija je : x=250, y=250
Trenutna pozicija je : x=0, y=90
Trenutna pozicija je : x=250, y=90
```

```
Trenutna pozicija je : x=0, y=240
Trenutna pozicija je : x=250, y=260
Trenutna pozicija je : x=0, y=100
Trenutna pozicija je : x=250, y=100
Trenutna pozicija je : x=0, y=250
Trenutna pozicija je : x=250, y=270
Trenutna pozicija je : x=0, y=110
Trenutna pozicija je : x=250, y=110
Trenutna pozicija je : x=0, y=260
Trenutna pozicija je : x=0, y=120
Trenutna pozicija je : x=250, y=120
Trenutna pozicija je : x=0, y=270
Trenutna pozicija je : x=0, y=130
Trenutna pozicija je : x=250, y=130
Trenutna pozicija je : x=0, y=140
Trenutna pozicija je : x=250, y=140
Trenutna pozicija je : x=260, y=150
Trenutna pozicija je : x=260, y=160
Trenutna pozicija je : x=10, y=150
Trenutna pozicija je : x=260, y=170
Trenutna pozicija je : x=10, y=160
Trenutna pozicija je : x=260, y=180
Trenutna pozicija je : x=10, y=170
Trenutna pozicija je : x=10, y=180
Trenutna pozicija je : x=260, y=190
Trenutna Pozicija je : x=2600, y=200
Trenutna pozicija je : x=10, y=190
Trenutna pozicija je : x=260, y=210
```

```
Trenutna pozicija je : x=10, y=200
Trenutna pozicija je : x=260, y=220
Trenutna pozicija je : x=10, y=210
Trenutna pozicija je : x=260, y=230
Trenutna pozicija je : x=10, y=220
Trenutna pozicija je : x=260, y=240
Trenutna pozicija je : x=10, y=230
Trenutna pozicija je : x=260, y=250
Trenutna pozicija je : x=10, y=240
Trenutna pozicija je : x=260, y=260
Trenutna pozicija je : x=10, y=250
Trenutna pozicija je : x=260, y=270
Trenutna pozicija je : x=10, y=260
Trenutna pozicija je : x=10, y=270
Trenutna pozicija je : x=260, y=0
Trenutna pozicija je : x=260, y=10
Trenutna pozicija je : x=260, y=20
Trenutna pozicija je : x=260, y=30
Trenutna pozicija je : x=260, y=40
Trenutna pozicija je : x=260, y=50
Trenutna pozicija je : x=260, y=60
Trenutna pozicija je : x=260, y=70
Trenutna pozicija je : x=260, y=80
Trenutna pozicija je : x=260, y=90
Trenutna pozicija je : x=260, y=100
Trenutna pozicija je : x=260, y=110
Trenutna pozicija je : x=260, y=120
Trenutna pozicija je : x=260, y=130
Trenutna pozicija je : x=260, y=140

Rezultat agenta je : x=9, y=39
```

Slika 5-2 Ispis izvođenja SDS programa

Dakle može se zamijetiti sa svi agenti, odnosno dretve programa, pretražuju neovisno u svom kvadrantu, te ispisuju svoju poziciju nakon pomaka za 10 piksela. Nakon što jedan agent pronađe rješenje i ispiše ga, on to dojavljuje drugim agentima, koji se nakon toga gase i program završava.

5.2 Ispitivanje programa za otkrivanje promjena u slikama

Programu su za ulaz dane dvije slike. Prva slika je identična velikoj slici kao u prethodnom eksperimentu, dok je druga slika slična prvoj, uz određeni broj nasumično izmijenjenih piksela. Ulaz programa je kao što je prikazano na slici Slika 4-2. Rezultat koji je program dao u obliku bmp slike je prikazan na slici Slika 5-3. Promjene se istaknute crvenim kvadratićima radi lakšeg uočavanja.

Slika 5-3 Rezultat programa za detekciju promjena

5.3 Ispitivanje programa za umetanje digitalnog vodenog žiga i njegovo izvlačenje

Program koji je testiran je preuzet sa Digital Watermarking [6]. Program se naziva Xie. Napravljen od strane Petera Meervalda, a implementira algoritam za umetanje i ekstrakciju digitalnog vodenog žiga razvijen od Liehue Xie i Gonzalo R. Arcea. Program radi isključivo sa slikama pgm formata, koje su dimenzija 512x512 piksela i koje su 8 bitne. Program radi kao MS Dos aplikacija.

