UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Boštjan Čargo

NEINFORMIRANO ODKRIVANJE PODVOJENIH REGIJ V DIGITALNIH SLIKAH

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Aleš Leonardis

Ljubljana, 2009

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01514/2008
Datum: 01.09.2008

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **BOŠTJAN ČARGO**

Naslov: **NEINFORMIRANO ODKRIVANJE PODVOJENIH REGIJ V DIGITALNIH SLIKAH**

BLIND DETECTION OF DUPLICATE REGIONS IN DIGITAL IMAGES

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Proučite problematiko samodejnega odkrivanja digitalnih ponaredkov na področju slik, pri čemer se osredotočite na neinformirane postopke za odkrivanje podvojenih regij. Načrtajte in implementirajte ustrezne algoritme, s pomočjo katerih je možno odkrivanje podvojenih regij. Analizirajte učinkovitost teh algoritmov ter jih eksperimentalno ovrednotite na ustreznih slikovnih vzorcih.

Mentor:
prof. dr. Aleš Leonardis

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:
prof. dr. Franc Solina

Besedilo je oblikovano s prostim urejevalnikom besedil OpenOffice.Org Writer.
Slike so pripravljene s prostim urejevalnikom slik The GIMP.

Zahvala

Prijazno se zahvaljujem vsem, ki so mi na kakršenkoli način pomagali pri izdelavi diplomske naloge. Med njimi izpostavljam mentorja prof. dr. Aleša Leonardisa, ki me je usmerjal in mi svetoval, dr. Matjaža Jogana, ki mi je priskrbel iztočnice za raziskovanje problemskega področja, dr. Alina C. Popescuja, ki mi je posredoval pojasnila v zvezi z izvedbo glavnega algoritma, prof. dr. Hanyja Farida, ki je omogočil uporabo nekaterih preizkusnih slik, prijatelje in sodelavce, ki so me venomer vzpodbujali, starše, ki so verjeli vame v trenutkih, ko sam sebi nisem verjel, in Manco, ki mi je pokazala pot naprej.

Posebna zahvala gre tudi vsem požrtvovalnim razvijalcem širom sveta, ki prostovoljno prispevajo k nadaljnjemu razvoju prostega programja in s tem omogočajo, da ideja o svobodi živi naprej. Za izdelavo tega diplomskega dela je bilo med drugim uporabljeno tudi prosto programje. Enako zahvalo namenjam snovalcem, vzdrževalcem in denarnim podpornikom prosto dostopnih informacijskih zbirk ter vsem posameznikom in skupinam, ki prispevajo vsebine zanje.

Nenazadnje se zahvaljujem tudi vsem, ki ohranjanja narave in lepše prihodnosti planeta Zemlje ne prepuščajo naključju.

Posvečeno vsem,
ki sta jim življenjski vodili svoboda in resnica.

Kazalo

Seznam uporabljenih kratic in simbolov

Kratice

ADT – abstract data type – abstraktni podatkovni tip

BST – binary search tree – dvojiško iskalno drevo

CCD – charge-coupled device – vrsta svetlobnega tipala

CMOS – complementary metal-oxide-semiconductor – vrsta svetlobnega tipala

CFA – color filter array – mreža barvnih filtrov

CPV – cumulative percent variance – delež skupne variance

DRD – Duplicate Region Detector – detektor podvojenih regij; glavni algoritem, ki ga obravnava diplomska naloga

JDK – Java 2 Platform Standard Edition Development Kit – razvojno okolje za Java 2 Platform Standard Edition

JPEG – Joint Photographic Experts Group – standard, ki določa izgubni in brezizgubni zapis dvorazsežnih rastrskih slik polne barvne globine ter ustrezen kodek za njihovo branje in zapisovanje

JRE – Java 2 Platform Standard Edition Runtime Environment – izvajalno okolje za Java 2 Platform Standard Edition

LGPL – GNU Lesser General Public License – vrsta licence s pogoji razširjanja prostega programja

PCA – principal component analysis – analiza glavnih komponent

PNG – Portable Network Graphics – uveljavljen brezizgubni zapis dvorazsežnih rastrskih slik polne barvne globine

SNR – signal-to-noise ratio – razmerje med signalom in šumom

Simboli

N – št. slikovnih elementov v preiskovani sliki

N_b – št. blokov, ki jih pridobimo iz preiskovane slike (v Dodatku A označeno s simbolom n)

b – št. slikovnih elementov v bloku (v Dodatku A označeno s simbolom d)

u_k, v_k – vektorske komponente

$\vec{r}_k$, $\vec{r}_l$ – vektorja, ki hranita zapis dveh izvirnih blokov oziroma vhodnih vektorjev

$\vec{a}_k$, $\vec{a}_l$ – vektorja pripadajočih poenostavljenih predstavitev dveh izvirnih blokov

$\vec{s}_i$, $\vec{s}_j$ – vektorja pripadajočih poenostavljenih predstavitev v leksikografsko urejenem seznamu

i, j – indeksa vektorjev v leksikografsko urejenem seznamu

$\vec{I}_k$ – ortonormirani vektorji baze prostora

H_n – natančna zgornja meja medindeksne razlike vektorskih parov kandidatov

C – množica kandidatov

H_d – prag velikosti absolutnih medsebojnih odmikov bločnih parov

H_f – prag pogostnosti enakih medsebojnih odmikov bločnih parov

$M_{a,b}$ – množica vektorskih parov poenostavljenih predstavitev blokov, katerih pripadajoči pari slikovnih izsekov imajo enak medsebojni dvodimenzionalni odmik $((\mathrm{Fx_a}, \mathrm{Fy_b}))$

(x_i, y_i), (x_j, y_j) – dvodimenzionalna odmika vektorjema $\vec{s}_i$ in $\vec{s}_j$ pripadajočih slikovnih izsekov od zgornjega levega roba slike

$(\mathrm{Fx_a}, \mathrm{Fx_b})$ – medsebojni dvodimenzionalni odmik slikovnih izsekov, ki pripadata vektorjema poenostavljenih predstavitev nekega bločnega para

o – absolutna vrednost medsebojnega odmika dveh slikovnih izsekov

ε – delež zanemarjene variance vzdolž glavnih osi prostora blokov, pridobljenih iz preiskovane slike

D_t – št. dimenzij postora poenostavljenih predstavitev blokov (v Dodatku A označeno s simbolom p)

λ_i – lastne vrednosti kovariančne matrike, dobljene po metodi PCA

$\vec{q}_i$ – vektorji poenostavljenih predstavitev, katerih komponente so kvantizirane

Q – št. kvantizacijskih stopenj

$\mathbf{Q}$ – matrika vektorjev $\vec{q}_i$

$\mathbf{S}$ – matrika leksikografsko urejenih vektorjev $\vec{q}_i$

f – velikost vektorja značilnic preiskovane slike

N_{BST} – št. vozlišč dvojiškega iskalnega drevesa, ki hrani različne medsebojne dvodimenzionalne odmike med slikovnimi izseki kandidatov

M_1 – velikost dela porabljenega delovnega pomnilnika, na katerega vpliva velikost vektorja značilnic

M_2 – velikost dela porabljenega delovnega pomnilnika, na katerega vpliva velikost dvojiškega iskalnega drevesa

T – pričakovana (povprečna) časovna zahtevnost

T_w – časovna zahtevnost v najslabšem primeru

S – pričakovana (povprečna) prostorska zahtevnost

S_w – prostorska zahtevnost v najslabšem primeru

r – polmer filtrirnega okna filtra mediana

S_{key} – velikost pomnilnika, ki ga v JRE zaseda ključ vozlišča

S_{val} – velikost pomnilnika, ki ga v JRE zaseda vrednost v vozlišču

S_{lref}, S_{rref} – velikost pomnilnika, ki ga v JRE zaseda referenca na levega oziroma desnega naslednika

S_{obj} – velikost pomnilnika, ki ga v JRE zasedajo sistemski podatki, ki so potrebni za predstavitev vsakega javanskega objekta

$\mathbf{R}$ – matrika vhodnih vektorjev, s katerimi so zapisani izvirni bloki

$\vec{x}$ – vektor povprečnih vrednosti koordinat vhodnih vektorjev

$\mathbf{X}$ – matrika enakih vektorjev $\vec{x}$

$\mathbf{P}$ – matrika prilagojenih vhodnih vektorjev, katerih povprečje vsake koordinate je 0

$\mathbf{K}$ – kovariančna matrika, dobljena po metodi PCA

$\vec{v}_i$ – lastni vektorji kovariančne matrike

$\mathbf{F}$ – vektor značilnic

$\mathbf{A}$ – matrika izhodnih vektorjev z zapisi poenostavljenih predstavitev blokov

$\mathbf{P_a}$ – matrika približkov prilagojenih vhodnih vektorjev

$\mathbf{R_a}$ – matrika približkov vhodnih vektorjev

Povzetek

Diplomska naloga se nanaša na področje digitalne obdelave slik. Njen namen je osvetliti področje samodejnega odkrivanja digitalnih ponaredkov, znotraj tega pa opisati enega od neinformiranih postopkov za odkrivanje podvojenih regij: tako imenovani algoritem Duplicate Region Detector (DRD). Algoritem je osnovan na analizi glavnih komponent, poenostavitvi slikovnih blokov in njihovemu medsebojnemu leksikografskemu primerjanju. Izvedbo v programskem jeziku java smo preizkusili na množici pozitivnih in negativnih primerov. Z dobljenimi rezultati smo ovrednotili uspešnost in učinkovitost ter odpornost proti izbranim postopkom digitalne obdelave. Poglaviten zaključek je ocena, da moramo za izboljšanje uspešnosti opustiti vnaprej določene vrednosti parametrov odkrivanja in te vrednosti določati na podlagi lastnosti oziroma vsebine preiskovanih slik.

Ključne besede

računalniški vid, ponarejanje slik, podvojena regija, analiza glavnih komponent, poenostavljena predstavitev, seznam kandidatov, izločanje kandidatov, slika podvojitev

Abstract

This work refers to the research area of digital image processing. Its main purpose is to elucidate the field of automatic digital forgery detection and, within its scope, describe a particular algorithm for blind detection of duplicated image regions: the so-called Duplicate Region Detector (DRD). The algorithm is based on principal component analysis, reduction of image blocks representations, and their lexicographical comparison. Our java implementation was tested on a population with positive and negative examples. Test results enabled us to assess efficiency and efficacy, as well as robustness against specific digital image processing operations. The principal conclusion is a thesis that the algorithm can only be made more efficient by abandoning fixed parameter values that are set in advance in favour of values that are defined based on characteristics of each input image.

Keywords

computer vision, image tampering, duplicated image region, principal component analysis, reduced representation, candidate list, candidate removal, duplication map

Poglavje 1

Uvod

Dandanes smo priča izrednemu razmahu digitalnih vsebin. Vsebine, shranjene v digitalnih zapisih na digitalnih nosilcih podatkov, se z vse večjo hitrostjo uveljavljajo na vseh področjih človekovega življenja in ustvarjanja. Razlogi za uveljavljanje digitalnih zapisov v svetu večpredstavnosti se nam danes zdijo nekaj samo po sebi umevnega; o njih se pravzaprav ne vprašujemo več. V preteklosti pa je bilo drugače.

Tehnološki razvoj je v novejši zgodovini nanizal vrsto razlogov za uvedbo in uporabo digitalnih vsebin. Vsako razvojno obdobje je z vsaj enim razlogom upravičevalo prednost digitalnih vsebin pred analognimi. Sprva je bil glavni razlog povezan z raziskovanjem, preizkušanjem in nadgrajevanjem novih tehnologij. Nadalje je bilo vsebine, shranjene v digitalnih zapisih, predvsem možno lažje obdelovati za najrazličnejše namene ter jih hitreje, enostavneje in predvsem veliko ceneje razmnoževati in razpošiljati. Sčasoma se je pokazalo, da so digitalne vsebine tudi prikladnejše za evidentiranje in katalogizacijo. Kasneje je postal pomemben vidik trajnosti vsebin[1]. V zadnjem času igrata glavno vlogo pri odločanju za digitalno tehnologijo njena bistveno nižja cena in skoraj že izključna dostopnost.

Zlorabe v digitalnem svetu

Skupaj s poplavo cenovno zelo dostopnih digitalnih naprav za zajemanje, prenos, obdelavo in shranjevanje podatkov ter ob nesluteni razširjenosti interneta prinašajo digitalne vsebine s seboj tudi številna tveganja in razmeroma veliko nevarnost zlorab. Med slednjimi izpostavljamo nepooblaščen dostop do vsebin in njihovo prisvojitev, nedovoljeno spreminjanje, razmnoževanje ter prikrivanje identitete avtorja oziroma imetnika avtorskih pravic. Poseben primer zlorab je tudi ponarejanje in nadomeščanje pristnih vsebin s ponaredki.

S ciljem preprečevati zlorabe digitalnih vsebin znanstveniki razvijajo raznovrstne zaščitne in diagnostične metode. Ponavadi so to visokospecializirani postopki, ki bolj ali manj učinkovito preprečujejo oziroma odkrivajo določeno vrsto zlorab. Obstajajo

1 Problematika trajnosti je razdelila strokovno javnost. Začetno trdno prepričanje, da so vsebine v digitalnih zapisih v splošnem trajnejše od tistih v tradicionalnih analognih oblikah, je namreč splahnelo. Seveda pa moramo ločevati trajnost zapisov od sposobnosti ohranjanja vsebin.

pa tudi metode, ki so učinkovite na več področjih. Primer takšnih metod je opremljanje vsebin z **digitalnimi vodnimi žigi** (*digital watermarks*), katere izpeljanke so na primer uspešne tako pri določanju lastništva kot pri ugotavljanju pristnosti vsebin. Kljub naporom, vloženim v raziskave na omenjenem področju, pa se zdi, da so iznajdljivi zlorabljalci vedno vsaj en korak v prednosti in da tveganja zlorab digitalnih vsebin ni mogoče v celoti odpraviti.

Metoda oziroma algoritem, na katerega se osredotočamo v tej diplomski nalogi, je namenjena ugotavljanju pristnosti digitalnih slik. S potrebnimi prilagoditvami je uporabna tudi pri obravnavi drugih vrst digitalnih vsebin, na primer zvočnih posnetkov in videov. Takšno širšo uporabnost lahko običajno pripišemo tudi večini drugih metod, ki se ukvarjajo z reševanjem istega problema.

1.1 PROBLEMATIKA PONAREJANJA DIGITALNIH FOTOGRAFIJ

Ponarejanje fotografij je staro skoraj toliko kot fotografija sama. Znani so zgodovinski primeri, ko se je fotografije ponarejalo z namenom politične ali celo vojne propagande in manipuliranja javnega mnenja preko množičnih občil. V takratnih preddigitalnih časih je bilo ponarejanje zahteven in dolgotrajen postopek, ki je terjal posebno opremo in visoko usposobljenost izvajalcev. Vsi ti dejavniki so botrovali njegovi razmeroma visoki ceni. Pojavitev in uveljavitev digitalnih naprav in digitalnih vsebin sta vse našteto korenito spremenili.

Danes je ponarejanje fotografij enostavnejše kot kadar koli prej. Nizkocenovni in hkrati kakovostni digitalni fotoaparati in optični čitalniki, dostopna strojna oprema, brezplačno ali celo prosto programje za obdelavo digitalnih slik in barvni tiskalniki z verodostojnim izpisom ponujajo možnost ustvarjanja prepričljivih ponaredkov praktično vsakomur. Omenjene okoliščine znatno povečujejo verjetnost ponarejanja fotografij, ki s tem izgubljajo svojo dokazno vrednost. Uspešna uporaba ponaredkov v sodnih postopkih ali množičnih občilih namreč lahko zasuka razvoj dogodkov v napačno smer in pusti dolgotrajne neželene, nemalokrat celo škodljive posledice.

Z vse bolj izpopolnjenimi orodji za obdelavo digitalnih slik postajajo digitalni ponaredki izjemno prepričljivi. Najboljši ponaredki ne vsebujejo niti najmanjših vidnih sledov ponarejanja oziroma digitalne obdelave, torej se na videz ne ločijo od pristnih fotografij. Dodatno težavo pri odkrivanju ponaredkov predstavlja vse večja količina digitalnih slik, ki se kopičijo na nosilcih podatkov. Z uporabo človeških zmogljivosti je naraščajoče količine podatkov praktično nemogoče preveriti v še razumnem času. Postavlja se nam vprašanje, kako se uspešno spopasti z odkrivanjem ponaredkov. Človeku mora tu očitno priskočiti na pomoč računalnik.

1.1.1 METODE ZA UGOTAVLJANJE PRISTNOSTI DIGITALNIH FOTOGRAFIJ

K reševanju problema ugotavljanja pristnosti digitalnih fotografij lahko pristopimo z uporabo enega od dveh principov. Bistvena razlika med njima je v izvajanju postopka

naknadne obdelave nastalih fotografij na način, ki nam bo kasneje omogočil čim lažje odkrivanje nedovoljenih sprememb v njih.

Prvi princip: fotografijo takoj po njenem nastanku opremimo s posebno prostim očem nevidno oznako, tako imenovanim digitalnim vodnim žigom. Ta oznaka vsebuje podatke o lastništvu slike, v katero je vstavljena, in je občutljiva na postopke digitalne obdelave. Ko želimo za določeno fotografijo ugotoviti ali je pristna, enostavno preverimo, če pripadajoča slika vsebuje prvotno vstavljeni digitalni vodni žig. Postopkom, ki uporabljajo ta princip, pravimo **informirani postopki**[2] (*non-blind techniques*).

Drugi princip: pri ugotavljanju pristnosti določene fotografije se zanašamo na posledice, ki jih digitalna obdelava pusti v pripadajoči digitalni sliki, ne da bi bilo to nujno opazno. Postopkom, ki upoštevajo ta princip, pravimo **neinformirani postopki** (*blind techniques*) [14].

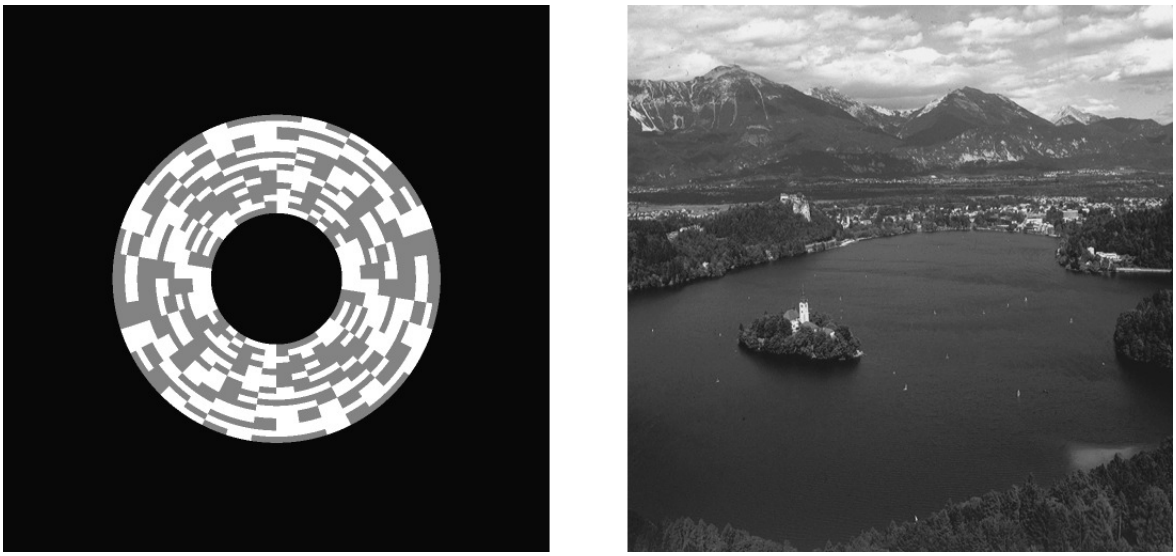
Opišimo na kratko informirane in neinformirane postopke ter njihove prednosti in slabosti.

1.1.2 LASTNOSTI INFORMIRANIH POSTOPKOV

Pri informiranih postopkih v digitalno sliko vstavimo posebno oznako, ki nam bo pomagala pri naknadnem ugotavljanju ali je bila slika brez naše vednosti spremenjena. Oznaka v sliki mora biti prostim očem nevidna; postopek njenega vstavljanja ne sme zmanjšati zaznavne kakovosti izvirne neoznačene slike. Oznaka mora imeti tudi lastnost šibkosti: to pomeni, da jo morajo običajni postopki digitalne slikovne obdelave občutno popačiti ali celo uničiti.

Skupina informiranih postopkov za ugotavljanje pristnosti digitalnih fotografij obsega množico različnih zaščitnih shem. Posamezna shema narekuje vstavljanje oznake bodisi v **prostorsko domeno** (*spatial domain*) [5], **frekvenčno domeno** (*frequency domain*) [4], **domeno valčkov** (*wavelet domain*) [16] ali katero drugo domeno digitalne slike. Izbira sheme je odvisna predvsem od zahtev po odkrivanju različnih posegov v digitalno sliko: v določenih primerih želimo ugotoviti že najmanjše spremembe slike, drugod pa takšne spremembe dopuščamo in se osredotočamo le na bolj grobe posege. Sprememba, ki je ponavadi ne obravnavamo kot nedovoljen poseg v sliko, je na primer shranjevanje v zapisu JPEG z višjo stopnjo stiskanja slikovnih podatkov. Včasih nas zanima le določena vrsta operacij, ki so bile izvedene med digitalno slikovno obdelavo, in smemo druge prezreti. Med kopico zaščitnih shem, ki so bile razvite v tovrstne namene, navedimo na primer **krhke digitalne vodne žige** *(fragile watermarks)* [13], **vstavljene podpise** (*embedded signatures*) [17] in **hibridne digitalne vodne žige** (*hybrid watermarks*) [12]. Primer digitalnega vodnega žiga, ki ga vstavljamo v frekvenčno domeno, je prikazan na sliki 1.1.

2 Tu velja opozorilo, da imata izraza *informirano* in *neinformirano* v drugih kontekstih lahko drugačen pomen. Primer takega konteksta je področje digitalnega vodnega tiska, kjer z zvezama *informiran* postopek in *neinformiran* postopek povemo, da postopek za odkrivanje digitalnega vodnega žiga v označeni sliki za svoje delovanje potrebuje izvirno neoznačeno sliko, oziroma da te slike ne potrebuje [3].



Slika 1.1: Primer digitalnega vodnega žiga in slike, opremljene z njim.[3]

Glavna pomanjkljivost informiranih postopkov je zahteva po čimprejšnji vstavitvi digitalnega vodnega žiga v nastalo fotografijo. V idealnem primeru to pomeni, da moramo digitalno fotografijo opremiti z digitalnim vodnim žigom že pred prvim zapisovanjem v notranji pomnilnik oziroma na pomnilniško napravo; samo v tem primeru bomo zagotovili kar največjo zanesljivost in uspeli omejiti zlorabe na najmanjšo možno mero. Takoj po zapisovanju na izbrani medij je digitalna slika namreč že lahko tarča zlorabljalcev. Ta dokaj stroga zahteva pogojuje izvajanje tovrstnih postopkov s posebej za ta namen opremljenimi digitalnimi fotoaparati. Dodatna oteževalna okoliščina za informirane postopke je dejstvo, da se zanašajo na naslednjo predpostavko: "Digitalnega vodnega žiga, ki ga najdemo v neki označeni sliki, po njeni digitalni obdelavi ne moremo na enostaven način izbrisati iz slike in ga ponovno vstaviti vanjo." V času nastajanja te diplomske naloge ta predpostavka še ni ne potrjena ne ovržena. Če je predpostavka napačna, potem je smiselnost uporabe informiranih postopkov za ugotavljanje pristnosti digitalnih fotografij v celoti vprašljiva.

1.1.3 LASTNOSTI NEINFORMIRANIH POSTOPKOV

Neinformirani postopki za ugotavljanje pristnosti digitalnih fotografij se ne opirajo na predhodno označevanje pripadajočih slik z digitalnimi podpisi oziroma vodnimi žigi. Njihovi vhodni podatki so kakršnakoli – ne nujno označena – digitalna slika. Postopki iz te skupine tako odkrivajo ponaredke zgolj na podlagi posledic digitalne obdelave, ki so vidne v neki slikovni domeni ali pa se odražajo kako drugače, na primer kot spremembe statističnih lastnosti slike. Ugotovljiva oziroma dovolj izrazita odstopanja

3 Predstavljen je krožno-simetrični digitalni vodni žig, ki je bil predlagan in podrobno opisan v [23]. Leva slika prikazuje izgled vodnega žiga v amplitudni domeni frekvenčnega spektra. Vodni žig je bil z operacijo seštevanja vstavljen v amplitudno domeno frekvenčnega spektra fotografije Blejskega jezera. Desna slika prikazuje rezultat: digitalno fotografijo, ki vsebuje vodni žig.

od običajnih vrednosti nakazujejo na možnost poneverbe.

Tipičen postopek, ki ga uporabljajo ponarejevalci za izdelavo digitalnih ponaredkov, je zaporedje naslednjih operacij:

1. Zajem digitalne fotografije in začetno shranjevanje na pomnilniški medij, praviloma v izgubnem zapisu JPEG.
2. Geometrijska obdelava slike: **sprememba velikosti** (*cropping*), **zasuk** (*rotation*), **razteg** ali **skrčenje** (*scaling*).
3. Kopiranje slikovnih elementov iz enega dela slike na druge dele slike (podvajanje regij).
4. Sprememba svetlosti celotne ali posameznih delov slike z uporabo nelinearne transformacije, na primer funkcije gama.
5. Dodajanje majhne količine umetnega šuma z namenom prikritja obdelave slike.
6. Shranjevanje obdelane slike, praviloma v izgubnem zapisu JPEG.

Opazovalec, ki presoja verodostojnost končnega izdelka, najverjetneje ne opazi vidnih sledov uporabljenega postopka ponarejanja. Toda tudi pri najboljših ponaredkih se v digitalno obdelani sliki odražajo nevidne spremembe, ki so posledica operacij obdelave. Ponavadi so te spremembe posebne korelacije, ki jih v pristnih slikah ne najdemo. Za uspešno odkrivanje takšnih korelacij zadošča že uporaba razmeroma enostavnih algoritmov.

Zgoraj opisani postopek spremeni digitalno obdelano sliko tako, da vanjo v naslednjem vrstnem redu vnese pet različnih sprememb:

- posledice ponovnega vzorčenja,
- podvojene regije,
- posledice različnih nelinearnih transformacij svetlosti,
- različne variance količine šuma in
- posledice dvakratnega stiskanja z algoritmom JPEG.

Prednost neinformiranih postopkov pred informiranimi je očitna, saj pri njih odpade dodatni korak vnaprejšnje obdelave digitalnih slik, ki jih želimo obvarovati pred nedovoljenimi posegi. Ti postopki so zato bolj univerzalni od informiranih. Po drugi strani lahko uporaba neinformiranih postopkov v nekaterih primerih sproži lažni alarm. To se zgodi, ko postopki kot ponaredke obravnavajo tudi digitalne fotografije, ki niso bile ponarejene. Razlog za takšne napačne obravnave tiči v dejstvu, da neinformirani postopki težje ločujejo med operacijami digitalne obdelave, ki so za določen kontekst še dovoljene, in tistimi, ki so prepovedane.

1.1.4 PREGLED PODROČJA NEIFORMIRANEGA ODKRIVANJA DIGITALNIH PONAREDKOV

Popescu je v [20] in skupaj s Faridom v [21] predlagal in opisal nekatere neinformirane postopke, s katerimi si lahko pomagamo pri preverjanju pristnosti digitalnih fotografij. Postopki temeljijo na isti predpostavki: digitalna obdelava slike spremeni bodisi frekvenčno domeno, določene statistične ali katere druge lastnosti slike, ne da bi končni opazovalec to nujno opazil. Ti postopki so sicer le del množice vseh postopkov, ki služijo istemu namenu:

- odkrivanje posledic **ponovnega vzorčenja** (*resampling*)

  Postopki ponarejanja pogosto terjajo spajanje izsekov več fotografij. Običajno je za željeno prilagoditev potrebno posamezen izsek raztegniti, skrčiti ali zasukati. Vsaka od teh operacij v praksi pomeni "polaganje" izseka na novo vzorčevalno mrežo z uporabo določene metode interpolacije. Ponovno vzorčenje vnese v fotografijo prostim očem nevidne korelacije, ki jih lahko algoritemsko zaznamo.

- odkrivanje posledic **interpolacije mreže barvnih filtrov** (*color filter array (CFA) interpolation*)

  Večina digitalnih fotoaparatov je opremljena z enim samim slikovnim tipalom (CCD ali CMOS), ki preko mreže barvnih filtrov zajema barvno informacijo iz okolice. Čeprav je informacija sestavljena iz treh barvnih komponent, fotoaparat na mestu posameznega slikovnega elementa pridobi vzorec le za eno komponento: rdečo, zeleno ali modro. Vzorca za preostali dve komponenti sta naknadno določena s pomočjo interpolacije sosednjih vzorčnih vrednosti pripadajočih barvnih komponent. Ta postopek je znan pod imenom interpolacija mreže barvnih filtrov in v digitalno fotografijo vnese prostim očem nevidne korelacije med vzorci iste barvne komponente. Med digitalno obdelavo slike te korelacije oslabijo ali se celo izničijo. S posebnim algoritmom lahko takšno spremembo zaznamo, kar utegne biti znak, da imamo opravka z digitalnim ponaredkom.

- odkrivanje posledic **dvakratnega stiskanja slikovnih podatkov po standardu JPEG** (*double JPEG compression*)

  Verjetnost, da sta pristna digitalna fotografija in na njej osnovan ponaredek shranjena v izgubnem zapisu JPEG, je relativno velika. Upravičenost te domneve potrjujeta tako dejstvo, da večina digitalnih fotoaparatov omogoča shranjevanje posnetkov samo v zapisu JPEG, kot tudi učinkovitost in posledično množična uporaba tega zapisa. Če v pripadajoči digitalni sliki, za katero preverjamo pristnost, najdemo sledove dvakratnega stiskanja slikovnih podatkov, je potrebno sliko obravnavati kot potencialni ponaredek. Posledice dvakratnega stiskanja slikovnih podatkov po standardu JPEG se odražajo v tipičnih spremembah slikovnega histograma, ki jih ni težko zaznati.

  Pri tem seveda ne smemo pozabiti, da so enake posledice možne tudi pri povsem običajnih in legitimnih operacijah, kot je na primer naknadno

zmanjševanje ločljivosti digitalne fotografije, čemur sledi ponovno shranjevanje – najverjetneje spet v zapisu JPEG.

- odkrivanje **podvojenih regij** (*duplicated regions*)

  Običajna operacija pri izdelavi digitalnega ponaredka je odstranjevanje neželenih oseb ali predmetov s fotografije. Pogosto je izvedena kot kopiranje drugih delov slike preko izbranih področij. Pri uspešnih ponaredkih so posledice te operacije prostim očem neopazne, zato si moramo pomagati s posebnimi algoritmi. Eden od njih je osrednja tema te diplomske naloge in je podrobno opisan v naslednjem poglavju. Enostavno povedano gre za uporabo metode **analize glavnih komponent** (*principal component analisys – PCA*) na slikovnih blokih konstantne velikosti. Rezultati te operacije so kasneje ovrednoteni, pri čemer igra glavno vlogo gradnja seznama bločnih parov in njegovo leksikografsko urejanje. Algoritem je dovolj učinkovit, da pod določenimi pogoji odkrije in prostorsko določi podvojene regije tudi v slikah z naknadno dodanim umetnim šumom, v slikah, ki so bile naknadno stisnjene z izgubnim algoritmom za stiskanje in v slikah. ki so bile naknadno filtrirane s filtrom mediana.

- odkrivanje posledic **različnih nelinearnih transformacij svetlosti** (*different luminance non-linearities*)

  Da bi izboljšali zaznavno kakovost digitalnih fotografij, digitalni fotoaparati običajno izvajajo nelinearne transformacije na svetlosti slikovnih elementov. Ponavadi so parametri določene nelinearne transformacije izbrani glede na dinamiko fotoaparata oziroma njegovega svetlobnega tipala ter posnetega prizora in so enaki za vsa področja na fotografiji. Sledovi nelinearnih transformacij z različnimi parametri so zato potencialni znak poneverbe, saj kažejo na uporabo različnih fotoaparatov, na snemanja v različnih svetlobnih pogojih ali na kombinacijo obojega. Obstaja tudi verjetnost, da je neka nelinearna transformacija uporabljena samo na določenem delu slike z namenom prikriti poneverbo.

  Zaradi nelinearnih transformacij svetlosti se v frekvenčni domeni slike pojavijo korelacije. Če so na sliki izvedene različne svetlostne transformacije, so tudi korelacije različne. Te korelacije lahko zaznamo z uporabo metod večspektralne analize, ki temeljijo na cenilkah za določanje **vrednosti gama** (*gamma value*).

- neinformirano ocenjevanje **lokalne variance šuma v ozadju** (*local background noise variance*)

  Vse digitalne fotografije vsebujejo šum. Količina šuma v določeni fotografiji je odvisna od več dejavnikov. Tako šum v fotografijo običajno vnese že samo zajemanje fotografije s fotoaparatom (šum pretvorbe analogno-digitalno) ali stiskanje slikovnih podatkov pri shranjevanju v datoteko. Seveda je šum lahko tudi umetno dodan v fotografijo. S posebnimi cenilkami lahko ocenimo, kakšno je razmerje med signalom in šumom na določenem delu slike. Pri pristnih digitalnih fotografijah sta količina šuma in z njo povezano razmerje med signalom in šumom enaka na celotni sliki.

Po spajanju izsekov več fotografij v isto sliko je jakost signala v različnih delih tako pridobljene slike lahko različna; enako velja tudi za količino šuma. Zato je kot parameter za preverjanje pristnosti digitalnih fotografij bolj kot razmerje med signalom in šumom primerna lokalna varianca količine šuma. Varianca ni odvisna od lokalne jakosti signala. Če je varianca na različnih delih slike znatno različna, je to zanesljiv znak, da je bila fotografija naknadno obdelana.

Opisani postopki se opirajo na različne temelje. Zato lahko povečamo stopnjo zaupanja v končno odločitev o pristnosti digitalne fotografije, če pri preverjanju uporabimo več različnih postopkov in smiselno povežemo rezultate.

1.1.4 NEINFORMIRANI POSTOPKI ZA ODKRIVANJE PODVOJENIH REGIJ

V nadaljevanju besedila se bomo osredotočili na odkrivanje podvojenih regij, zato naštejmo nekaj postopkov, ki so jih raziskovalci predlagali v ta namen:

- postopek na osnovi **afino invariantnih regij** (*affine invariant regions*)

  Ta postopek je v izvirni obliki namenjen **odkrivanju približnih dvojnikov** (*near-duplicate detection*) v množici digitalnih slik, vendar ga je možno prilagoditi za odkrivanje podvojenih regij na sliki. Deluje na osnovi določanja regij, ki so invariante za afino preslikave. Postopek je opisan v [24].

- postopek na osnovi **momentnih funkcionalov, invariantnih na zabrisanje slike** (*blur moment invariants*)

  Podvojene regije odkrijemo z medsebojnim primerjanjem ustreznih slikovnih momentov, ki so izračunani za posamezne slikovne izseke enakih velikosti. Postopek je predlagan v [15], teoretična podlaga zanj pa je opisana v [8].

- postopek s preverjanjem soseščine v **k-razsežnem drevesu** (*k-dimensional tree*)

  Najprej razdelimo preiskovano sliko na bloke; pri tem uporabimo princip prekrivajočih se slikovnih izsekov enake velikosti. Iz blokov sestavimo k-razsežno drevo, v katerem potem na podlagi ujemanja preiskujemo bližnje soseščine za pare blokov z enakimi ali podobnimi vzorci slikovnih elementov. Opis postopka najdemo v [11].

- postopek na osnovi **analize glavnih komponent**

  Ta postopek je obravnavan v tej diplomski nalogi.

- postopek z uporabo **diskretne valčne transformacije** (*discrete wavelet transform*)

  Najprej razdelimo preiskovano sliko na bloke po principu prekrivajočih se slikovnih izsekov. Na vsakem bloku izvedemo dvostopenjsko diskretno valčno transformacijo in iz dobljenih (sedmih) komponent sestavimo vektor. Nato seznam pripadajočih vektorjev leksikografsko uredimo in nadaljujemo s koraki, ki so enaki tistim v postopku na osnovi analize glavnih komponent. Celoten

postopek je opisan v [2].

- postopek z uporabo diskretne valčne transformacije in **singularnega razcepa** (*singluar value decomposition*)

  Izračunamo diskretno valčno transformacijo celotne preiskovane slike. Potem po principu prekrivajočih se slikovnih izsekov razdelimo njeno nizkofrekvenčno valčno komponento na bloke, ki jih nadalje poenostavimo z uporabo singularnega razcepa. Nadaljevanje postopka je podobno postopku na osnovi analize glavnih komponent. Ta postopek je opisan v [9].

1.2 CILJI DIPLOMSKE NALOGE

V okviru izdelave diplomske naloge smo si zadali naslednje cilje:

1. Izvesti algoritem za odkrivanje podvojenih regij, opisan v [20], v programskem jeziku java.

   Izdelana aplikacija mora podpirati tako interaktivno kot paketno preverjanje slik. V obeh režimih dela mora na podlagi ugotovljenih značilnosti določiti, ali je bolj verjetno, da je posamezna preiskana digitalna fotografija pristna (negativni primer) ali prirejena oziroma ponarejena (pozitivni primer). V pozitivnih primerih mora označiti območja, ki sovpadajo s podvojenimi regijami in njihovimi podvojitvami.

2. Preveriti algoritem glede pravilnosti obravnave oziroma ločevanja med pristnimi in ponarejenimi slikami z uporabo množice preizkusnih slik.

   V preizkusni množici naj bodo pristne slike in ponaredki. Množica naj poleg pravih fotografij vsebuje tudi nekaj umetno izdelanih slik, ki služijo za ponazoritev principov delovanja algoritma.

3. Preveriti odpornost algoritma proti izbranim postopkom digitalne obdelave, ki stremijo k preslepitvi algoritma.

   Preizkusna množica naj vsebuje samo pozitivne primere, od katerih so bili nekateri po osnovni digitalni slikovni obdelavi naknadno podvrženi različnim operacijam, na primer dodajanju umetnega šuma, stiskanju z izgubnim algoritmom za stiskanje ali filtriranju s filtrom mediana.

4. Opisati možne razširitve in nadaljnje izboljšave algoritma.

1.3 PREGLED VSEBINE

V prvem poglavju smo bralca uvedli v naslovno temo: od digitalnih vsebin, prek problematike poneverb in načinov odkrivanja digitalnih ponaredkov smo prešli do razlage neinformiranih postopkov s poudarkom na odkrivanju podvojenih regij.

V drugem poglavju bomo najprej utemeljili izbiro zasnove za algoritem DRD, v

nadaljevanju pa podrobno opisali vse faze algoritma, ocenili njegovi računski zahtevnosti ter podali formalen opis njegovih korakov. V tretjem poglavju bomo opisali preizkusno množico, navedli priporočene vrednosti parametrov odkrivanja in nakazali njihov vpliv na rezultate. Predstavili bomo tako primere pravilnih in napačnih obravnav pristnih in ponarejenih slik, kot tudi primere dveh vrst postopkov digitalne obdelave: tistih, ki onesposobijo algoritem, in tistih, ki ga ne. Poglavje bomo zaokrožili z obravnavo učinkovitosti algoritma. V četrtem poglavju bomo podali sklepne ugotovitve in nakazali možne smeri nadaljnjega raziskovanja za razširitev in izboljšavo algoritma. V dodatkih bomo predstavili metodo analize glavnih komponent in raziskovalca informirali o nekaterih podrobnostih naše izvedbe algoritma DRD.

Poglavje 2

Odkrivanje podvojenih regij z algoritmom DRD

Podvajanje regij oziroma delov digitalnih slik gotovo sodi med najbolj pogoste operacije manipuliranja z digitalnimi fotografijami. Uporablja se za odstranjevanje neželenih ljudi, živali ali predmetov iz prikazanega prizora. Če je operacija izpeljana profesionalno, za seboj ne pusti sledov, ki bi jih lahko ugotovili zgolj z opazovanjem.

Predmet obravnave te diplomske naloge je neinformiran postopek oziroma algoritem, ki samodejno odkriva podvojene regije v digitalnih rastrskih slikah in določa njihove lege. V nadaljevanju besedila ga bomo označevali z izrazom **detektor podvojenih regij** (*duplicate region detector*) oziroma s kratico DRD.

2.1 ZAČETNI PREMISLEKI

Algoritem DRD se uspešno izogne dvema ovirama, ki znatno otežujeta odkrivanje podvojenih regij:

- preveliki časovni zahtevnosti in
- občutljivosti na naknadno digitalno obdelavo slike po podvajanju regij.

Na prvo omenjeno oviro naletimo, če želimo podvojene regije odkrivati z **izčrpnim preiskovanjem** (*exhaustive search*). Pri uporabi takšnega postopka moramo v digitalni rastrski sliki medsebojno primerjati slikovne bloke več kot ene oziroma v najslabšem primeru kar vseh možnih velikosti. Časovna zahtevnost take primerjave narašča eksponentno s številom slikovnih elementov, zato je njena učinkovitost za praktično uporabo odločno premajhna. Premostitev ovire leži v primerjavi slikovnih blokov ene same velikosti, ki jo ocenimo kot najprimernejšo. Izkušnje kažejo na to, da morajo biti bloki znatno manjši od podvojenih regij, če naj bo postopek uspešen.

Na drugo oviro naletimo, če želimo izvesti primerjavo slikovnih blokov neposredno,

brez predhodne prilagoditve njihove vsebine. Po tej metodi v okviru primerjave določenega para blokov primerjamo kar izvirni vsebini obeh pripadajočih slikovnih izsekov. Primerjava ne da želenih rezultatov, če so bili na digitalni sliki po podvajanju regij uporabljeni postopki, ki stremijo k prikritju poneverbe slike. Primera takih postopkov sta dodajanje umetnega šuma in uporaba izgubnih algoritmov za stiskanje. Rešitev je primerjava **poenostavljenih predstavitev** (*reduced representations*) slikovnih blokov.

Algoritem DRD spada v skupino neinformiranih postopkov. V našem primeru to pomeni, da za delovanje potrebuje samo digitalno sliko z domnevno prirejeno oziroma ponarejeno fotografijo. Algoritem prav tako ne zahteva dodatnih informacij o obliki in velikosti podvojenih regij. Velikost blokov, na katere razdelimo sliko, je vnaprej določena in se med izvajanjem postopka ne spreminja.

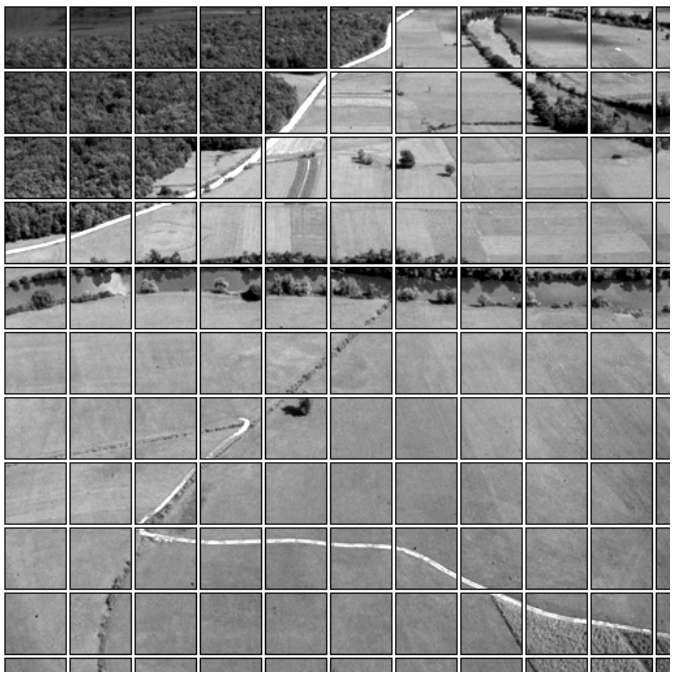
2.2 DELOVANJE ALGORITMA DRD

Tako vhodni kot izhodni podatki algoritma DRD so sivinske rastrske slike. Algoritem na vhodu pričakuje sliko, na kateri bo odkrival morebitno podvojena območja. Na izhod pa dá sliko podvojitev, na kateri so označena podvojena območja. Če takih območij ne najde, pusti sliko podvojitev prazno.