Programom se prvo kreira potpis, odnosno ključ koji će se primijeniti za generiranje lomljivog digitalnog vodenog žiga.. Ključ koji je korišten u eksperimentima je 'Stochastic Diffusion Search'. Nakon što korisnik stvori svoj potpis on ga programom implementira u sliku. Potom može izmijeniti sliku, i programom izvući potpis nazad. Program kao rezultat daje korelaciju izvučenog i originalnog potpisa. Što je korelacija veća manji dio slike je promijenjen, i obratno. Detaljne upute za korištenje programa se mogu pronaći u Manualu na Digital Watermarking[6].

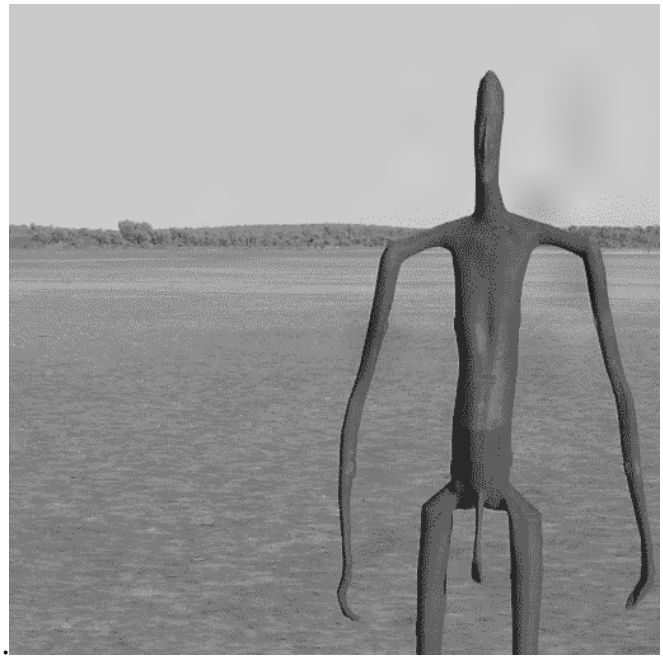
Program je testiran tako da je odabrano 10 slika koje zadovoljavaju kriterije programa. Svim slikama je dodan žig, te su potom mijenjanje. Nakon toga se izvlačio potpis i gledana je njegova korelacija s originalnim potpisom.

Svaka slika je testirana 15 puta na određeni način. Prvi stupac tablica sa rezultatima pokazuje rezultate koji su ostvareni tako što su uklanjani blokovi slika od gornjeg lijevog kuta. To znači na je sa koordinata 50 x 50 uklonjen blok slike veličine 50 x 50 piksela, potom 100 x 100 itd.

Drugi stupac pokazuje rezultate u kojima su blokovi uklanjani od donjeg desnog kuta, sa koordinata 462 x 462. Blokovi slika su uklanjani prema gore lijevo, u veličinama 50 x 50, 100x 100 piksela itd.

Treći stupac prikazuje rezultate gdje su blokovi uklanjani od sredine slike, koordinate 256 x 256 prema rubovima u opisanim veličinama blokova.

Navedeni način testiranja je odabran tako da se omogući isto mijenjanje svih slika.



Slika 5-4 Prva slika za testiranje Xie programa

U nastavku su prikazane slike nad kojima su vršeni eksperimenti i dobiveni rezultati. Za sliku Slika 5-4 korelacija ključa koji se dobije iz neizmijenjene slike je 0,900. Ostatak rezultata je dan u tablici Tablica 5-1.

Tablica 5-1 Rezultati izvođenja Xie programa za prvu sliku

	1.	2.	3.