Ko algoritem prebere vhodno sliko, izvede šest glavnih korakov. Opisani so v nadaljevanju.

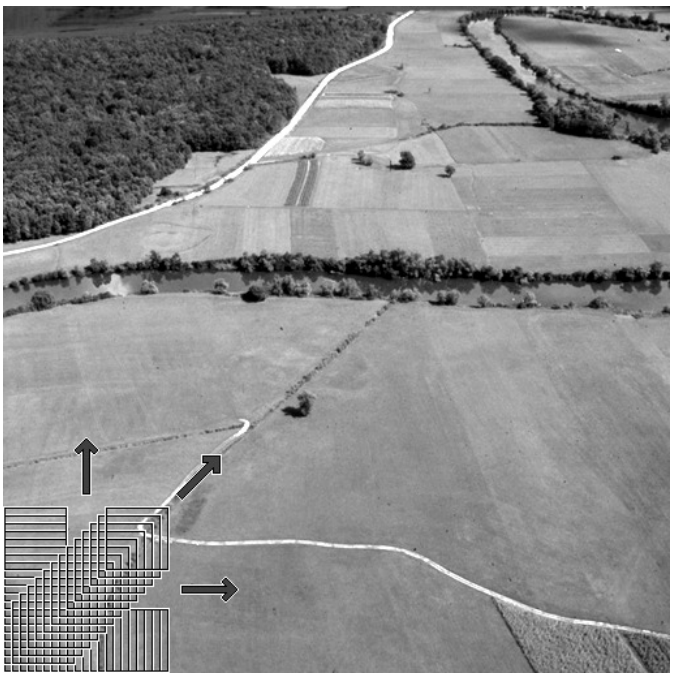
1. **Razdeli sliko na kvadratne bloke enake velikosti.**

   Za razdelitev slike na bloke ne uporabimo "naivne" metode, s katero bi sliko razkosali kar po navidezni mreži, ki poteka po robovih slikovnih izsekov. Metodo prikazuje slika 2.1. Uporaba te metode bi namreč omejila uspešnost algoritma DRD le na slike, v katerih bi bile podvojene regije enako oddaljene od mrežnih oglišč. Algoritem bi bil odvisen od naključja in zato uspešen le v redkih robnih primerih ter v praksi povsem neuporaben.



Slika 2.1: "Naivna" razdelitev slike na bloke.

Zato, da izničimo vpliv lege podvojenih regij na uspešnost odkrivanja, bloke iz slike pridobimo z metodo prekrivajočih se slikovnih izsekov. Metoda je prikazana na sliki 2.2.



Slika 2.2: Razdelitev slike, ki omogoča odkrivanje podvojenih regij ne glede na lego.

Pri tej metodi so sosednji slikovni izseki zamaknjeni za en slikovni element v vodoravni ali navpični smeri. Z uporabo te metode pridobimo precej več blokov kot pri "naivni" metodi; posledično s tem iz slike izluščimo bistveno več informacij, ki nas zanimajo.

Izbira oblike blokov se zdi intuitivna, vendar ima svojo utemeljitev. S kvadratno obliko namesto pravokotne algoritem pri odkrivanju podvojenih regij z enako natančnostjo obravnava obe smeri na sliki: vodoravno in navpično. Z uporabo pravokotnih blokov bi postal algoritem pristranski, saj bi z večjo natančnostjo obravnaval tisto razsežnost slike, ki bi ustrezala daljši stranici pravokotnika. Prav tako je izbira take poravnanosti blokov, da so njihove stranice vzporedne s stranicami slike, zaradi enostavnosti najboljša. Pri pristopu z uporabo zasukanih blokov bi bil prvi in verjetno ne edini problem ustrezna predstavitev slikovnih elementov tistih delov blokov, ki bi segali čez rob slike.

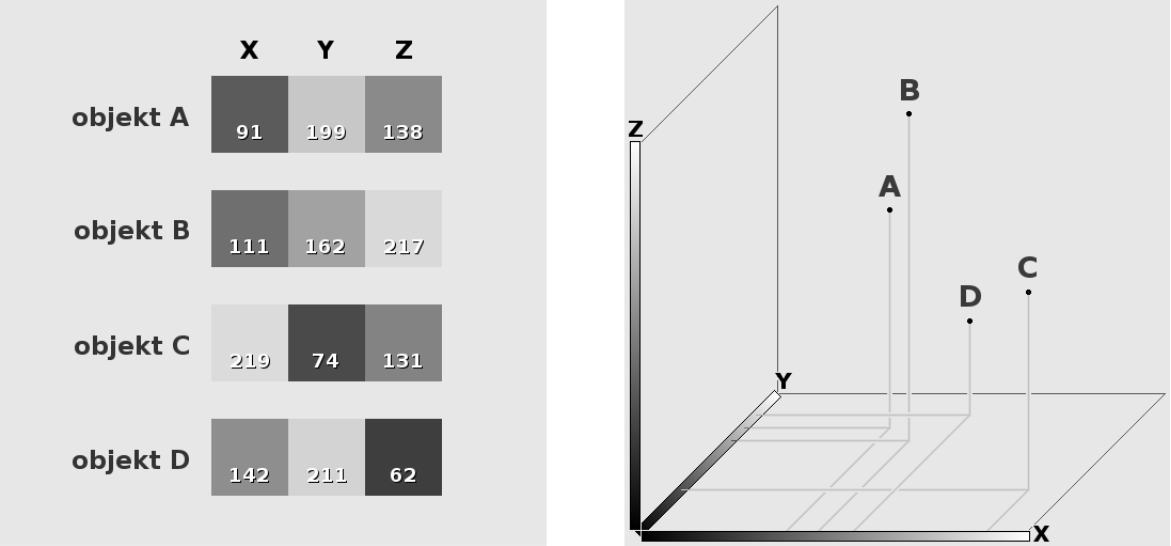
2. Z uporabo metode PCA določi poenostavljene predstavitve blokov.

Avtorji profesionalnih digitalnih ponaredkov vložijo nemalo napora v brisanje sledov ponarejanja; le na ta način lahko dosežejo zadostno prepričljivost. Če je v procesu digitalne obdelave izvedel podvajanje regij, ponarejevalec običajno doda celotni sliki ali njenim delom umetni šum ali pa izvede katero drugo operacijo, ki spremeni visokofrekvenčne komponente slike. S tem posegom doseže, da podvojene regije in njihove podvojitve niso več popolnoma enake. V nasprotnem primeru bi bili sledovi podvajanja regij očitni že na prvi pogled in zato zlahka ugotovljivi.

Zaradi naknadnih operacij, ki vpeljejo razlike med podvojenimi regijami in njihovimi podvojitvami, v prvem koraku algoritma DRD pridobljeni bloki niso primerni za neposredno primerjavo. Potrebujemo primerjavo, ki bi zanemarila podrobnosti in s tem razlike v visokofrekvenčnih komponentah. Tako primerjavo pa lahko izvedemo, če namesto izvirnih slikovnih blokov medsebojno primerjamo njihove poenostavljene predstavitve.

Poenostavljena predstavitev (*reduced representation*) pomeni takšno predstavitev določenega podatkovnega objekta, ki vsebuje le njegove bistvene značilnosti, ne pa tudi njegovih podrobnosti. Eden od načinov določanja poenostavljene predstavitve je uporaba metode analize glavnih komponent (PCA) [22]. Z uporabo metode PCA bomo najprej določili poenostavljene predstavitve blokov, pridobljenih v prvem koraku algoritma DRD.

Za metodo PCA niso bloki nič drugega kot točke v prostoru, čigar razsežnost ustreza številu slikovnih elementov v blokih. Vsak slikovni element v nekem bloku predstavlja eno koordinato bloku pripadajoče točke. Vrednosti posameznih koordinat točke so določene s svetlostjo pripadajočih slikovnih elementov. Bolj kot sta si dva bloka podobna, bliže v prostoru ležita točki, s katerima sta predstavljena. Slika 2.3 ponazarja predstavitev blokov s tremi slikovnimi elementi v prostoru.



Slika 2.3: Enostavni podatkovni objekti in njihova predstavitev v prostoru.

Metoda PCA določi poenostavljene predstavitve blokov tako, da preslika začetne predstavitve blokov v višjerazsežnem prostoru v ustrezne predstavitve v nižjerazsežnem prostoru. Nižjerazsežni prostor je določen z lastnimi vektorji matrike, ki jo sestavljajo mere kovarianc med posameznimi koordinatami podatkov v višjerazsežnem prostoru. Podrobnejši opis metode PCA je podan v dodatku A.

3. Zgradi seznam poenostavljenih predstavitev blokov in ga leksikografsko uredi.

Po tem, ko smo določili poenostavljene predstavitve blokov, lahko začnemo s postopkom primerjanja. V ta namen poenostavljene predstavitve najprej zapišemo v obliki vektorjev, s katerimi nato zgradimo seznam. V seznam poleg same predstavitve posemeznega bloka shranimo tudi položaj pripadajočega slikovnega izseka na preiskovani sliki. Položaj je določen z odmikom skrajno levega zgornjega slikovnega elementa izseka od skrajno levega zgornjega slikovnega elementa celotne slike.

Na tem mestu naj spomnimo, da tudi za poenostavljene predstavitve blokov velja ista zakonitost, kot za izvirne bloke: čim bolj podobni sta si dve predstavitvi, tem bliže v prostoru poenostavljenih predstavitev ležita točki, ki sta določeni s pripadajočima vektorjema. V enačbah 2.1 sta $\vec{r_k}$ in $\vec{r_l}$ oznaki vektorjev, ki ustrezata izvirnima bokoma, $\vec{a_k}$ in $\vec{a_l}$ predstavljata vektorja pripadajočih poenostavljenih predstavitev, vrednosti u in v so vektorske komponente, oznake $\vec{I}$ pa ortonormirani vektorji, ki predstavljajo bazo prostora.

$$\begin{aligned} \vec{r_k} &= \sum_b u_b \vec{I_b} \quad , \quad \vec{r_l} = \sum_b v_b \vec{I_b} \\ \vec{a_k} &= \sum_t u_t \vec{I_t} \quad , \quad \vec{a_l} = \sum_t v_t \vec{I_t} \end{aligned} \tag{2.1}$$

$$(\vec{r}_k - \vec{r}_l)^2 = \sum_b (u_b - v_b)^2 \approx \sum_t (u_t - v_t)^2$$

Urejanje seznama vektorjev s poenostavljenimi predstavitvami izvedemo z upoštevanjem leksikografskega vrstnega reda. Pri tem za **ureditveni ključ** (*sortkey*) zaporedoma uporabimo posamezne koordinate vektorjev, v katerih so zapisane poenostavljene predstavitve blokov. Na ta način bodo vektorji blokov, katerih poenostavljene predstavitve so podobne, razvrščeni blizu skupaj.

4. Določi množico kandidatov.

V tem koraku bomo sestavili množico kandidatov. Z izrazom *kandidat* bomo označili vsak par blokov preiskovane slike, za katera predpostavljamo, da utegneta vsebovati enaki območji: en blok podvojeno regijo, drugi blok pa njeno podvojitev.

Vhodni podatki za ta korak so urejen seznam poenostavljenih predstavitev blokov. V tem seznamu vektorji podobnih predstavitev ležijo blizu skupaj, vektorji morebitnih enakih predstavitev pa so sosednji. Za določanje kandidatov uporabimo naslednje merilo: če je razlika med indeksoma dveh vektorjev v urejenem seznamu poenostavljenih predstavitev manjša od vnaprej predpisanega praga H_n, potem bloka, ki pripadata tema predstavitvama, predstavljata enega kandidata. V enačbi 2.2 pomeni C množico kandidatov, $\vec{s}_i$ in $\vec{s}_j$ pa sta vektorja poenostavljenih predstavitev nekega bločnega para.

$$\mathrm{C} = \{(\vec{s}_i, \vec{s}_j),\ i \neq j \wedge |i-j| < H_n\} \tag{2.2}$$

Celotno množico kandidatov dobimo tako, da opisano merilo preverimo za vsak možen par vektorjev v seznamu. Uporaba praga nam zagotavlja, da moč te množice ostane v še obvladljivih mejah.

Množico kandidatov predstavimo s posebnim seznamom, v katerem vsakega kandidata predstavimo s položajema obeh pripadajočih slikovnih izsekov na preiskovani sliki.

5. Izloči neprave kandidate.

V množici kandidatov se lahko znajdejo tudi pari vektorjev, katerih pripadajoča bloka na preiskovani sliki ne vsebujeta podobnih območij zaradi posledic izvedene digitalne obdelave, pač pa zaradi drugih razlogov. To seveda niso pravi kandidati in jih moramo zato izločiti.

Nepravi kandidat je lahko:

- par slikovnih izsekov, ki vsebujeta del dveh po naključju podobnih območij na preiskovani sliki (nepravi kandidat *prvega tipa*) ali
- par slikovnih izsekov s premajhnim medsebojnim odmikom; torej dveh izsekov, ki vsebujeta del istega območja na preiskovani sliki (nepravi kandidat *drugega tipa*).

Primere nepravih kandidatov prvega oziroma drugega tipa prikazujeta sliki 2.4 in 2.5.

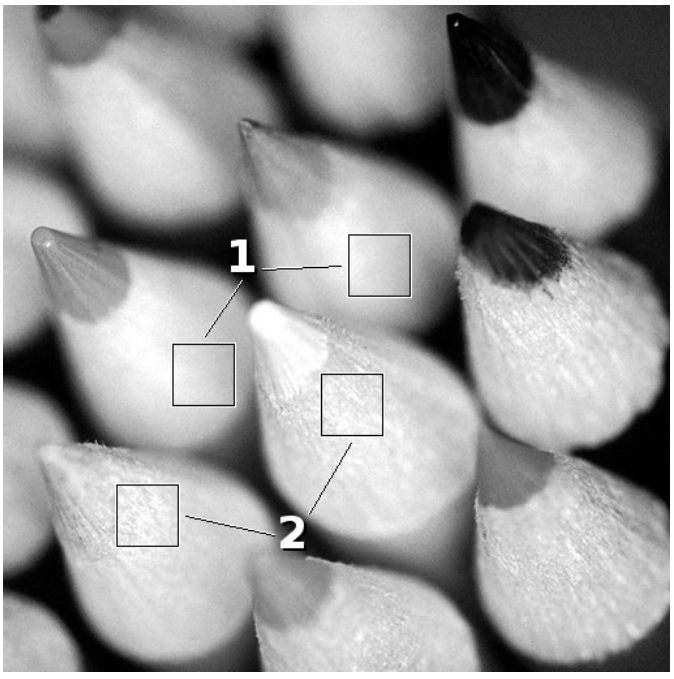
Izločitev nepravih kandidatov iz množice izvedemo z upoštevanjem dveh meril. S posameznim merilom izločimo vse neprave kandidate enega tipa.

Neprave kandidate *prvega tipa* izločimo tako, da iz množice odstranimo pare, za katere je pogostnost velikosti medsebojnega odmika njim pripadajočih slikovnih izsekov preko celotne množice manjša od vnaprej predpisanega praga H_f. V formuli 2.3 pomeni C množico kandidatov, $\vec{s_i}$ in $\vec{s_j}$ sta vektorja poenostavljenih predstavitev nekega bločnega para, $\mathrm{Fx_a}$ in $\mathrm{Fy_b}$ predstavljata medsebojni odmik v vodoravni oziroma navpični smeri, $\mathrm{M_{a,b}}$ pa je množica parov vektorjev poenostavljenih predstavitev blokov, ki imajo enak medsebojni odmik.

$$\mathrm{M_{a,b}} = \{(\vec{s_i}, \vec{s_j}),\ (x_i - x_j = \mathrm{Fx_a}) \wedge (y_i - y_j = \mathrm{Fy_b})\} \wedge \left|\mathrm{M_{a,b}}\right| < H_f \Rightarrow (\vec{s_i}, \vec{s_j}) \notin \mathrm{C} \tag{2.3}$$

Ob predpostavki, da so bloki znatno manjši od podvojenih regij, neko podvojeno regijo in njeno podvojitev sestavljata dve podobni skupini blokov. Pri tem mora vsaka skupina vsebovati zadostno število blokov in se torej mora isti medsebojni odmik pojaviti pri zadostnem številu kandidatov.

V posebnih primerih algoritem DRD iz množice kandidatov ne uspe izločiti nepravih kandidatov prvega tipa. To se zgodi takrat, ko preiskovana slika vsebuje dve po naključju zelo podobni območji, ki sta precej večji od blokov s katerimi smo razdelili sliko. Isti medsebojni odmik se tedaj pojavi pri velikem številu kandidatov, njegova pogostnost pa utegne preseči predpisani prag. Algoritem DRD zato takšni območji napačno obravnava kot podvojeno regijo in njeno podvojitev.



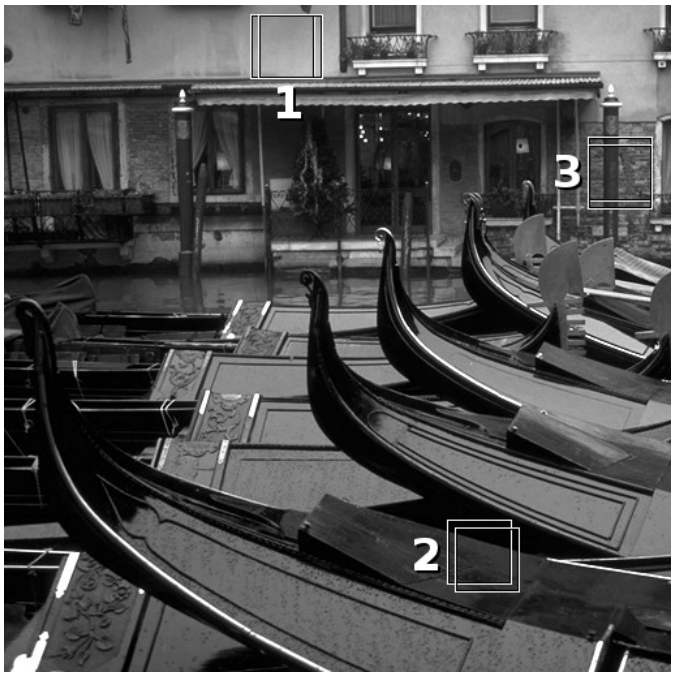
Slika 2.4: Primera nepravih kandidatov prvega tipa.

Neprave kandidate *drugega tipa* izločimo tako, da iz množice odstranimo pare, za katere velikost medsebojnega odmika pripadajočih slikovnih izsekov ne dosega vnaprej predpisanega praga H_d. V formuli 2.3 sta $(x_i,\ y_i)$ in $(x_j,\ y_j)$ dvodimenzionalna odmika dveh slikovnih izsekov od levega zgornjega kota slike. oznake $\vec{s_i}$, $\vec{s_j}$ in C pa imajo enak pomen kot v enačbi 2.2.

$$\sqrt{(x_i - x_j)(y_i - y_j)} < H_d \Rightarrow (\vec{s_i}, \vec{s_j}) \notin \mathrm{C} \tag{2.4}$$

Slikovna izseka, katerih medsebojni odmik je na primer manjši od dolžine stranice bloka, zajemata del istega območja preiskovane slike. Izseka zato ne predstavljata pravega kandidata.

Pod pogojem, da je vrednost predpisanega praga ustrezno določena, je algoritem DRD pri izločanju kandidatov drugega tipa vedno uspešen.



Slika 2.5: Primeri nepravih kandidatov drugega tipa.

6. Izriši sliko podvojitev.

V prejšnjem koraku smo po izločitvi nepravih kandidatov pridobili množico parov blokov preiskovane slike, ki določajo podvojena območja digitalne slike. Sedaj bomo izrisali sliko podvojitev.

Izhodiščna slika za postopek izrisa je prazna sivinska slika črne barve. Vsak par blokov označimo na sliki podvojitev tako, da središčni element prvemu bloku pripadajočega slikovnega izseka obarvamo z enim sivim odtenkom, središčni element drugemu bloku pripadajočega slikovnega izseka pa z drugim sivim odtenkom. Ko je označitev končana za celotno množico bločnih parov. so na sliki

podvojitev izrisana območja, katerih lege in oblike sovpadajo s podvojenimi regijami in njihovimi podvojitvami.

Slika 2.6: Primer slike podvojitev.

Slika 2.6 prikazuje tipično sliko podvojitev. Na njej takoj opazimo številne osamljene slikovne elemente, ki nas nekako motijo. Dobljena slika podvojitev ima dve očitni pomanjkljivosti:

1. Sivi slikovni elementi na sicer neoznačenih območjih.

   Ti elementi so napačno označeni zato, ker so kljub izločitvi nepravih kandidatov prvega in drugega tipa v množici kandidatov ostali še nadaljnji nepravi kandidati. Na ta način je bil ohranjen vsak blok, za katerega velja naslednje:

   - na sliki obstaja njemu podoben blok, medsebojni odmik obeh pripadajočih slikovnih izsekov pa presega prag H_d. Hkrati sta si njuni poenostavljeni predstavitvi dovolj podobni, da pripadajoča vektorja v urejenem seznamu pripadata isti soseščini. Velikost soseščine določa vrednost H_n.

2. Črni slikovni elementi znotraj mejá sicer označenih območij.

   Do napačne neoznačitve teh elementov pride zato, ker so bili iz množice kandidatov poleg nepravih izločeni tudi pravi kandidati. Izločen je bil vsak blok, za katerega velja naslednje:

   - na preiskovani sliki obstaja njegova podvojitev, medsebojni odmik obeh pripadajočih slikovnih izsekov pa presega prag H_d in
   - na sliki obstaja njemu podoben blok, ki je od njega oddaljen za manj kot H_d.

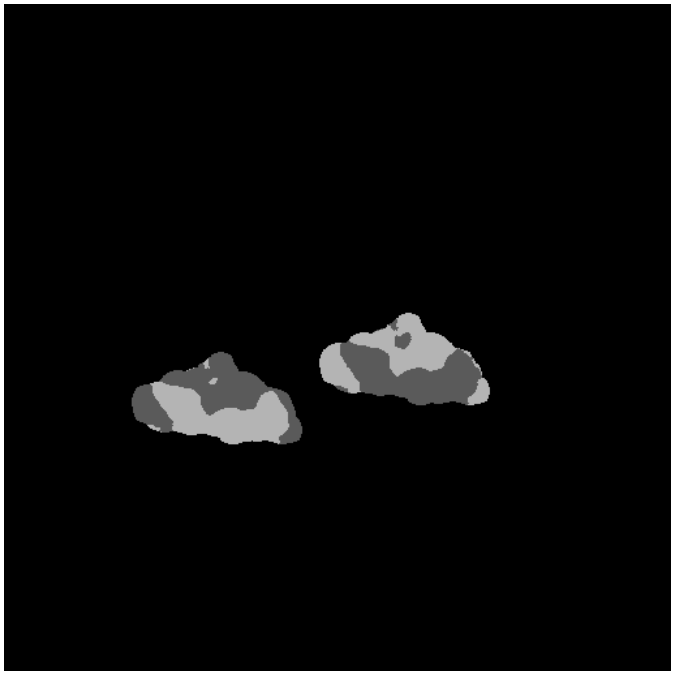
Hkrati sta si njuni poenostavljeni predstavitvi dovolj podobni, da pripadajoča vektorja v urejenem seznamu pripadata isti soseščini.

Algoritem zato takšne bloke pomotoma obravnava kot neprave kandidate drugega tipa.

Da bi povečali kakovost slike podvojitev moramo obe pomanjkljivosti odpraviti. Za začetek si pomagamo z ugotovitvijo, da dobljena slika podvojitev že vsebuje zadostno količino informacij, ki nas zanimajo. Postopek izboljšave je torej enostavnejši, če se ne vračamo več k množici kandidatov, pač pa obstoječo sliko v nekaj korakih digitalno obdelamo[4].

Za odpravo prve pomanjkljivosti sliko podvojitev najprej podvržemo morfološki operaciji **razjedanja** oziroma **erozije** (*erosion*) in nato še morfološki operaciji **razširjanja** oziroma **dilatacije** (*dilation*). Razjedanje odstrani osamljene označene slikovne elemente v neoznačenih okolicah. Operacija pa ima tudi stranski učinek: zaradi razjedanja robov označenih območij se njihova velikost zmanjša. Razširjanje spet ustrezno poveča območja in s tem izniči neželeni učinek. Na dobljeni sliki nato odpravimo še drugo pomanjkljivost in sicer z izvedbo istih operacij v obratnem vrstnem redu; najprej izvedemo razširjanje, potem pa še razjedanje. Razširjanje zapolni vrzeli v sicer označenih okolicah, pri tem pa ima podoben stranski učinek. Zaradi razširjanja zunanjih robov označenih območij se namreč njihova velikost poveča. Razjedanje poskrbi za izničenje tega učinka.

Rezultat tega koraka algoritma DRD je prikazan na sliki 2.7.



Slika 2.7: Slika podvojitev po odpravi obeh pomanjkljivosti.

4 Opisani koraki digitalne obdelave imajo podoben učinek, kot bi ga imela predhodno izločanje neupravičeno ohranjenih nepravih kandidatov iz množice kandidatov in vračanje neupravičeno izločenih pravih kandidatov vanjo.

S tem korakom algoritma DRD je odkrivanje podvojenih regij končano. Slika 2.8 prikazuje izvirno sliko, za katero je algoritem DRD določil sliko podvojitev na sliki 2.7.

Slika 2.8: Izvirna preiskana slika, za katero je algoritem DRD določil sliko podvojitev.

2.3 RAČUNSKA ZAHTEVNOST ALGORITMA DRD

Če naj določen algoritem ovrednotimo kot primeren za uporabo v praksi, potem morata biti sprejemljivi obe njegovi računski zahtevnosti: časovna in prostorska. V tem podpoglavju bomo pokazali, da za algoritem DRD to drži.

Ocenili bomo računski zahtevnosti algoritma DRD za preiskavo ene vhodne slike. Parameter N, ki v ocenah algoritemskih zahtevnosti pomeni število vhodnih podatkov, v tem primeru pomeni število slikovnih elementov v posamezni preiskovani sliki.

2.3.1 ČASOVNA ZAHTEVNOST

Za oceno časovne zahtevnosti najprej analizirajmo algoritem, nato pa na podlagi ocen časovnih zahtevnosti posameznih korakov ocenimo časovno zahtevnost celotnega algoritma.

Ocene časovnih zahtevnosti korakov algoritma DRD so podane v naslednjem podpoglavju. Glavni delež k celotni časovni zahtevnosti prispevata dva koraka:

- leksikografskego urejanje seznama vektorjev s poenostavljenimi predstavitvami blokov (korak A) in
- odstranjevanje nepravih kandidatov prvega tipa iz množice kandidatov (korak B).

Doprinosi ostalih korakov so reda O(N) ali O(1).

Časovna zahtevnost metode PCA v primerjavi z omenjenima korakoma ni kritična, saj je ocenjena z O(1). Razlog za tako ugodno oceno je pravilna izbira postopka, ki izvaja PCA, v odvisnosti od dveh parametrov: števila slikovnih izsekov (N_b) in števila slikovnih elementov v enem izseku (b). Velikost blokov (b) je privzeta in ni odvisna od velikosti vhodne slike, zato jo obravnavamo kot konstanto.

Med prispevki k časovni zahtevnosti koraka A dominira časovna zahtevnost algoritma **urejanja s porazdelitvami** (*quicksort*) za leksikografsko urejanje, ki je v pričakovanem primeru (povprečju) enaka O($N \log(N)$), v najslabšem primeru pa O(N^2).

Časovno zahtevnost koraka B določata gradnja dvojiškega iskalnega drevesa in iskanje v njem. Razpršenost vhodnih podatkov (dvorazsežnih medsebojnih odmikov obeh slikovnih izsekov istega kandidata), ki jih uporabimo pri gradnji dvojiškega iskalnega drevesa, je velika. To nam zagotavlja, da se drevo nikoli ne izrodi do te mere, da bi se časovna zahtevnost pomembno povečala. Pri neizrojenih drevesih je časovna zahtevnost vstavljanja N elementov kar O($N \log(N)$), enako časovno zahtevnost pa ima tudi iskanje N elementov v drevesu.

2.3.2 PROSTORSKA ZAHTEVNOST

Po analizi korakov algoritma DRD ni težko pokazati, da je njegova prostorska zahtevnost enaka O(N).

Glede na ocenjeni časovni in prostorski zahtevnosti algoritma DRD lahko torej zaključimo, da je algoritem povsem primeren za uporabo v praksi. Za razred manjša računska zahtevnost v primerjavi z odkrivanjem podvojenih regij z izčrpnim preiskovanjem predstavlja bistveno izboljšavo.

2.4 FORMALEN OPIS KORAKOV ALGORITMA DRD

V tem podpoglavju bomo formalno opisali korake, ki jih izvaja algoritem DRD. Poleg opisa vsakega koraka bomo podali ocene obeh njegovih računskih zahtevnosti. Pomen oznak je sledeč:

T, S – pričakovana (povprečna) časovna oziroma prostorska zahtevnost,

T_W, S_W – časovna oziroma prostorska zahtevnost v najslabšem primeru.

Algoritem DRD sestavljajo naslednji koraki:

1. Parametru N priredi vrednost, ki ustreza številu slikovnih elementov v sivinski rasterski sliki.

   $\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(1)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(1)$.

2. Nastavi vrednost parametrov za odkrivanje podvojenih regij:

   - Parametru b priredi število slikovnih elementov v bloku.

     Bloki so kvadratne oblike. Dolžina njihove stranice je enaka $\sqrt{b}$.

     Bloki ustrezajo slikovnim izsekom, ki jih dobimo z zamikanjem v vodoravni oziroma navpični smeri za en slikovni element. Za skupno število dobljenih blokov N_b zato velja zveza:

     $$N_b = (\sqrt{(N)} - \sqrt{(b)} + 1)^2 \tag{2.5}$$

   - Parametru ε priredi **delež skupne variance** (*cumulative percent variance – CPV*) v vhodnih podatkih, ki naj ga metoda PCA zanemari.

     Parameter ε neposredno vpliva na določitev razsežnosti prostora poenostavljenih predstavitev, natančno pa ta prostor določajo izbrani lastni vektorji kovariančne matrike.

   - Parametru Q priredi število kvantizacijskih stopenj.

     Pri običajnih sivinskih slikah z barvno globino 8 bitov je število kvantizacijskih stopenj enako 256, saj je v teh slikah prav toliko možnih svetlostnih nivojev sive barve.

   - Parametru H_n priredi natančno zgornjo mejo medindeksne razlike parov vektorjev v leksikografsko urejenem seznamu vektorjev (matriki), ki določajo množico kandidatov.

     Vsak vektor v tem seznamu vsebuje zapis poenostavljene predstavitve enega bloka.

   - Parametru H_f priredi prag pogostnosti enakih medbločnih odmikov.

   - Parametru H_d priredi prag velikosti medbločnih odmikov.

   $\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(1)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(1)$.

3. Z uporabo metode PCA določi poenostavljeno, D_t-razsežno predstavitev $\vec{a}_i$, i=1,...,N_b vsakega bloka, ki je bil prvotno zastopan z b slikovnimi elementi. Vrednost D_t je izbrana tako, da zadošča enačbi:

   $$1 - \varepsilon = \frac{\sum_{i=1}^{D_t} \lambda_i}{\sum_{i=1}^{b} \lambda_i} \tag{2.6}$$

   Vrednosti λ_i so lastne vrednosti kovariančne matrike, ki jih izračuna metoda PCA, in so enake variancam posameznih podatkovnih komponent. Razmerje na desni

strani enačbe je delež skupne variance.

$\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(N)$.

4. Sestavi matriko **Q** velikosti $N_b \times b$, katere vrstice so vektorji s poenostavljenimi predstavitvami blokov, po komponentah kvantizirani po naslednji formuli:

$$\vec{q}_i = \lfloor \vec{a}_i / Q \rfloor \tag{2.7}$$

$\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(N)$.

5. V matriki **Q** leksikografsko uredi vrstice in dobljeno matriko označi s simbolom **S**. Vrstice matrike **S** označi s simboli $\vec{s}_i$. Koordinate skrajno levih zgornjih elementov slikovnih izsekov, katerih pripadajoči bloki so predstavljeni s simboli $\vec{s}_i$, označi z (x_i, y_i).

$\mathrm{T}(N) = \mathrm{O}(N \log(N))$, $\mathrm{T_W}(N) = \mathrm{O}(N^2)$, $\mathrm{S}(N) = \mathrm{O}(\log(N))$, $\mathrm{S_W}(N) = \mathrm{O}(N)$.

6. Za vsak par vrstic $\vec{s}_i$ in $\vec{s}_j$ v matriki **S**, za kateri velja zveza $|i - j| < H_n$, dodaj koordinatni par (x_i, y_i) in (x_j, y_j) na poseben seznam.

$\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(N)$.

7. Za vsak koordinatni par na seznamu izračunaj dvorazsežni medsebojni odmik pripadajočih slikovnih izsekov, ki je definiran kot:

$$\begin{gathered}(x_i - x_j,\ y_i - y_j),\ \text{če velja } x_i - x_j > 0 \\ (x_j - x_i,\ y_i - y_j),\ \text{če velja } x_i - x_j < 0 \\ (0, |y_i - y_j|),\ \text{če velja } x_i = x_j\end{gathered} \tag{2.8}$$

Izračunan odmik dodaj v dvojiško iskalno drevo, ki hrani različne vrednosti odmikov, ugotovljenih za vse koordinatne pare (kandidate).

$\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N \log(N))$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(N)$.

Najslabši primer (gradnja izrojenega dvojiškega iskalnega drevesa in iskanja v takšnem drevesu), za katerega velja $\mathrm{T_W}(N) = \mathrm{O}(N^2)$, se z algoritmom DRD ne more primeriti. Izsledki preizkusov namreč kažejo, da so vhodni podatki za gradnjo drevesa izrazito leksikografsko neurejeni[5], kar zagotavlja zadostno uravnoteženost drevesa.

5 Zaporedni slikovni izseki tipične digitalne fotografije so glede na vsebino (svetlost slikovnih elementov) leksikografsko neurejeni. V splošnem se neurejenost ohrani pri poenostavljanju predstavitev njim pripadajočih blokov in zato velja tudi za vrstice matrike **Q** (korak 5). Vrstice te matrike pa so urejene po nekem drugem ključu: dvodimenzionalnem odmiku pripadajočih slikovnih izsekov od levega zgornjega kota slike. Za vrstice matrike **S** (korak 6) velja ravno obratno: urejene so po vsebini, neurejene pa po odmiku pripadajočih slikovnih izsekov od kota slike. Slednja neurejenost se ohrani tudi pri medsebojnih odmikih pridobljenih parov blokov, ki predstavljajo vhodne podatke za gradnjo dvojiškega iskalnega drevesa.

8. S seznama odstrani vse koordinatne pare, čigar vrednost dvorazsežnega medsebojnega odmika se v celotnem seznamu pojavi manj kot H_f-krat.

   Ta korak zagotavlja, da k sliki podvojitev ne prispevajo posamezni pari blokov, katerih poenostavljene predstavitve so si po naključju podobne. Ob predpostavki, da so bloki znatno manjši od podvojenih regij, morata neko podvojeno regijo in njeno podvojitev nujno sestavljati dve podobni skupini blokov, pri čemer vsaka skupina vsebuje zadostno število blokov.

   $\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N \log(N))$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(\log(N))$.

9. S seznama odstrani vse koordinatne pare, čigar absolutna vrednost dvorazsežnega medsebojnega odmika ne dosega ali presega pragovne vrednosti H_d.

   Ta korak zagotavlja, da pri izrisu slike podvojitev ne upoštevamo nobenega takega bločnega para, čigar pripadajoča slikovna izseka se na vhodni sliki prekrivata.

   Absolutno vrednost dvorazsežnega mesebojnega odmika računamo po naslednji ustaljeni formuli:

   $$o_a = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.9}$$

   V enačbi 2.9 predstavljata (x_i, y_i) in (x_j, y_j) dvodimenzionalna odmika dveh slikovnih izsekov od levega zgornjega kota slike.

   $\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(1)$.

10. Izriši sliko podvojitev. Za izris uporabi informacije o legah pripadajočih slikovnih izsekov tistih koordinatnih parov, ki so ostali na seznamu. En koordinatni par ustreza enemu podvojenemu slikovnemu izseku in njegovi podvojitvi.

    Slika podvojitev je v naši izvedbi predstavljena kot črna podlaga, na kateri so z različnima odtenkoma sive barve označena območja kopiranih slikovnih elementov (podvojene regije) in območja kopij teh slikovnih elementov (podvojitve regij).

    Jedri (*kernels*), ki se uporabita v operacijah erozije in dilatacije, nista odvisni od velikosti preiskovane slike, zato ne povečata računske zahtevnosti tega koraka.

    $\mathrm{T}(N) = \mathrm{T_W}(N) = \mathrm{O}(N)$, $\mathrm{S}(N) = \mathrm{S_W}(N) = \mathrm{O}(N)$.

S formalnim opisom korakov algoritma DRD zaključujemo drugo poglavje. V naslednjem poglavju si bomo pogledali, kako uspešen in učinkovit je algoritem v praksi.

Poglavje 3

Uspešnost in učinkovitost algoritma DRD

Delovanje algoritma DRD bomo ovrednostili z upoštevanjem dveh meril: uspešnosti in učinkovitosti. S pomočjo izsledkov opravljenih preizkusov bomo podali ustrezne ocene zanju.

V okviru ugotavljanja *uspešnosti* bomo prikazali, kako natančen je algoritem pri ločevanju pristnih slik od ponaredkov. Prav tako bomo navedli nekaj operacij digitalne obdelave slik, ki algoritem onesposobijo in nekaj operacij, na nekatere je algoritem odporen.

V okviru ugotavljanja *učinkovitosti* bomo v povezavi s formalno že določeno računsko zahtevnostjo predstavili tipično porabo pomnilniškega prostora in časa. Obenem bomo tudi ocenili, kako vhodni podatki, torej vsebina preiskovane slike, vplivajo na obe lastnosti.

Lastnosti preizkusnih slik

Algoritem DRD smo preizkusili na množici sivinskih slik 8-bitne barvne globine. Velikost večine slik je bila 512×512 elementov, nekaj slik je bilo tudi manjših. Množica je vsebovala pozitvne in negativne primere. V kontekstu preizkusnih slik izraz *pozitivni primer* pomeni sliko, v kateri obstajajo (umetno) podvojene regije, torej ponaredek. Izraz *negativni primer* pomeni pristno sliko. Razmerje med številoma pozitivnih in negativnih primerov ni bilo vnaprej načrtovano. Pozitivni primeri so vključevali v medijih objavljene ponarejene fotografije (dobljene na svetovnem spletu), za namene diplomskega dela narejene slike s podvojenimi regijami in doma ustvarjene "umetne" slike, ki niso vsebovale fotografij. Negativne primere so predstavljale digitalne fotografije, dobljene na spletu, doma narejene digitalne fotografije in nekaj optično prečitanih slik.

Medtem ko smo za ugotavljanje učinkovitosti dejansko uporabili dve zaključeni preizkusni množici (podmnožici množice vseh slik, ki so bile na razpolago), smo pri ugotavljanju uspešnosti uporabili posamezne primere, ki so po pričakovanjih dali oprijemljive rezultate.

3.1 USPEŠNOST

Uspešnost algoritma DRD lahko enačimo z deležem tistih primerov v celotni množici preizkusnih primerov, ki jih algoritem pravilno obravnava: če gre za pozitivni primer, ga obravnava kot ponaredek, če pa gre za negativni primer, ga obravnava kot pristno sliko. *Prag odločanja* je število ugotovljenih podvojenih blokov, ki loči ponaredke od pristnih slik. V naši izvedbi algoritma DRD ima prag odločanja vrednost 1. Če torej po izločitvi nepravih kandidatov v množici kandidatov ostane vsaj en bločni par, potem je preiskana slika obravnavana kot ponaredek.

Kot bomo videli v nadaljevanju poglavja, v množici digitalnih slik obstajajo tako pristne slike z velikim številom navidezno podvojenih blokov, kot tudi ponaredki, pri katerih je nemogoče odkriti že en sam podvojen blok. To pomeni, da ne moremo izbrati takšnega praga odločanja, ki bi nam vsaj teoretično zagotavljal stoodstotno natančnost. Tej natančnosti se zato lahko le približamo. Pri tem nam pomaga pravilna izbira vrednosti preostalih parametrov odkrivanja.

3.1.1 UPORABLJENE VREDNOSTI PARAMETROV IN NJIHOV VPLIV NA OBRAVNAVO SLIK

Med preizkušanjem algoritma DRD smo za vse parametre z izjemo H_n uporabili vrednosti, predlagane v [20]. Te so se po opravljenih preizkusih izkazale za povsem primerne. Zbrane so v naslednji tabeli.

Tabela 3.1: Vrednosti parametrov, ki smo jih uporabili za preizkušanje uspešnosti algoritma DRD.

Parameter (oznaka)	**Uporabljene vrednosti**
Število slikovnih elementov v bloku (b)	64
Delež zanemarjene variance vzdolž glavnih osi prostora (ε)	0,01
Število kvantizacijskih stopenj (Q)	256
Natančna zgornja meja medindeksne razlike vektorskih parov, ki predstavljajo kandidate (H_n)	2, 3, 5, 7, 9, 10
Prag pogostnosti enakih medbločnih odmikov (H_f)	128

Prag velikosti medbločnih odmikov (H_d)	16

Vrednost 100, ki je za parameter H_n predlagana v [20], v naših preizkusih ni dala pričakovanih rezultatov. Tako pri pozitivnih kot negativnih primerih na slikah podvojitev ni bilo mogoče razbrati podvojenih regij, saj je bilo označenih bistveno preveč slikovnih izsekov. Poizkusi so pokazali, da je v veliki večini primerov vrednost 2 za H_n povsem primerna in algoritmu DRD zagotavlja pravilno obravnavo, torej uvrstitev preiskane slike med pozitivne ali negativne primere. Pri nekaterih od teh primerov, ki so tudi prikazani na slikah v tem poglavju, smo večje vrednosti za ta parameter uporabili zgolj za bolj natančo označitev podvojenih območij. Pri samo nekaj izjemah pa je bilo potrebno večjo vrednost za H_n dejansko uporabiti za pravilno obravnavo primerov.

Vrednosti parametrov imajo naslednji vpliv na potek, učinkovitost in uspešnost odkrivanja podvojenih regij z algoritmom DRD:

- b – Vpliva na učinkovitost in uspešnost.

  Majhne vrednosti pohitrijo delovanje in omogočajo odkrivanje manjših podvojenih regij. Premajhne vrednosti imajo za posledico premajhne in s tem medsebojno preveč podobne bloke, kar zmanjša uspešnost (napačana obravnava negativnih primerov). Velike vrednosti izrazito povečajo čas preiskovanja in obenem tudi zmanjšajo uspešnost (napačna obravnava pozitivnih primerov). Pri velikih vrednostih algoritem odkrije le zelo velike podvojene regije, ki pa so ponavadi brez večjih težav ugotovljive že s prostim očesom.

- ε – Vpliva na učinkovitost in uspešnost.

  Premajhne vrednosti imajo za posledico nezadostno (prevelik vektor značilnic), prevelike pa pretirano (premajhen vektor značilnic) poenostavitev blokov. V prvem primeru se podaljša čas preiskovanja, olajšana pa je tudi preslepitev algoritma z naknadno obdelavo slike po podvajanju regij (dodajanje šuma, izgubno stiskanje). V drugem primeru se skrajša čas preiskovanja in zmanjša uspešnost.