50x50	0,75	0,625	0,575
100x100	0,725	0,5	Ne može se dobiti žig
150x150	0,575	Ne može se dobiti žig	Ne može se dobiti žig
200x200	0,45	Ne može se dobiti žig	Ne može se dobiti žig
250x250	Ne može se dobiti žig	Ne može se dobiti žig	Ne može se dobiti žig

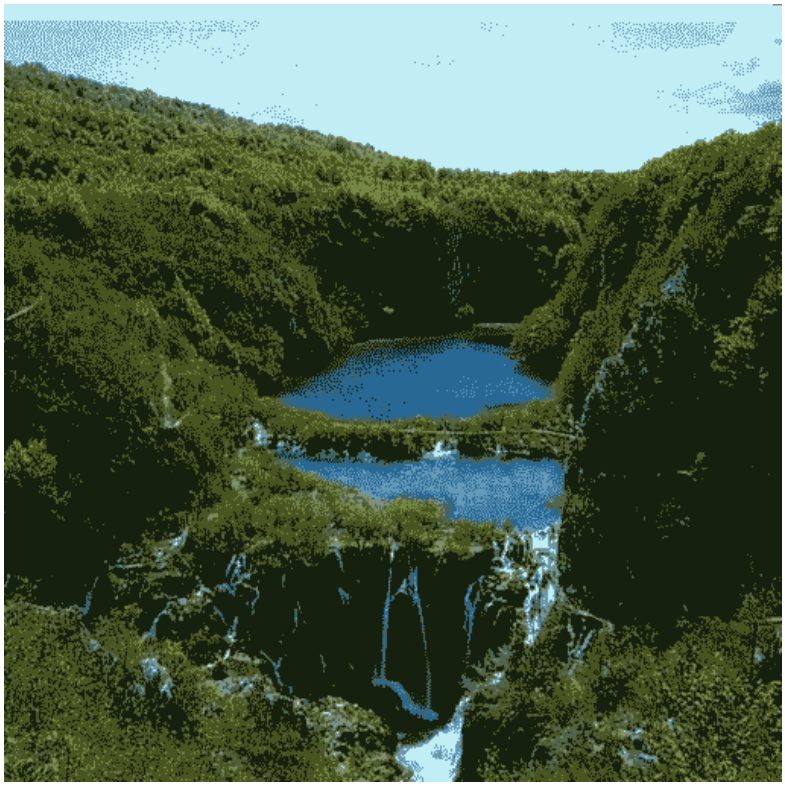
Za sliku Slika 5-5 korelacija ključa koji se dobije iz neizmijenjene slike je 0,925. Ostatak rezultata je dan u tablici Tablica 5-2.

Slika 5-5 Druga slika za testiranje Xie programa

Tablica 5-2 Rezultati izvođenja Xie programa za drugu sliku

	1.	2.	3.
50x50	0,750	0,875	0,800
100x100	0,770	0,900	0,800
150x150	0,800	0,725	0,700
200x200	0,670	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-6 korelacija ključa koji se dobije iz neizmijenjene slike je 0,950. Ostatak rezultata je dan u tablici Tablica 5-3.



Slika 5-6 Treća slika za testiranje Xie programa

Tablica 5-3 Rezultati izvođenja Xie programa za treću sliku

	1.	2.	3.
50x50	0,850	0,875	0,900
100x100	0,775	0,825	0,875
150x150	Ne može se izvući žig	0,800	0,775
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-7 korelacija ključa koji se dobije iz neizmijenjene slike je 1,000. Ostatak rezultata je dan u tablici Tablica 5-4**.**

Slika 5-7 Četvrta slika za testiranje Xie programa

Tablica 5-4 Rezultati izvođenja Xie programa za četvrtu sliku

	1.	2.	3.
50x50	0,925	0,950	0,900
100x100	0,850	0,900	0,850
150x150	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-8 korelacija ključa koji se dobije iz neizmijenjene slike je 1,000. Ostatak rezultata je dan u tablici Tablica 5-5.

Slika 5-8 Peta slika za testiranje Xie programa

Tablica 5-5 Rezultati izvođenja Xie programa za petu sliku

	1.	2.	3.
50x50	0,925	0,925	0,950
100x100	0,875	0,950	0,925
150x150	0,720	Ne može se izvući žig	Ne može se izvući žig
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-9 korelacija ključa koji se dobije iz neizmijenjene slike je 0,975. Ostatak rezultata je dan u tablici Tablica 5-6.