- Q – Vpliva na uspešnost.

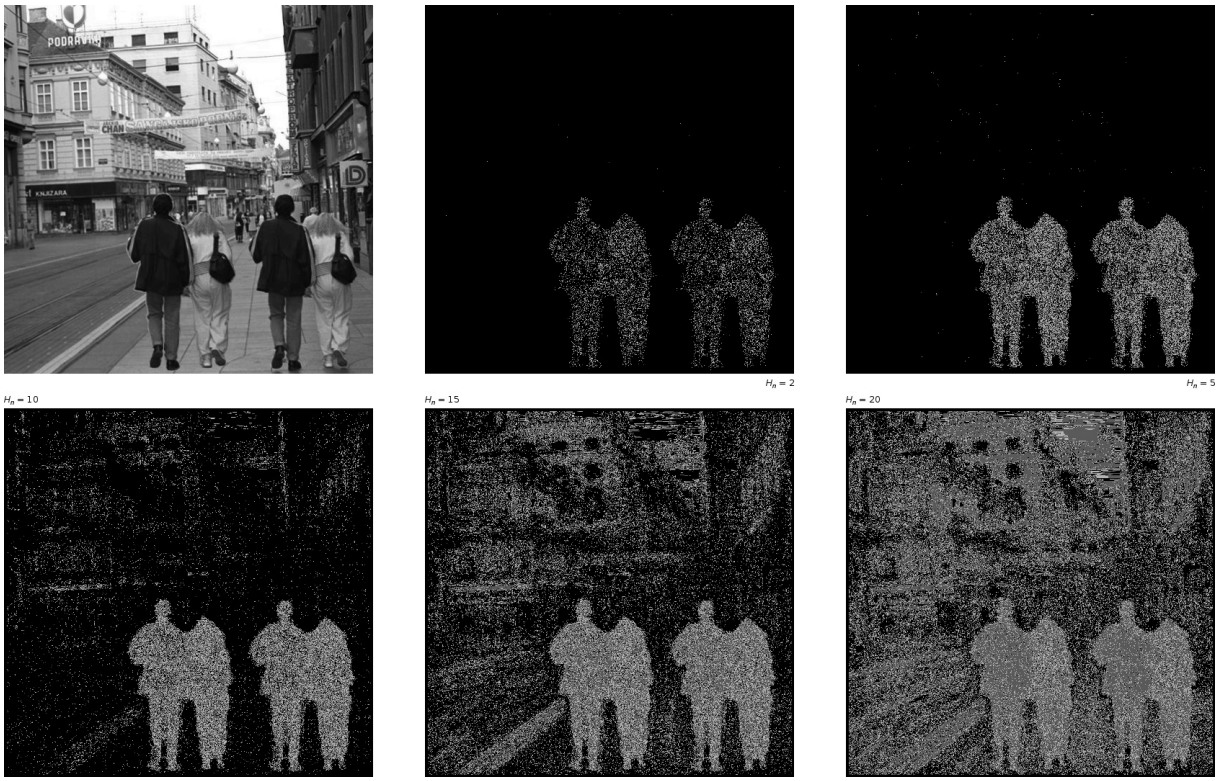
  Premajhne vrednosti vodijo do prevelike medsebojne podobnosti blokov in s tem manjše uspešnosti. Algoritem DRD preiskuje sivinske slike 8-bitne globine, zato je za ta parameter najbolj "naravna" izbira vrednost 256.

- H_n – Vpliva na učinkovitost in uspešnost.

  Vrednosti malo nad 2 v nekaterih primerih omogočijo pravilno obravnavo preiskovanih slik. V večini primerov z razmeroma velikimi podvojenimi regijami takšne vrednosti povečajo natančnost označitve podvojenih regij, pri primerih z manjšimi podvojenimi regijami pa to natančnost zmanjšajo. Še večje vrednosti parametra ne povečajo uspešnosti, pač pa le znatno podaljšajo čas preiskovanja.

  Vpliv parametra na rezultat je prikazan na sliki 3.1, kjer smo za bolj nazoren prikaz slik podvojitev namenoma izpustili operaciji erozije in dilatacije v koraku

6 algoritma DRD.



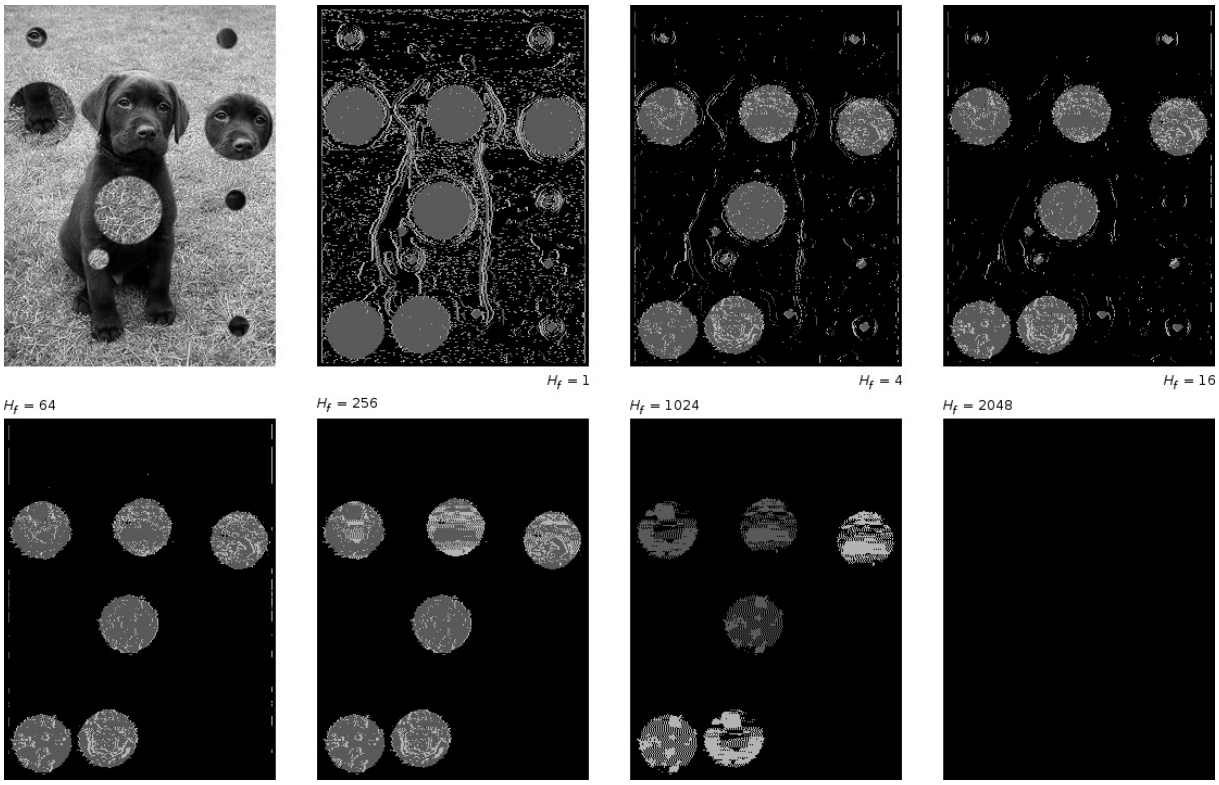
Slika 3.1: Vpliv parametra H_n na rezultat. Za prikazan primer dobimo po izvedbi erozije in dilatacije najbolj natančen rezultat z uporabo vrednosti 5.[6]

- H_f – Vpliva na uspešnost.

  Premajhne vrednosti zmanjšajo uspešnost zaradi napačne obravnave negativnih primerov (posamezni pari po naključju podobnih blokov so obravnavani kot podvojena regija). Prevelike vrednost zmanjšajo uspešnost zaradi napačne obravnave pozitivnih primerov (iz preiskave so izločene regije, ki so v resnici podvojene).

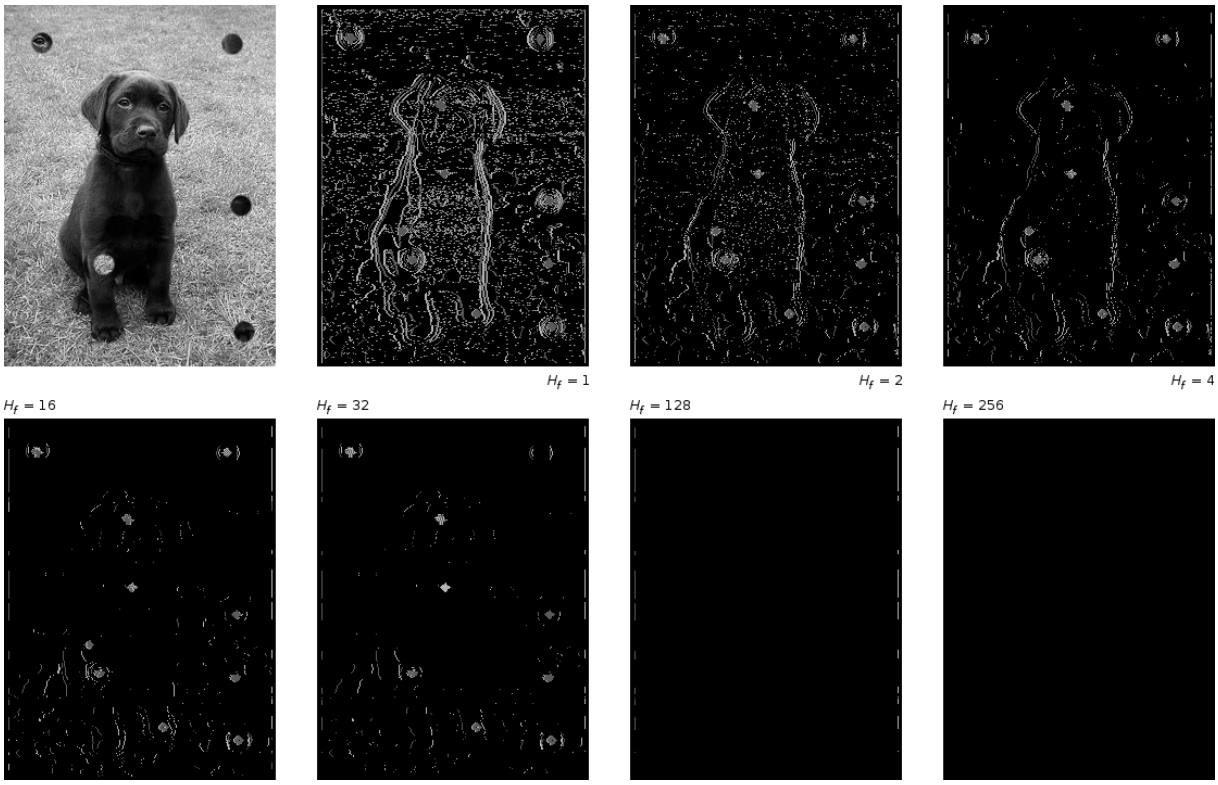
  Vpliv parametra H_f na rezultat je prikazan na slikah 3.2(a) in 3.2(b). Pri določanju slik podvojitev smo tudi tu namenoma izpustili operaciji erozije in dilatacije. Na sliki 3.2(a) so večje podvojene regije dovolj velike, da lahko s primerno vrednostjo parametra natačno določimo njihovo lego in obliko. Na sliki 3.2(b) premajhna velikost podvojenih regij glede na parameter b algoritmu DRD ne omogoča zadovoljive natančnosti.

6 Za bolj nazoren prikaz na slikah podvojitev nista bili izvedeni erozija in dilatacija. Vse slike podvojitev so bile dobljene z uporabo vrednosti 128 za parameter H_f.



Slika 3.2(a): Vpliv parametra H_f na rezultat.
Večje podvojene regije so glede na vrednost b dovolj velike.[7]

7 Za bolj nazoren prikaz na slikah podvojitev nista bili izvedeni erozija in dilatacija. Vse slike podvojitev so bile dobljene z uporabo vrednosti 2 za parameter H_n.



Slika 3.2(b): Vpliv parametra H_f na rezultat.
Podvojene regije so glede na vrednost b premajhne.[8]

- H_d – Vpliva na uspešnost.

  Premajhne vrednosti zmanjšajo uspešnost zaradi napačne obravnave negativnih primerov (dopuščamo, da se slikovna izseka posameznega kandidata na sliki prekrivata). Prevelike vrednosti prav tako zmanjšajo uspešnost, vendar na račun napačne obravnage pozitivnih primerov (podvojenih regij, ki ležijo blizu skupaj, algoritem ne odkrije).

3.1.2 PRIMERI PRAVILNE OBRAVNAVE PREISKOVANIH SLIK

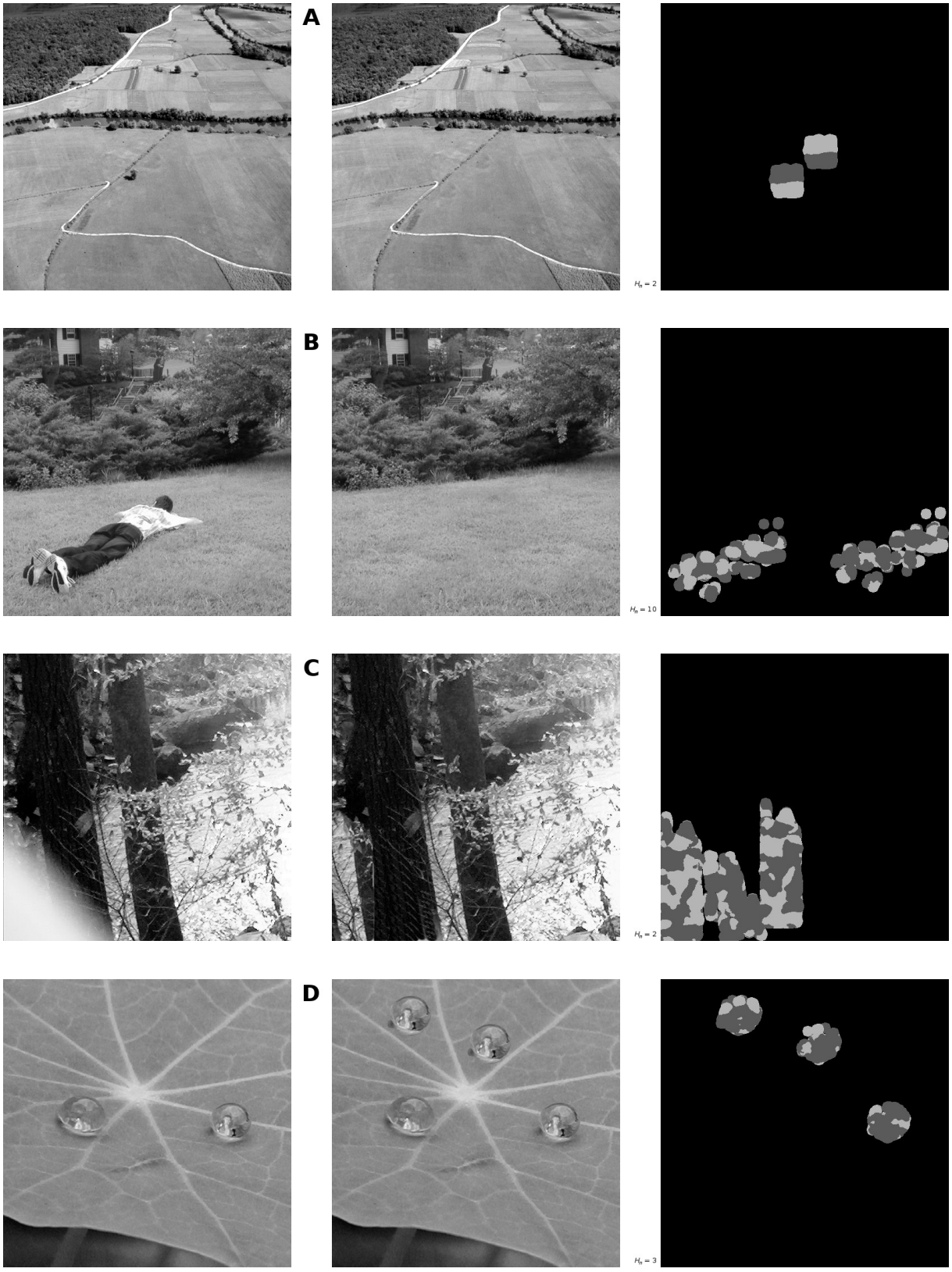
V tem razdelku bomo predstavili pozitivne in negativne primere, ki jih je algoritem DRD pravilno obravnaval, torej pravilno določil, ali je preiskovana slika ponarejena ali pristna.

Primeri uspešno odkritih podvojenih regij

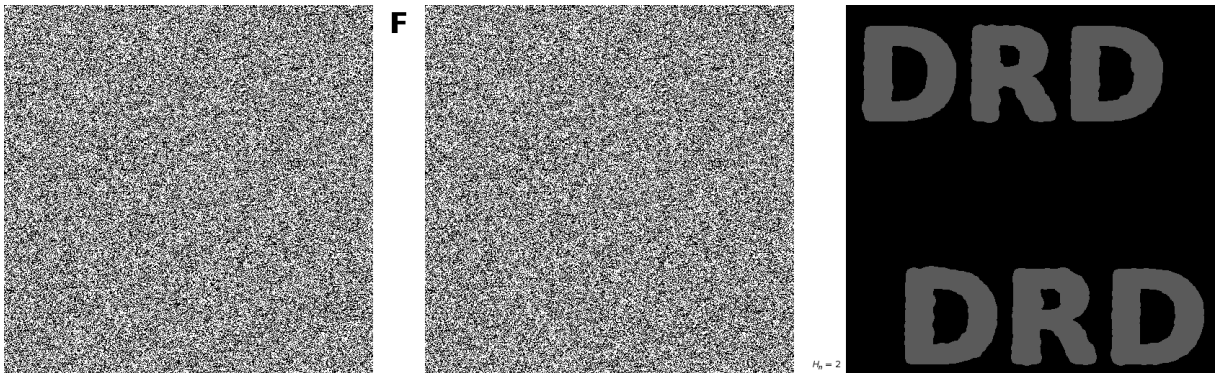
Slika 3.3 prikazuje pristno sliko (samo za nekatere primere), sliko s podvojenimi regijami in sliko podvojitev za nekaj primerov, ki jih je algoritem DRD pravilno uvrstil med pozitivne primere. Vrednost večja od 2 za parameter H_n je bila za primer

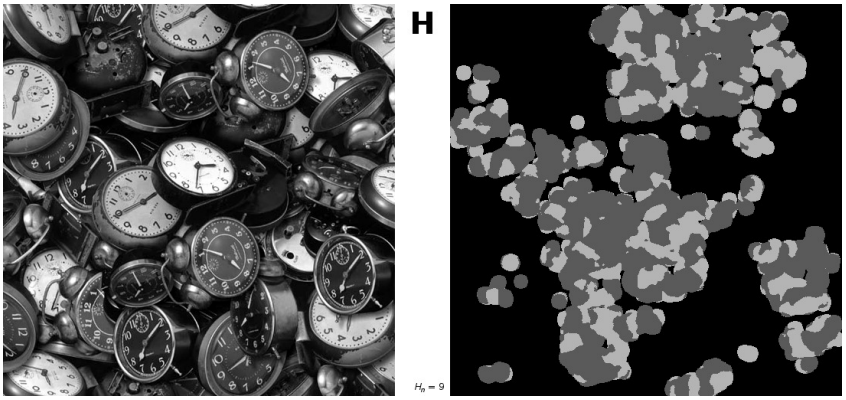
8 Za bolj nazoren prikaz na slikah podvojitev nista bili izvedeni erozija in dilatacija. Vse slike podvojitev so bile dobljene z uporabo vrednosti 2 za parameter H_n.

D uporabljena za zagotovitev pravilnosti obravnave, za nekatere od preostalih primerov pa le za bolj natančno določitev podvojenih območij.

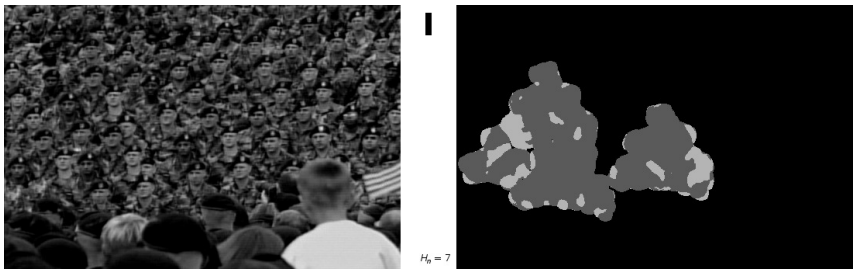


(nadaljevanje slike na naslednji strani)





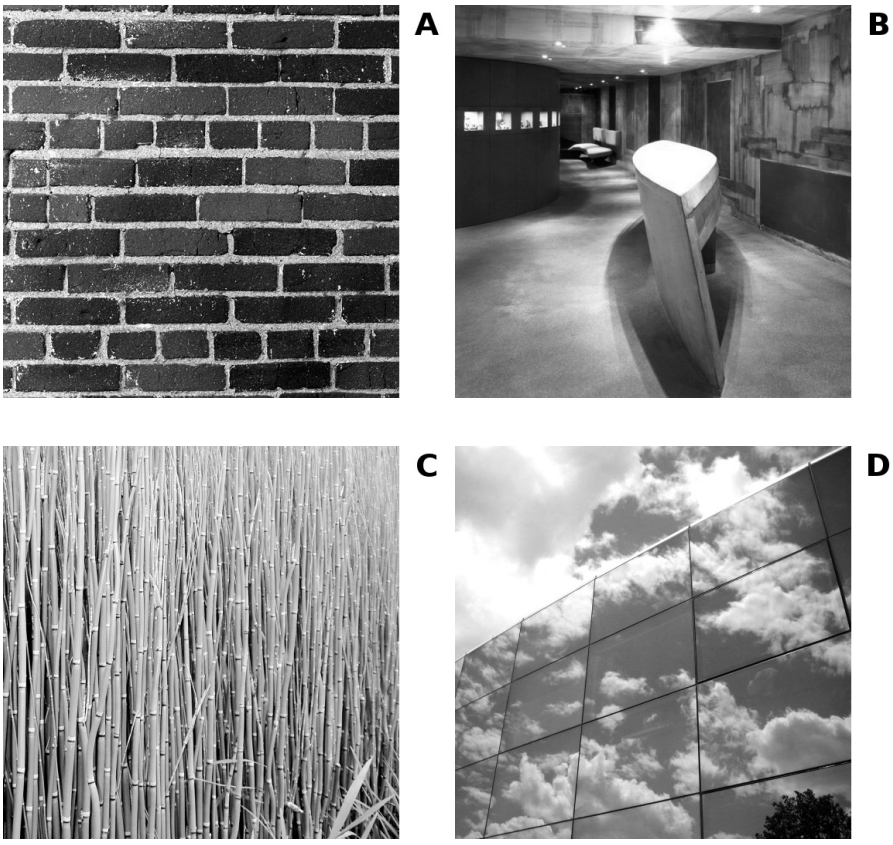
(nadaljevanje slike na naslednji strani)



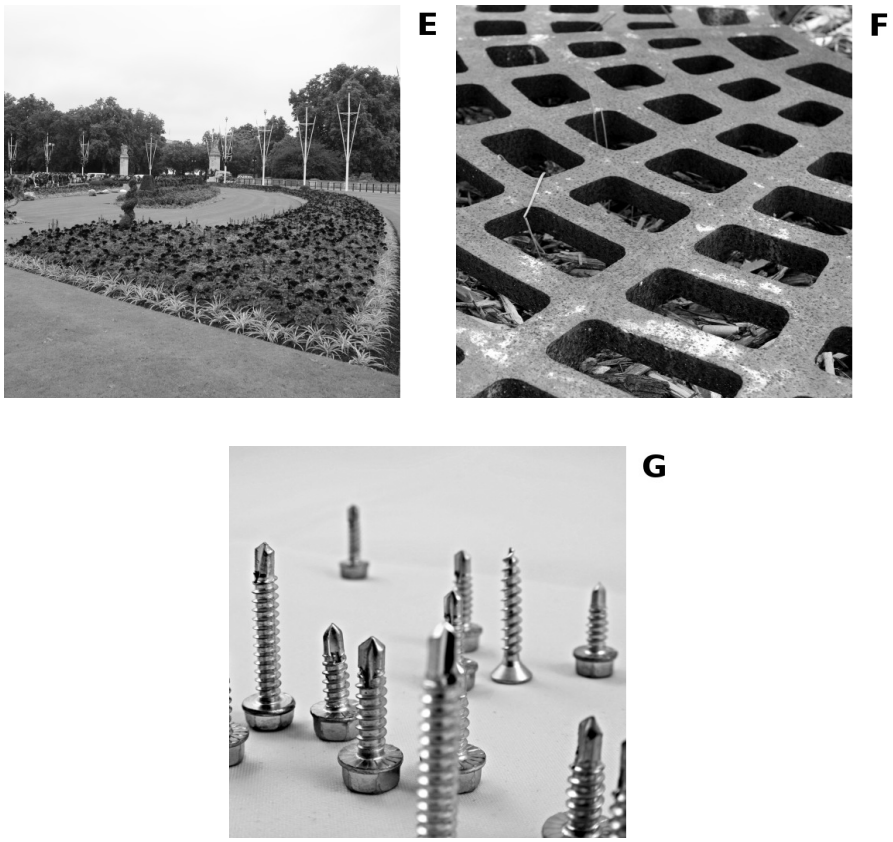
Slika 3.3: Primeri pravilno obravnavanih pozitivnih primerov.

Primeri pravilne obravnave pristnih slik

Slika 3.4 prikazuje nekaj primerov, ki jih je algoritem DRD pravilno uvrstil med negativne primere; slike podvojitev so zato ostale prazne. Parameter H_n za preiskavo vhodnih slik je bil nastavljen na vrednost 2.



(nadaljevanje slike na naslednji strani)



Slika 3.4: Primeri pravilno obravnavanih negativnih primerov.

3.1.3 PRIMERI NAPAČNE OBRAVNAVE PREISKOVANIH SLIK

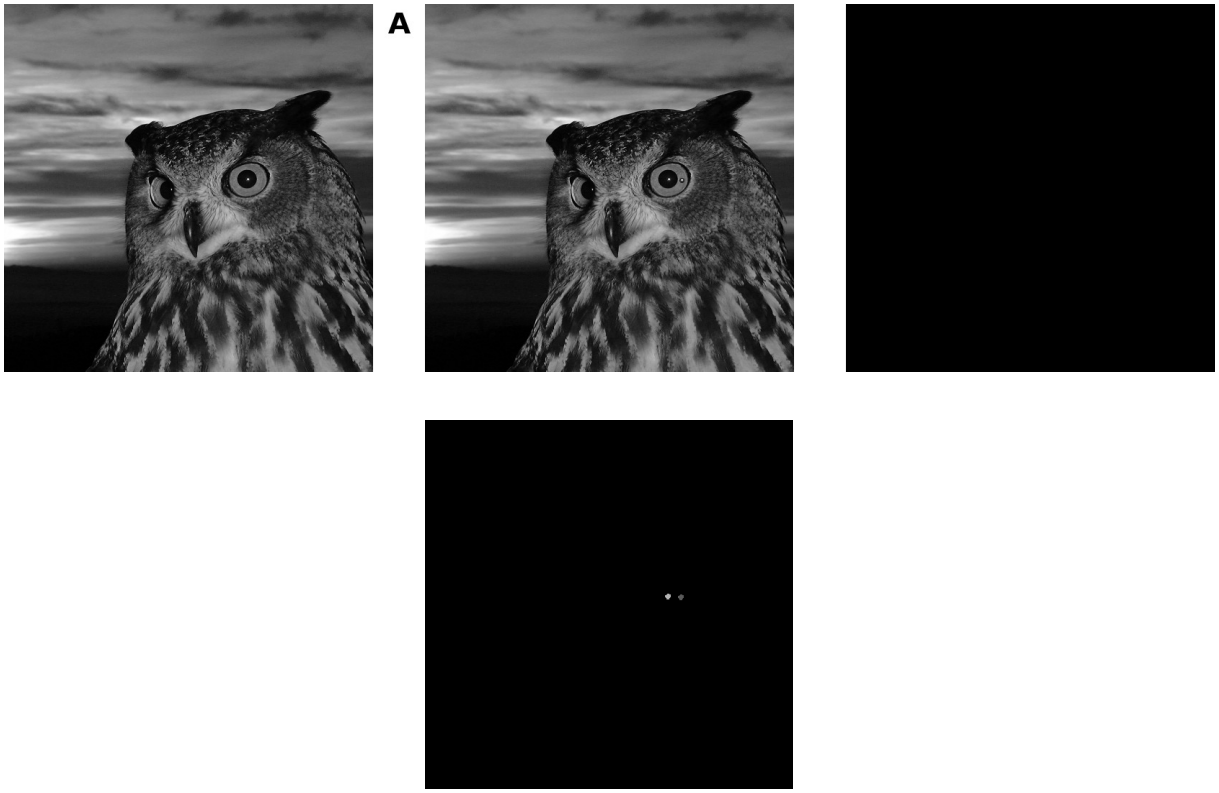
V tem razdelku bomo predstavili pozitivne in negativne primere, ki jih je algoritem DRD napačno obravnaval. To pomeni, da je prve obravnaval kot pristne slike, druge pa kot ponaredke.

Napačno ali neprepričljivo obravnavani pozitivni primeri

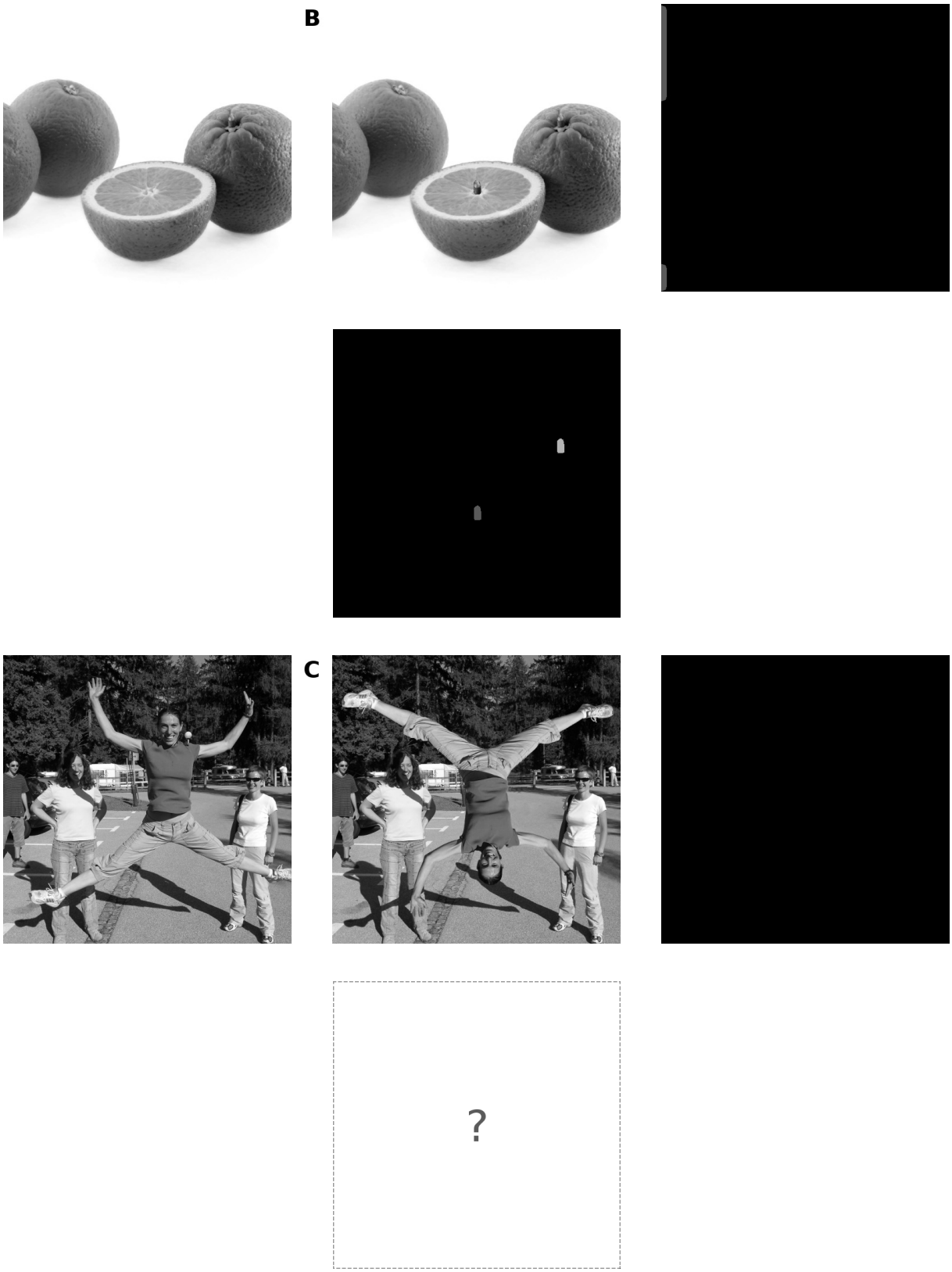
Slika 3.5 prikazuje pristno sliko, sliko s podvojenimi regijami, dobljeno sliko podvojitev in idealno sliko podvojitev za nekaj primerov, pri katerih algoritem DRD bodisi ni bil uspešen bodisi bi lahko bil bolj prepričljiv. Vse slike podvojitev so bile dobljene z uporabo vrednosti 2 za parameter H_n. V primeru C je šlo za retuširanje pristne slike, zato idealne slike podvojitev nismo mogli določiti.

Napačno obravnavo v primerih A in C je povzročila izbira vrednosti za parameter b, ki je bila za konkretne podvojene regije prevelika. Zmanjšanje vrednosti parametra H_f prav tako ni omogočilo odkritja podvojenih regij na levem sovinem očesu in v ozadju akrobatke. Podobne okoliščine smo imeli v primeru B, le da je tu algoritem DRD (pomotoma) uvrstil sliko med pozitivne primere ne da bi odkril dejansko podvojeno regijo – pomarančni pecelj.

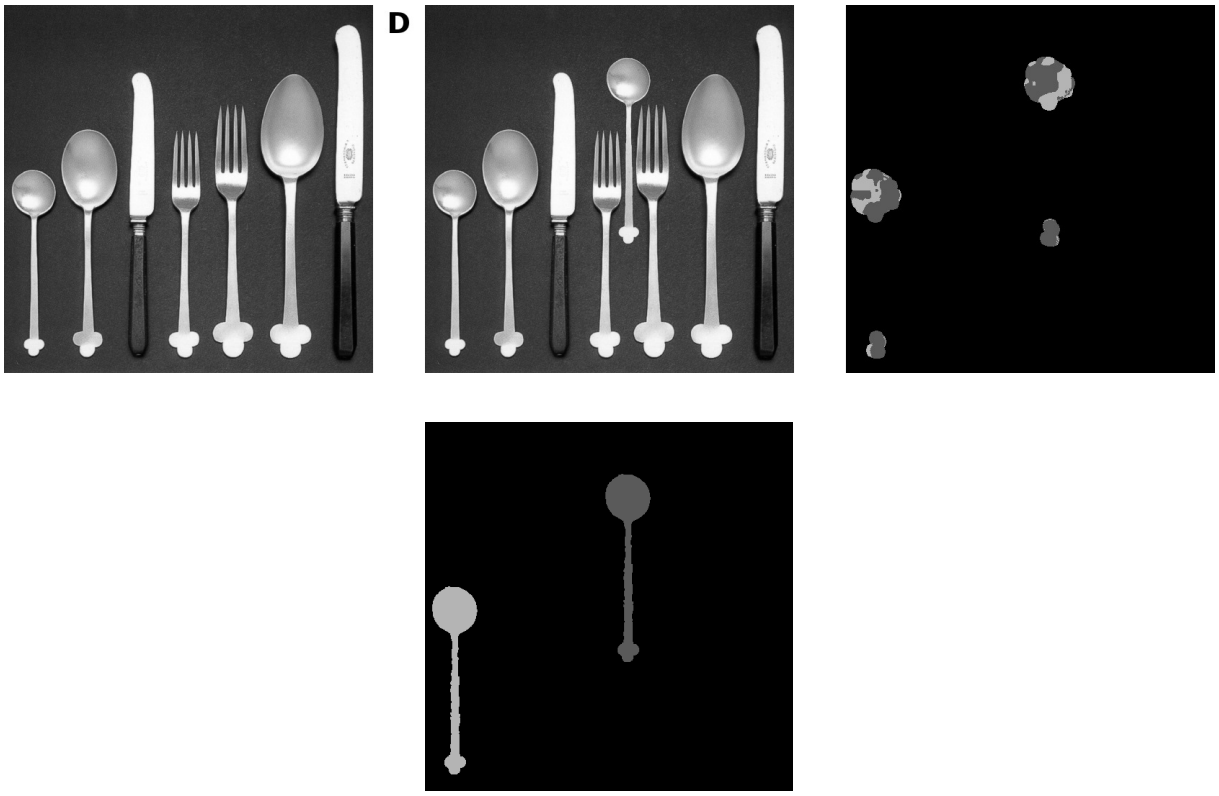
Primer D lepo ponazarja, kako lahko algoritem DRD "prezre" premajhne podvojene slikovne izseke. Tu smo s prilagoditvjio vrednosti parametra ε (iz 0,01 na 0,001) vseeno dosegli malo bolj natančno označitev podvojene regije, medtem ko je bil podoben ukrep pri primerih A, B in C neuspešen.



(nadaljevanje slike na naslednji strani)



(nadaljevanje slike na naslednji strani)

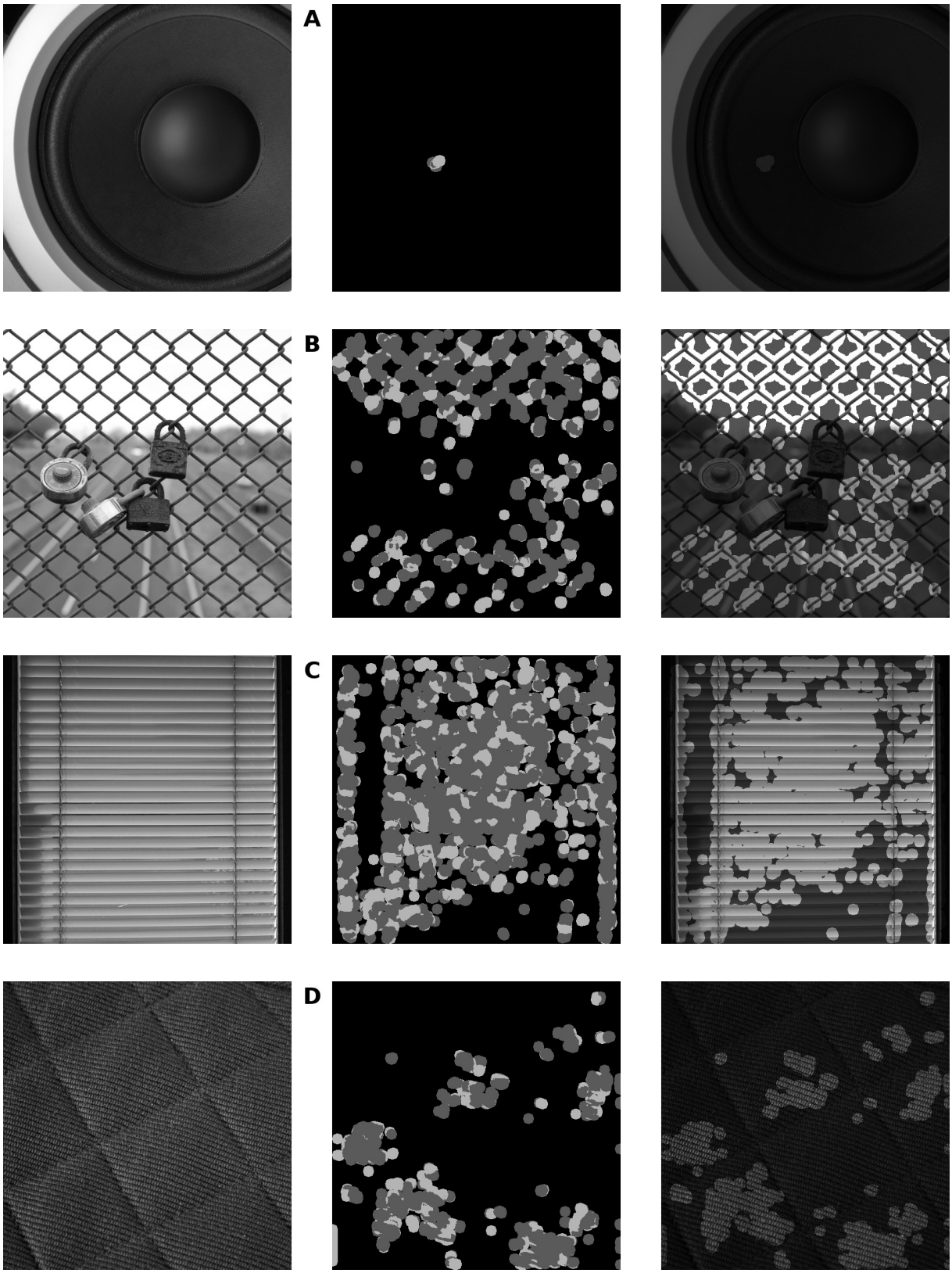


Slika 3.5: Primeri napačno obravnavanih pozitivnih primerov.

Napačno obravnavani negativni primeri

Slika 3.6 prikazuje preiskovano sliko, sliko podvojitev in označeno sliko za nekaj primerov, ki jih je algoritem DRD napačno uvrstil med pozitivne primere. Vse slike podvojitev so bile dobljene z uporabo vrednosti 2 za parameter H_n in vrednosti 128 za parameter H_f.

Razlog za neuspeh je bil v premajhni velikosti vektorjev značilnic, kar je privedlo do pretirane poenostavitve blokov. Takšne vektorje značilnic dobimo ponavadi iz slik z enovitimi območji ali ponavljajočimi se vzorci. Zmanjšanje vrednosti parametra ε na desetino prvotno uporabljene vrednosti (iz 0,01 na 0,001) je odpravilo težavo in omogočilo pravilno obravnavo prikazanih primerov, vendar hkrati tudi zmanjšalo robustnost, ki je ena najpomembnejših lastnosti algoritma DRD.



Slika 3.6: Primeri napačno obravnavanih negativnih primerov.

3.1.3 POSTOPKI NAKNADNE OBDELAVE, NA KATERE JE ALGORITEM ODPOREN

O robustnosti nekega pristopa lahko govorimo takrat, ko predhodno izvajanje omejenega nabora postopkov digitalne obdelave na preiskovani sliki bistveno ne zmanjša njegove uspešnosti oziroma natančnosti dobljenih rezultatov. Za algoritem DRD takšen nabor tvorijo shranjevanje v izgubni zapis JPEG, dodajanje umetnega šuma in filtriranje s filtrom mediana.