Slika 5-9 Šesta slika za testiranje Xie programa

Tablica 5-6 Rezultati izvođenja Xie programa za šestu sliku

	1.	2.	3.
50x50	0,850	0,900	0,850
100x100	0,725	0,825	0,850
150x150	Ne može se izvući žig	0,775	0,700
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-10 korelacija ključa koji se dobije iz neizmijenjene slike je 1,000. Ostatak rezultata je dan u tablici Tablica 5-7.

Slika 5-10 Sedma slika za testiranje Xie programa

Tablica 5-7 Rezultati izvođenja Xie programa za sedmu sliku

	1.	2.	3.
50x50	0,950	0,975	0,925
100x100	0,775	0,850	0,900
150x150	0,800	0,775	0,800
200x200	0,625	Ne može se izvući žig	0,575
250x250	0,450	Ne može se izvući žig	Ne može se izvući žig
300x300	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-11 korelacija ključa koji se dobije iz neizmijenjene slike je 0,925. Ostatak rezultata je dan u tablici Tablica 5-8.

Slika 5-11 Osma slika za testiranje Xie programa

Tablica 5-8 Rezultati izvođenja Xie programa za osmu sliku

	1.	2.	3.
50x50	0,800	0,900	0,825
100x100	0,775	Ne može se izvući žig	0,775
150x150	0,725	Ne može se izvući žig	0,600
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

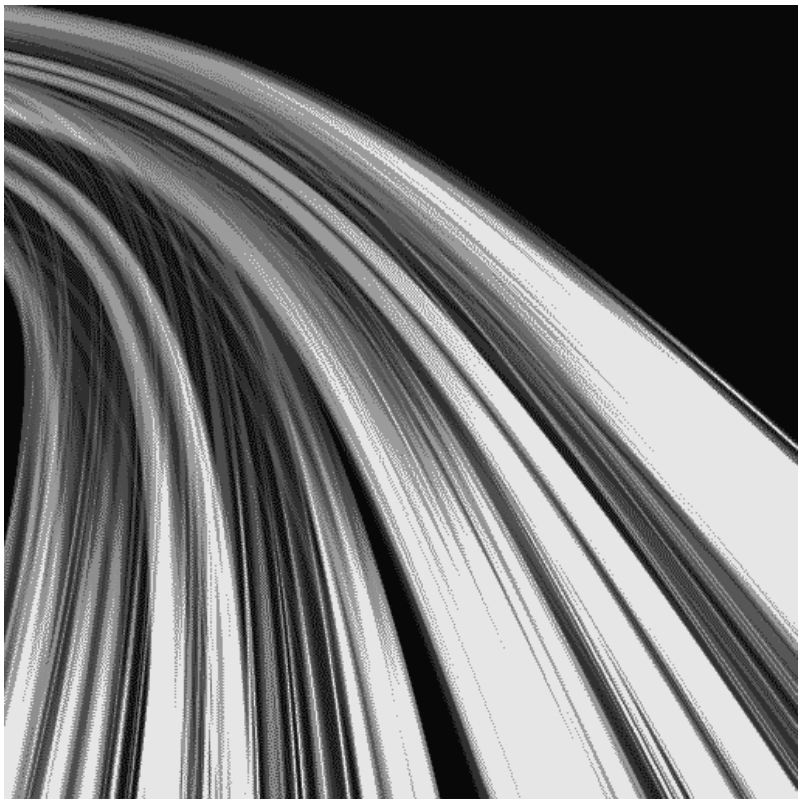
Za sliku Slika 5-12 korelacija ključa koji se dobije iz neizmijenjene slike je 1,000. Ostatak rezultata je dan u tablici Tablica 5-9.

Slika 5-12 Deveta slika za testiranje Xie programa

Tablica 5-9 Rezultati izvođenja Xie programa za devetu sliku

	1.	2.	3.
50x50	0,900	1,00	0,950
100x100	0,925	Ne može se izvući žig	0,925
150x150	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

Za sliku Slika 5-13 korelacija ključa koji se dobije iz neizmijenjene slike je 0,975. Ostatak rezultata je dan u tablici Tablica 5-10**.**



Slika 5-13 Deseta slika za testiranje Xie programa

Tablica 5-10 Rezultati izvođenja Xie programa za desetu sliku

	1.	2.	3.