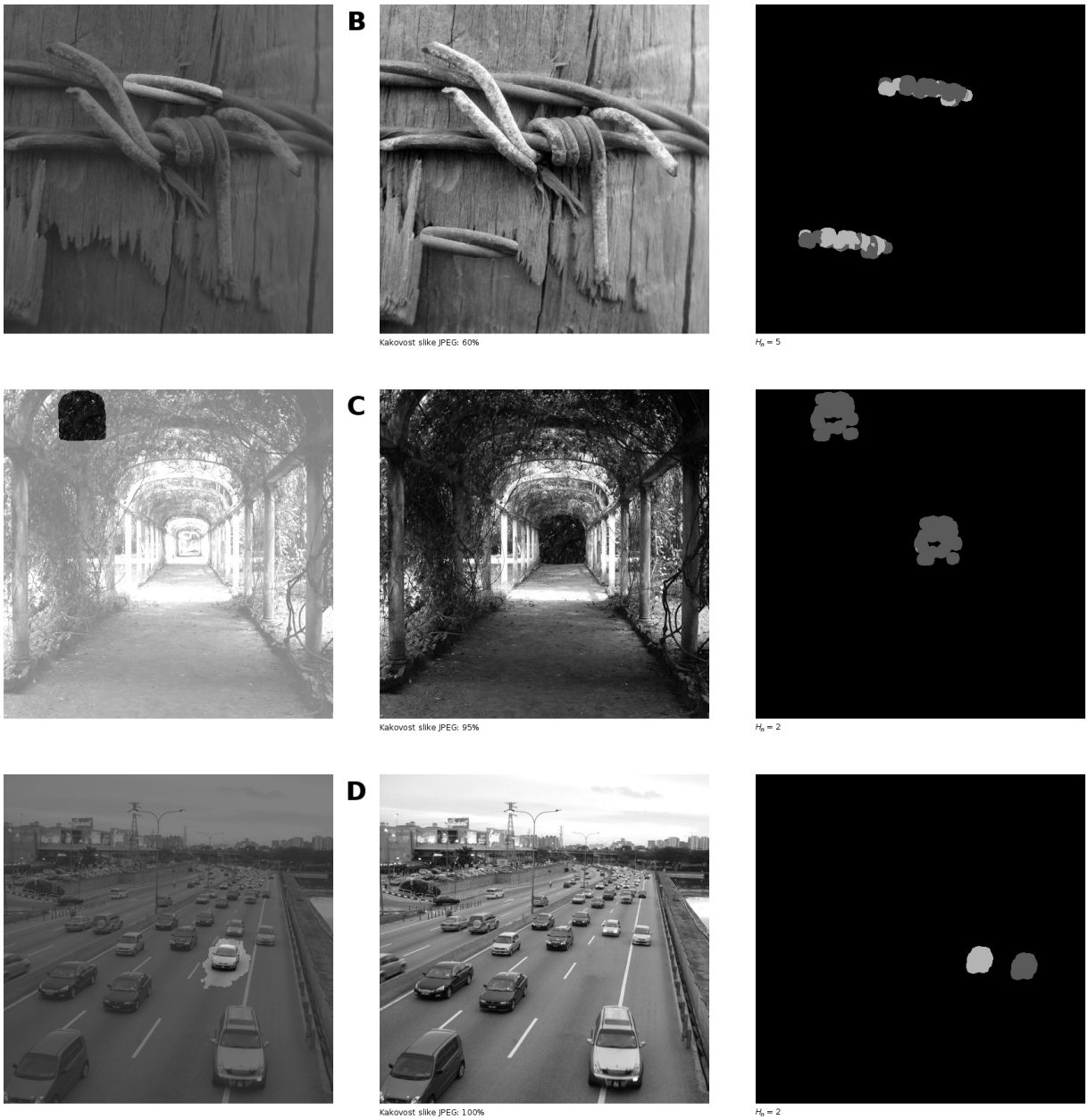
Pri preizkušanju odpornosti algoritma DRD na omenjene postopke smo se omejili na pravilnost obravnave pozitivnih in negativnih primerov, medtem ko nas natančnost označitve podvojenih regij ni toliko zanimala. Priimeri so namenoma vsebovali različne stopnje podrobnosti in podvojene regije različnih velikosti. Za vsako vhodno sliko smo poizkus opravili trikrat in sicer z nastavitvijo parametra H_n na vrednosti 2, 3 in 5.

Shranjevanje v izgubni zapis JPEG

Pri shranjevanju v izgubni zapis JPEG gre za stiskanje slike z izgubnim algoritmom za stiskanje. Manjša kakovost zapisa JPEG pomeni večjo izgubo kakovosti prikaza. Odpornost algoritma DRD proti temu postopku smo preizkušali tako, da smo vsako preiskovano sliko shranili v zapis JPEG kakovosti med 100 in 60 (v korakih po 5) in nato za vsako tako pridobljeno sliko preverili pravilnost obravnave. Preizkusi so pokazali, da je algoritem DRD pri nižjih kakovostih zapisa JPEG bistveno bolj uspešen pri večjih podvojenih regijah in obratno, medtem ko velikost vektorja značilnic na uspešnost praktično ne vpliva.

Na sliki 3.7 sta za nekaj preizkusnih primerov poleg slike podvojitev prikazani pristna slika s poudarjenimi podvojenimi regijami in preiskovana slika pri tisti najmanjši kakovosti zapisa JPEG, pri kateri smo z ustrezno izbiro vrednosti za H_n uspeli zagotoviti pravilno obravnavo. Za primer B smo tako s povečevanjem vrednosti H_n od 2 do 5 prag kakovosti zapisa JPEG, pri kateri je bila obravnava še pravilna, uspeli zmanjšati s 75 na 60. V primeru D ni šlo za enkratno podvojitev celotne regije, pač pa za postopno retušo z uporabo čopiča precej manjše velikosti.

(nadaljevanje slike na naslednji strani)



Slika 3.7: Primeri pravilno obravnavanih pozitivnih primerov, ki so bili shranjeni v izgubnem zapisu JPEG.

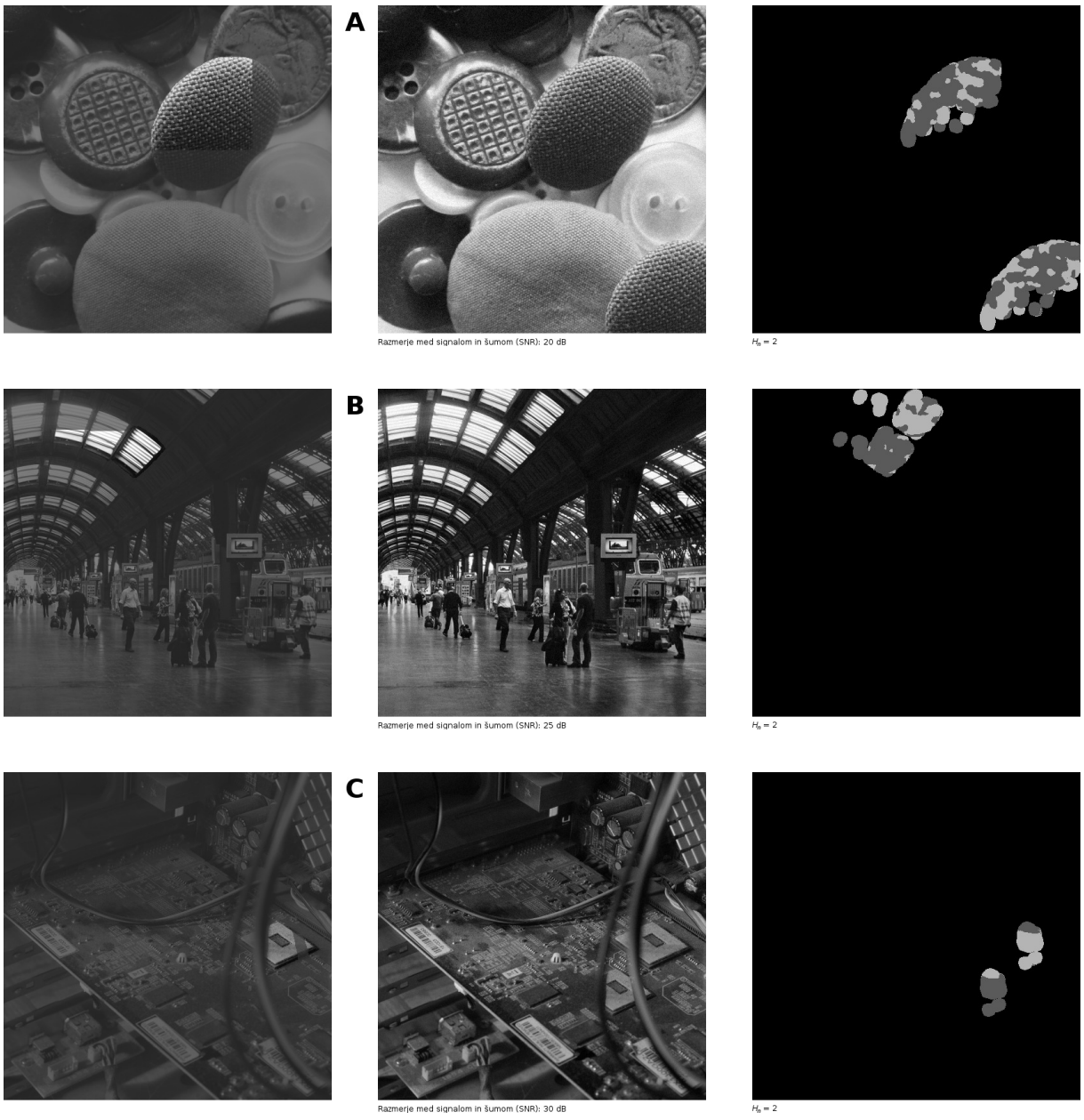
Pri shranjevanju v izgubni zapis JPEG bi v določenih robnih primerih ugotovili vpliv razmika med podvojeno regijo in njeno podvojitvijo na pravilnost obravnave. Ta vpliv je posledica delovanja algoritma za shranjevanje v zapis JPEG, ki pred izvedbo diskretne kosinusne transformacije vedno razdeli sliko na bloke velikosti 8×8.

Dodajanje umetnega šuma

Preizkusne primere za preverjanje algoritma DRD za odpornost proti temu postopku smo dobili tako, da smo v vsako preiskovano sliko dodali aditivni beli šum v razmerju

SNR med 35 in 20 (v korakih po 5). Pokazalo se je, da je algoritem DRD pri manjših razmerjih SNR uspešnejši pri večjih podvojenih regijah in obratno. Zaradi narave postopka dodajanja šuma konkreten razmik med podvojeno regijo in njeno podvojitvijo ne igra vloge pri pravilnosti obravnave.

Na sliki 3.8 sta za nekaj preizkusnih primerov poleg slike podvojitev prikazani pristna slika s poudarjenimi podvojenimi regijami in preiskovana slika s tistim najmanjšim razmerjem SNR, pri katerem je bila obravnava še pravilna. Povečevanje vrednosti H_n ni vplivalo na uspešnost. Razlog za napačno obravnavo primera D je enak tistemu za primer D na sliki 3.7.



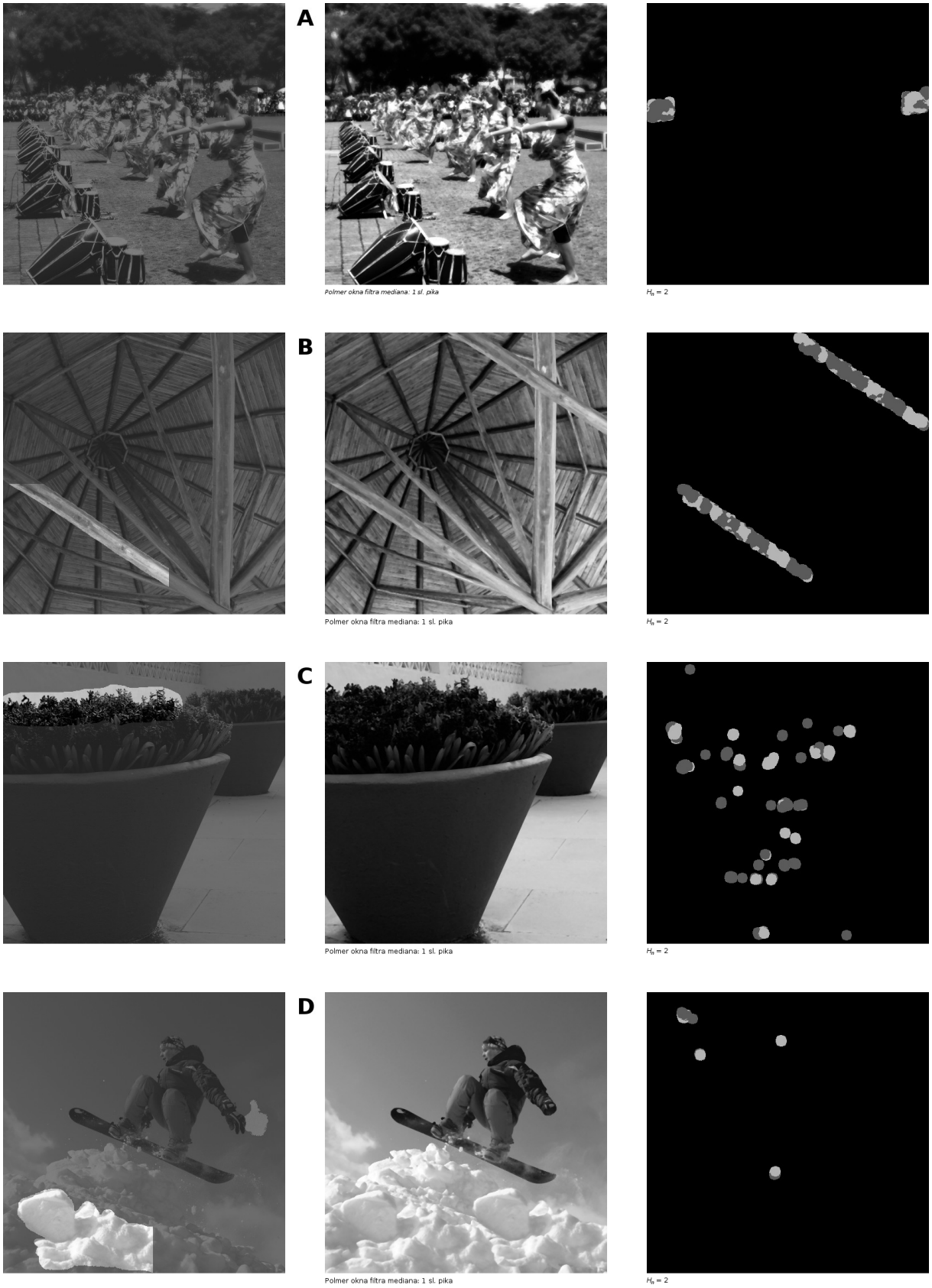
(nadaljevanje slike na naslednji strani)

Slika 3.8: Primeri pravilno in napačno obravnavanih pozitivnih primerov, ki jim je bil dodan aditivni beli šum.

Filtriranje s filtrom mediana

Na področju digitalne obdelave slik je ta postopek namenjen odstranjevanju šuma, pri čemer je boljši od običajnega **zabrisanja** (*blurring*), saj v sliki ohrani več podrobnosti. Preizkusne primere za preverjanje odpornosti proti temu postopku smo dobili tako, da smo vsako preiskovano sliko prefiltrirali s filtrom mediana z oknom polmera enega, dveh ali treh slikovnih elementov. Pri vrednosti 2 parametra H_n in velikosti polmera filtrirnega okna (r) 1 je algoritem DRD pravilno obravnaval dve tretjini od devetih preizkusnih primerov. Pri eni tretjini primerov je označil preveč slikovnih izsekov, torej tudi nepodvojene regije, kar je z vidika pravilnosti obravnave nedopustno. Pri velikosti r 2 je pravilno obravnaval en sam primer, pri vrednosti r 3 pa ni bil uspešen. Za ta postopek večje vrednosti parametra H_n niso povečale uspešnosti algoritma DRD na preizkusnih primerih.

Slika 3.9 za nekaj preizkusnih primerov prikazuje pristno sliko s poudarjenimi podvojenimi regijami, preiskovano sliko in sliko podvojitev. Primera A (velikost vektorja značilnic (f=16) in B (f=13) je algoritem pravilno obravnaval. Primera C (f=4) in D (f=6) je med pozitivne primere uvrstil po pomoti, saj je kot podvojene regije obravnaval področja, ki niso bila podvojena, dejansko podvojenih območij pa ni odkril.



Slika 3.9: Primeri pravilno in pomotoma pravilno obravnavanih pozitivnih primerov, ki so bili prefiltrirani s filtrom mediana.

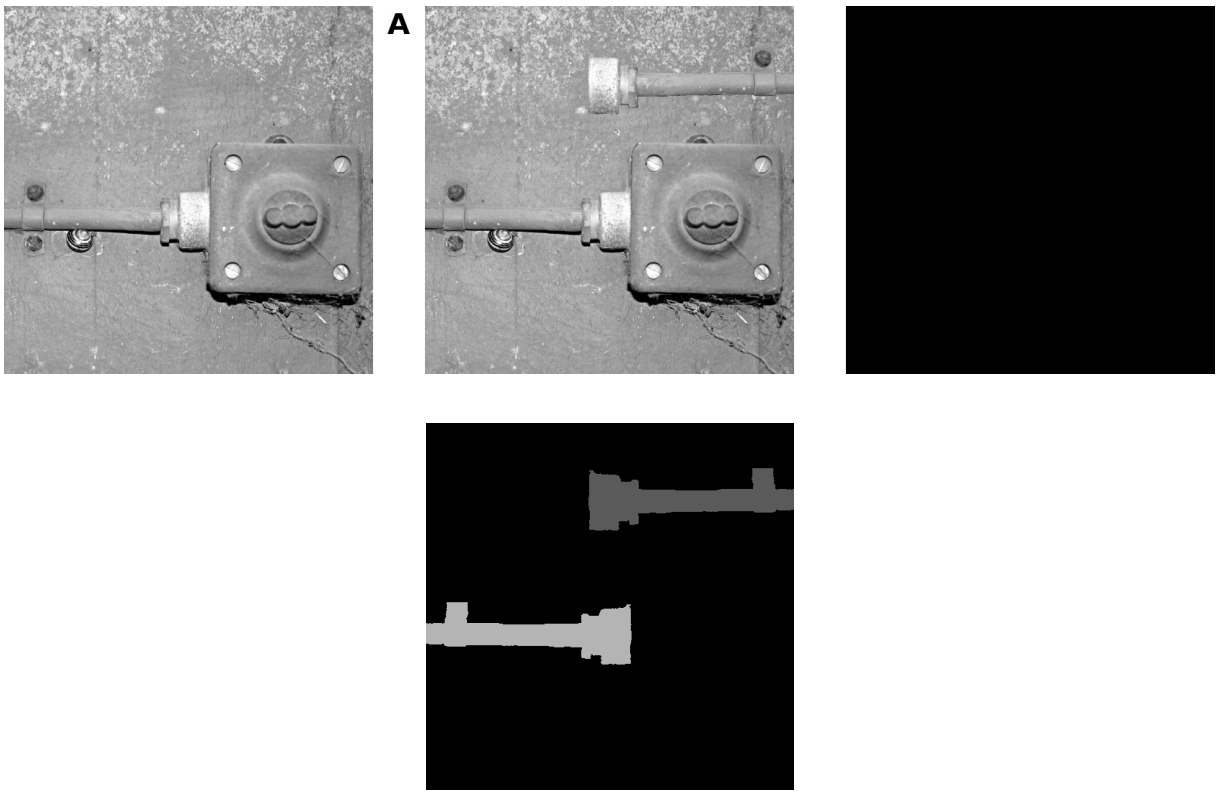
Zanimiva je ugotovitev, da je bil v nasprotju z izsledki preizkusov za shranjevanje v izgubni zapis JPEG in dodajanje umetnega šuma algoritem DRD pri preizkušanju odpornosti proti filtiranju s filtrom mediana na slikah z velikimi podvojenimi regijami le polovično uspešen.

3.1.4 POSTOPKI NAKNADNE OBDELAVE, KI ZMEDEJO ALGORITEM DRD

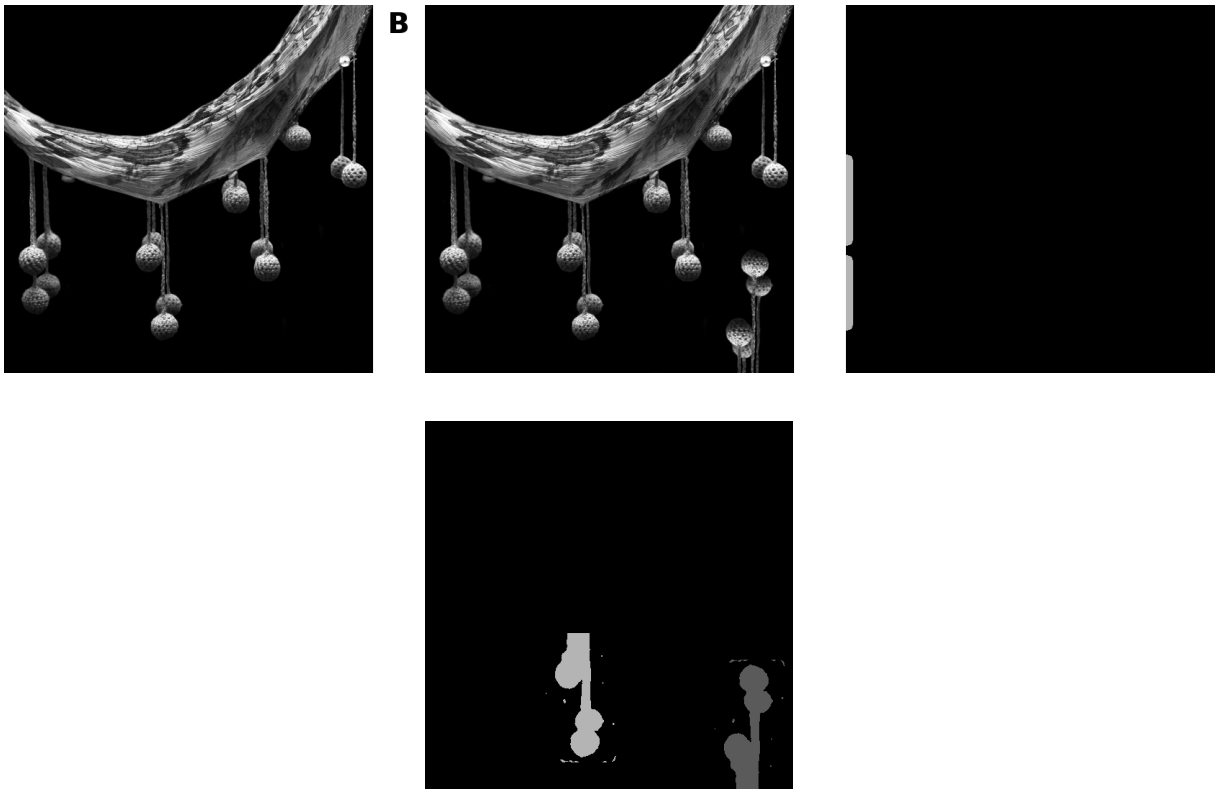
Poleg postopkov, ki zaradi robustnosti algoritma DRD ne zmanjšajo izrazito njegove uspešnosti, obstajajo tudi postopki, ki popolnoma zmedejo algoritem. Primeri takšnih postopkov so prezrcaljenje v katerikoli smeri, zasuk ter razteg oziroma skrčenje.