50x50	0,850	0,875	0,975
100x100	0,825	0,850	0,825
150x150	Ne može se izvući žig	0,775	0,750
200x200	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig
250x250	Ne može se izvući žig	Ne može se izvući žig	Ne može se izvući žig

5.3.1 Zaključak ispitivanja Xie programa

Kao što se vidi iz rezultata program uspješno izvlači vodeni žig sve dok je u prosjeku najviše primijenjeno 150 x 150 piksela iz slike, odnosno negdje oko 8,5 % slike. Ponegdje je moguće izvući žig i ako je veći dio slike promijenjen. To ovisi o samoj slici. Rezultati ukazuju na to da jednostavnije slike daju bolje rezultate. Jednostavnije slike su u ovom kontekstu one koje su cijele sličnih boja. Sama pozicija promjena nije bitnije utjecala na rezultate

6. Zaključak

Lomljivi digitalni vodeni žigovi su dobar način autentifikacije slika i zaštite prava. Iako su zasad još uvijek u razvoju predstavljaju moćan alat. Ukoliko se uspiju efikasno razviti slabo lomljivi digitalni vodeni žigovi, uz navedenu problematiku 2.4.1, će prevladati obične lomljive žigove.

Pokazano je da se SDS u principu može koristiti za stohastičke i dinamičke optimizacijske probleme. Prikazan je njegov algoritam i mogućnost prilagodbe istog. Njegova mogućnost laganog prilagođavanja između naglaska na lokalni prostor ili sveukupni prostor pretrage prostora rješenja, uz njegovu parcijalnu evaluaciju, čini ga potencijalno jako korisnim u primjeni rješavanja navedenih problema. Istražuje se njegova primjena u aplikacijama za traženje manjih dijelova sveukupne slike, npr. traženja očiju na licu.

Na kraju je prikazano jedno od mogućih rješenja za implementaciju tih dvaju tehnika. Unatoč činjenici što su obje relativno nove i još nerazvijene i neistražene vjeruje se da mogu osigurati rješenje za brojne probleme. SDS algoritmi se već pokušavaju implementirati u programe za prepoznavanje lica, a u rudimentarnom obliku SDS komunikacija se koristi u nekim bežičnim lokalnim mrežama za inicijalno spajanje bežičnih stanica. Proizvođač fotoaparata Nikon već ima razvijene program za autentifikaciju slika koje su napravljene određenim modelima njegovih fotoaparata. Zaštita je ostvarena rabljenjem lomljivih digitalnih vodenih žigova.

7. Literatura

[1] Prvi izvor: http://www.scholarpedia.org/article/Stochastic_diffusion_search

[2] Drugi izvor: Kris de Meyer, Slawomir J. Nasuto, Mark Bishop: Stochastic Diffusion Search: Partial Function Evaluation in Swarm Intelligence Dynamic Optimisation

[3] Treći izvor: Kennedy, J, Eberhart, R C (2001) Swarm Intelligence. Morgan Kaufmann

[4] Četvrti izvor: http://en.wikipedia.org/wiki/Digital_watermarking

[5] Peti izvor: R. Chandramouli, Nasir Memon, Majid Rabbani: Digital Watermarking

[6] Šesti izvor: http://www.cosy.sbg.ac.at/~pmeerw/Watermarking/source/

Sadržaj

Cilj ovog rada je upoznavanje s lomljivim digitalnim vodenim žigovima i stohastičkom difuzijskom pretragom. Objašnjena je osnovna tehnika implementacije vodenih žigova te je dan pregled pseudokoda na čijem principu se zasniva SDS. Također, dana je moguća implementacija vodenih žigova i SDS za otkrivanje promjena u zaštićenim slikama.

Ključne Riječi

lomljivi digitalni vodeni žigovi, stohastička difuzijska pretraga

Abstract

The goal of this paper is acquiring basic knowledge of digital watermarking and stochastic diffusion search. Therefore, basic technique of implementing digital watermarks is described along with pseudo code which is the core of the SDS. In addition to that one possible implementation of the two described is presented. The goal of implementation is detection of changes in protected pictures.

Keywords

fragile digital watermarks, stochastic diffusion search