Prezrcaljenje

Urejenost slikovnih elementov znotraj blokov je pomembna pri ugotavljanju medsebojne podobnosti poenostavljenih predstavitev. Prezrcaljenje podvojenih regij, ki ima za posledico spremembo zaporedja elementov znotraj slikovnih izsekov podvojene regije, ugotavljanje podobnosti med bloki povsem onemogoči. Primera prezrcaljenja sta prikazana na sliki 3.10. Za vsak primer so prikazane pristna slika, slika s podvojenimi regijami, dobljena slika podvojitev in idealna slika podvojitev.



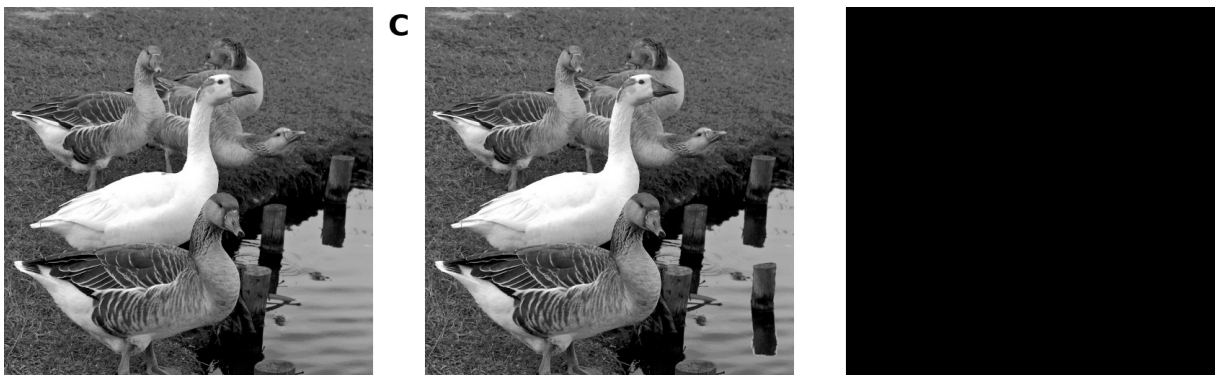
nadaljevanje slike na naslednji strani)



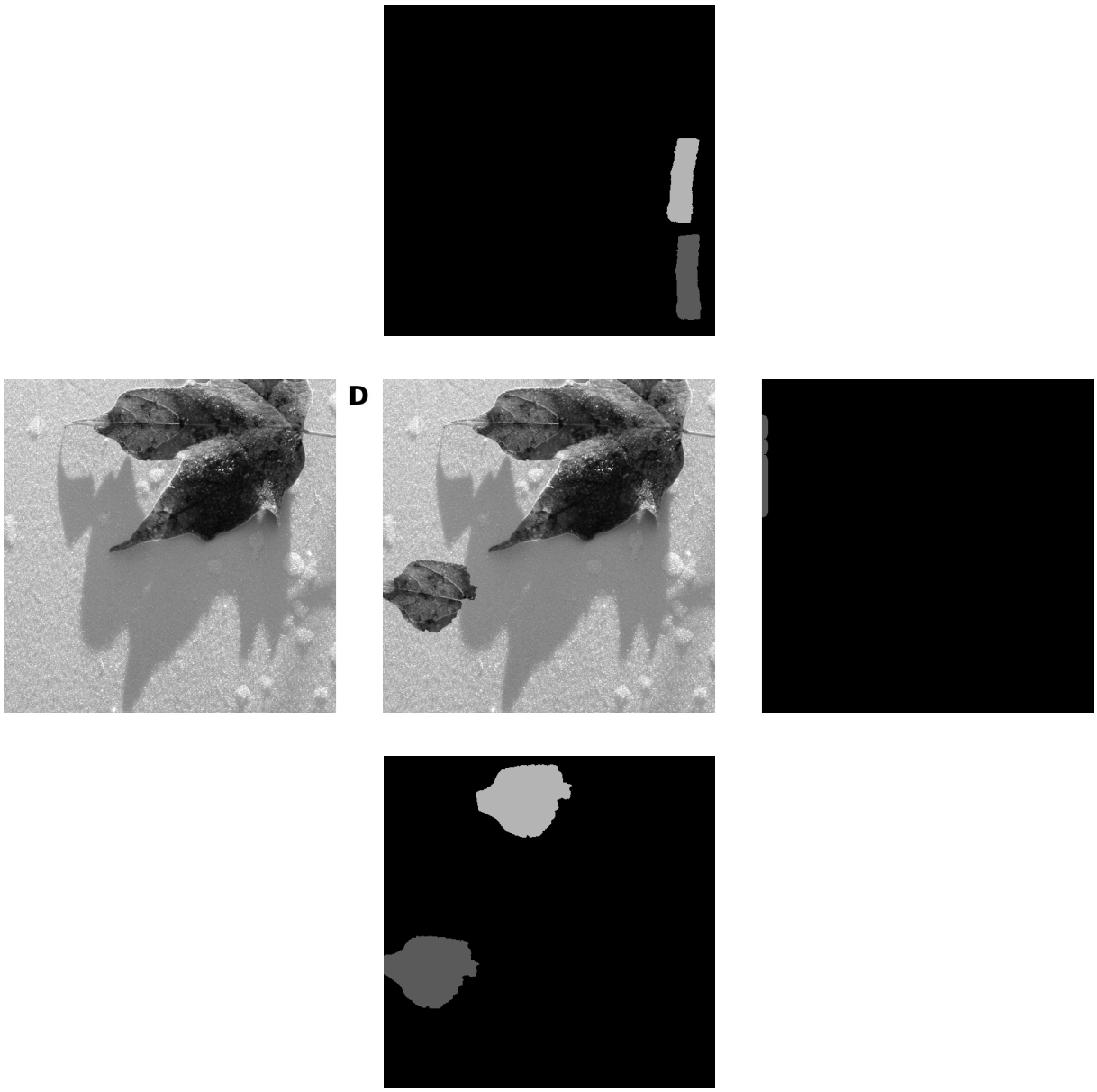
Slika 3.10: Primera slik s prezrcaljeno podvojeno regijo.

Zasuk

Zasuk prav tako prepreči uspešno odkritje poddvojenih regij, saj se slikovnim elementom znotraj pripadajočih slikovnih izsekov zaradi ponovnega vzorčenja spremeni svetlost. Problematičen je že zasuk za vsega nekaj kotnih stopinj. Pri obeh primerih na sliki 3.11 je pripadajoča dobljena slika podvojitev neuporabna.



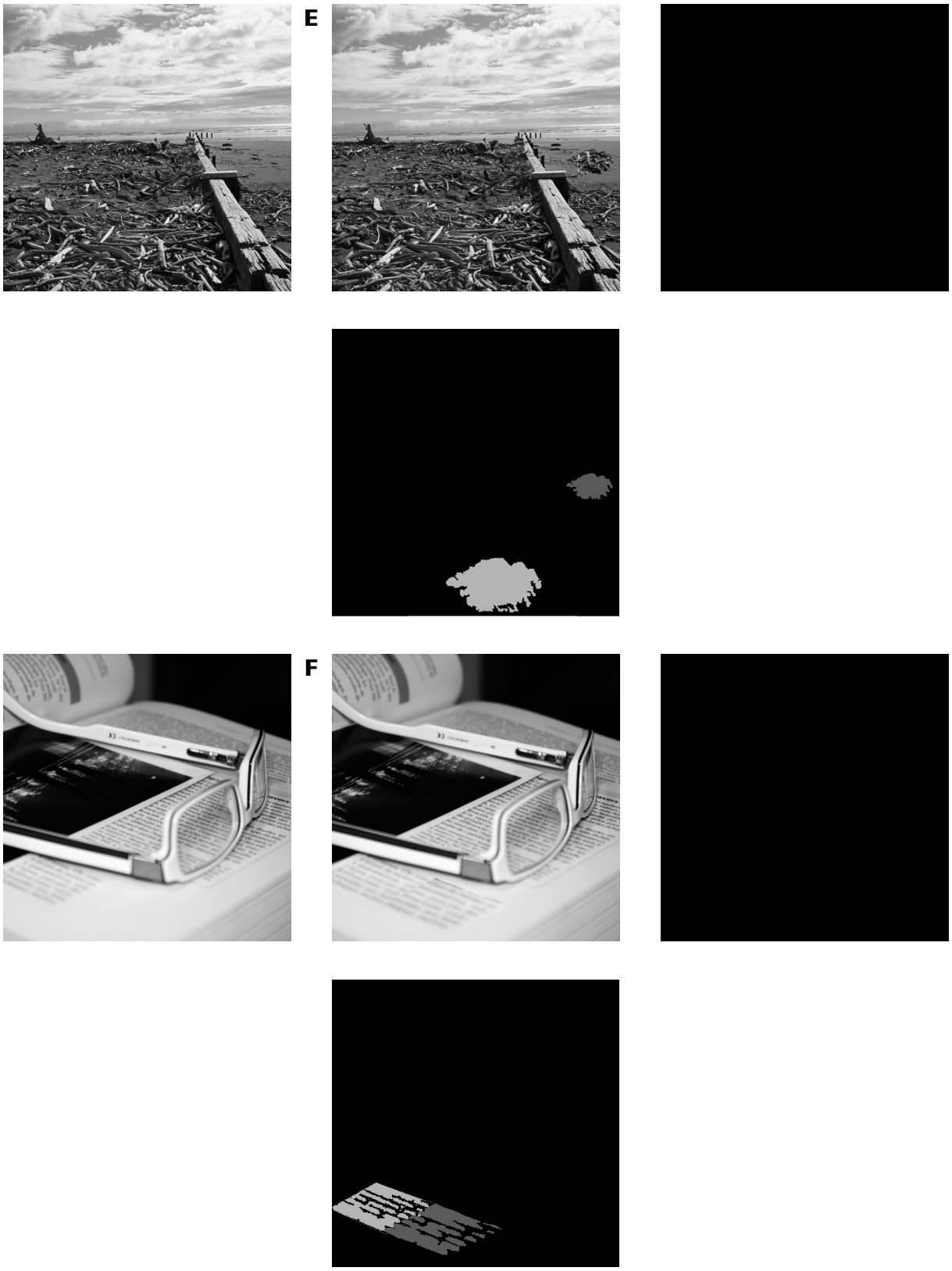
(nadaljevanje slike na naslednji strani)



Slika 3.11: Primera slik z zasukano podvojeno regijo.

Razteg oziroma skrčenje

Operaciji raztega in skrčenja zaradi ponovnega vzorčenja bistveno spremenita vsebino pripadajočih slikovnih izsekov, zato je algoritem DRD v teh okoliščinah neuspešen.



Slika 3.12: Primera slik s skrčeno oziroma raztegnjeno podvojeno regijo.

3.2 UČINKOVITOST

V tem podpoglavju bomo predstavili, kako vhodni podatki[9] vplivajo na porabo prostora in časa algoritma DRD med izvajanjem. Njegovo prostorsko in časovno zahtevnost smo že ugotovili. Dodatno pa je koristno vedeti, kakšen učinek imajo lastnosti (vsebina) preiskovane slike na prostor in čas, ki ga algoritem DRD dejansko porabi. Podajmo nekaj izsledkov, ki smo jih pridobili med preizkušanjem algoritma.

3.2.1 PREIZKUSNO OKOLJE IN PREIZKUSNI PRIMERI

V izmerjenih velikostih prostora in časih je zajeto tako izvajanje vseh korakov algoritma DRD, kot so opisani v podpoglavju 2.2 DELOVANJE ALGORITMA DRD, kot tudi kopiranje začasnih podatkov iz delovnega pomnilnika na disk in obratno ter brisanje začasnih datotek po koncu zadnjega koraka. Vse preizkuse smo izvajali v 32-bitnem okolju JRE (Windows XP Professional Service Pack 3, Java 2 Platform Standard Edition Runtime Environment v1.5.0.130, 1 GB delovnega pomnilnika). Meritve smo opravljali na dveh različnih procesorskih platformah. Za parametre odkrivanja smo uporabili vrednosti iz tabele 3.1, pri čemer je imel parameter H_n vseskozi vrednost 2.

Za namen ocenjevanja porabe prostora in časa smo ustvarili dve preizkusni množici: osnovno in umetno.

Osnovna preizkusna množica je zajemala 36 slik velikosti 512×512 elementov, med katerimi je bilo 14 digitalnih fotografij, 7 slik je bilo pridobljenih z optičnim čitanjem fotografsko razvitih ali natisnjenih fotografij, za 15 slik pa način prenosta v digitalno obliko ni bil ugotovljen. V celotni preizkusni množici je bilo 29 slik pristnih, 7 pa spremenjenih z uporabo podvojevanja regij.

Umetna preizkusna množica je zajemala 3 umetno pridobljene slike velikosti 512×512 elementov: mrežo vodoravnih črt debeline enega slikovnega elementa, sliko s približkom belega šuma in še eno sliko s približkom belega šuma, v kateri pa je bila podvojena regija.

3.2.2 VPLIV NA PORABO PROSTORA

Med opravljenimi preizkusi nismo merili natančne porabe pomnilnika. Tovrstne meritve so ponavadi razmeroma zapletene, zato smo se zadovoljili z ocenami oziroma izmerjenimi približki. Odvisnosti med vhodnimi podatki in porabo prostora smo poiskali z analizo posameznih korakov algoritma.

Vpliv na porabo delovnega pomnilnika

Na od vhodnih podatkov odvisno porabo delovnega pomnilnika vplivata dve vrednosti, ki se od slike do slike razlikujeta:

- velikost vektorja značilnic

9 S tem izrazom je mišljena vsebina vhodne slike in ne njena velikost.

Ta vrednost vpliva na skupno velikost poenostavljenih predstavitev vseh slikovnih blokov, ki se morajo hraniti v pomnilniku, dokler ni natančno določena množica kandidatov. Vektor značilnic je tem manjši, čim bolj so si bloki preiskovane slike medseboojno podobni in čim manj podrobnosti vsebujejo. Formula za izračun porabe delovnega pomnilnika (M_1), na katero vpliva velikost vektorja značilnic (f), je naslednja:

$$M_1 = (\sqrt{(N)} - \sqrt{(b)} + 1)^2 * f \tag{3.1}$$

V njej pomeni oznaka N število slikovnih elementov v preiskovani sliki, oznaka b pa število slikovnih elementov v bloku.

Za privzeto vrednost velikosti blokov (64) je na obeh preizkusnih množicah od vektorja značilnic odvisna poraba pomnilnika znašala najmanj 249 kB (za primer f=1) in največ 16 MB (za primer f=64)[10].

- velikost dvojiškega iskalnega drevesa

  Velikost dvojiškega iskalnega drevesa (število vozlišč) je tem večja, čim več je različnih medsebojnih dvodimenzionalnih odmikov parov slikovnih izsekov, ki predstavljajo posamezne kandidate. Formula za izračun porabe delovnega pomnilnika (M_2), na katero vpliva število vozlišč drevesa (N_{BST}), je naslednja:

$$M_2 = (S_{key} + S_{val} + S_{lref} + S_{rref} + S_{obj}) * N_{BST} \tag{3.2}$$

  V njej pomenijo spremenljivke S_{key}, S_{val}, S_{lref} in S_{rref} ter S_{obj} po vrsti velikost pomnilnika, ki ga zasedajo ključ (4 B), vrednost v vozlišču (4 B), referenci na levega in desnega naslednika (dvakrat 4 B) in sistemski javanski podatki, ki so potrebni za predstavitev vsakega objekta (8 B). V okolju JRE torej vsako vozlišče zasede 24 bajtov.

  Na osnovni preizkusni množici je od dvojiškega iskalnega drevesa odvisna poraba pomnilnika znašala najmanj 627 kB (za primer N_{BST}=26.761) in največ 4 MB (za primer N_{BST}=153.345)[11]. Na umetni preizkusni množici je ta poraba znašala najmanj 6 kB (za primer N_{BST}=254) in največ 881 kB (za primer N_{BST}=37.569).

Ocenjena skupna poraba delovnega pomnilnika je pri preizkusnih slikah presegla 750 MB. Primerjava med skupno in od vhodnih podatkov odvisno porabo kaže, da vhodni podatki zanemarljivo vplivajo na skupno porabo delovnega pomnilnika.

Vpliv na porabo sklada

Na porabo prostora na skladu najbolj vplivajo rekurzivni klici. V izvedbi algoritma DRD se rekurzija uporablja na dveh mestih:

- pri leksikografskem urejanju seznama poenostavljenih predstavitev blokov (urejanje s porazdelitvami) in

10 Vrednosti porabe pomnilnika so zaokrožene na cela števila.

11 Številske vrednosti porabe pomnilnika so navzgor zaokrožene.

- pri odstranjevanju nepravih kandidatov prvega tipa (operacije na dvojiškem iskalnem drevesu).

Pri leksikografskem urejanju imamo opravka z zelo velikim seznamom. Pri velikosti slik 512×512 elementov in velikosti blokov 8×8 elementov seznam vsebuje 255.025 elementov. Globina rekurzije pri urejanju je zato v določenih primerih precejšnja. Poleg velikosti nanjo vpliva tudi stopnja obstoječe urejenosti elementov. Globina je še največja takrat, ko so elementi bodisi medsebojno enaki bodisi izrazito neurejeni. V prvem primeru algoritem urejanja opravi številne nepotrebne zamenjave (tipičen primer je slika z mrežo vodoravnih črt). V drugem primeru algoritem urejanja opravi številne potrebne zamenjave (tipičen primer je slika s približkom belega šuma). Na obeh preizkusnih množicah je bila najmanjša izmerjena globina rekurzije 539, največja 127.766, povprečna pa je znašala 24.890.

Pri odstranjevanju nepravih kandidatov prvega tipa uporabljamo dvojiško iskalno drevo. To drevo je pri vseh preizkusnih slikah razmeroma majhne globine (neizrojeno), zato je temu primerna tudi globina rekurzije. Na preizkusnih slikah je bila največja izmerjena globina dreves 253, najmanjša 40, povprečna pa je znašala 56.

Z vidika porabe sklada je torej najbolj zahtevno urejanje s porazdelitvami.

Meritev za ugotavljanje porabe sklada smo zelo poenostavili. Za vsako preizkusno sliko smo namreč ugotavljali, ali za izvedbo algoritma zadošča velikost sklada 1, 5, 10 ali 50 MB. Za 20 slik iz osnovne preizkusne množice je tako zadoščala velikost 1 MB, za 15 slik 5 MB in za eno sliko 10 MB. V skladu s pričakovanji smo za vse slike iz umetne preizkusne množice potrebovali sklad velikosti 10 MB.

V nasprotju s porabo delovnega pomnilnika je torej poraba sklada precej odvisna od vhodnih podatkov.

Sklep o porabi pomnilniškega prostora

Preizkusi so pokazali, da vhodni podatki v praksi bistveno ne vplivajo na porabo delovnega pomnilnika. Ta lastnost je pomembna za morebitno praktično uporabo algoritma DRD. Poraba sklada je – čeprav močno odvisna od vhodnih podatkov – v primerjavi s celotno porabo pomnilniškega prostora zanemarljiva. Z izsledki preizkusov ocenjena poraba prostora torej potrjuje formalno ocenjeno prostorsko zahtevnost algoritma DRD, ki je podana v podpoglavju 2.3 RAČUNSKA ZAHTEVNOST ALGORITMA DRD.

3.2.3 VPLIV NA PORABO ČASA

Meritve potrebnega časa smo opravljali za vsako preizkusno množico posebej. Tabeli 3.2 in 3.3 prikazujeta izmerjene izvajalne čase algoritma DRD z izvedenim programom v preizkusnem okolju na obeh procesorskih platformah.

Tabela 3.2: Časi izvajanja algoritma DRD z izvedenim programom na osnovni preizkusni množici.

Procesorska platforma	**Povprečni čas**	**Najkrajši čas**	**Najdaljši čas**
Intel Pentium 4, 2,66 GHz	3 m 40 s	1 m 08 s	22 m 14 s
AMD Athlon 64 3000+, 1.800 MHz	3 m 37 s	1 m 07 s	20 m 16 s

Tabela 3.3: Časi izvajanja algoritma DRD na umetno pridobljenih slikah.

Procesorska platforma	**Čas izvajanja**		
	Horizontal[12]	*Noise*	*Noise_tampered*
Intel Pentium 4, 2,66 GHz	8 m 59 s	51 m 07 s	49 m 27 s
AMD Athlon 64 3000+, 1.800 MHz	9 m 02 s	36 m 44 s	35 m 24 s

Izvajanje v preizkusnem okolju je za množično uporabo v praksi prepočasno. Razlogi za nizko hitrost so obravnavani v razdelku Pohitritev algoritma poglavja 4 in razdelku IZBIRA PROGRAMSKEGA JEZIKA dodatka B. Iz rezultatov tudi sledi, da je poraba časa precej bolj odvisna od vhodnih podatkov kot poraba prostora. Sliki s približkom belega šuma sta na primer znatno upočasnili izvajanje.

Dejavniki, ki vplivajo na porabo časa

Meritve so pokazale, da na čas izvajanja algoritma DRD najbolj vpliva prvi od treh korakov, ki imajo formalno največjo časovno zahtevnost: to je leksikografsko urejanje vrstic matrike poenostavljenih predstavitev blokov. Druga dva koraka, vstavljanje v dvojiško iskalno drevo in iskanje v njem, sta v praksi zaradi precej manjšega števila rekurzivnih klicev bistveno hitrejša[13].

Na dejansko trajanje urejanja s porazdelitvami v primeru algoritm DRD vplivata dva dejavnika:

- velikost vektorja značilnic

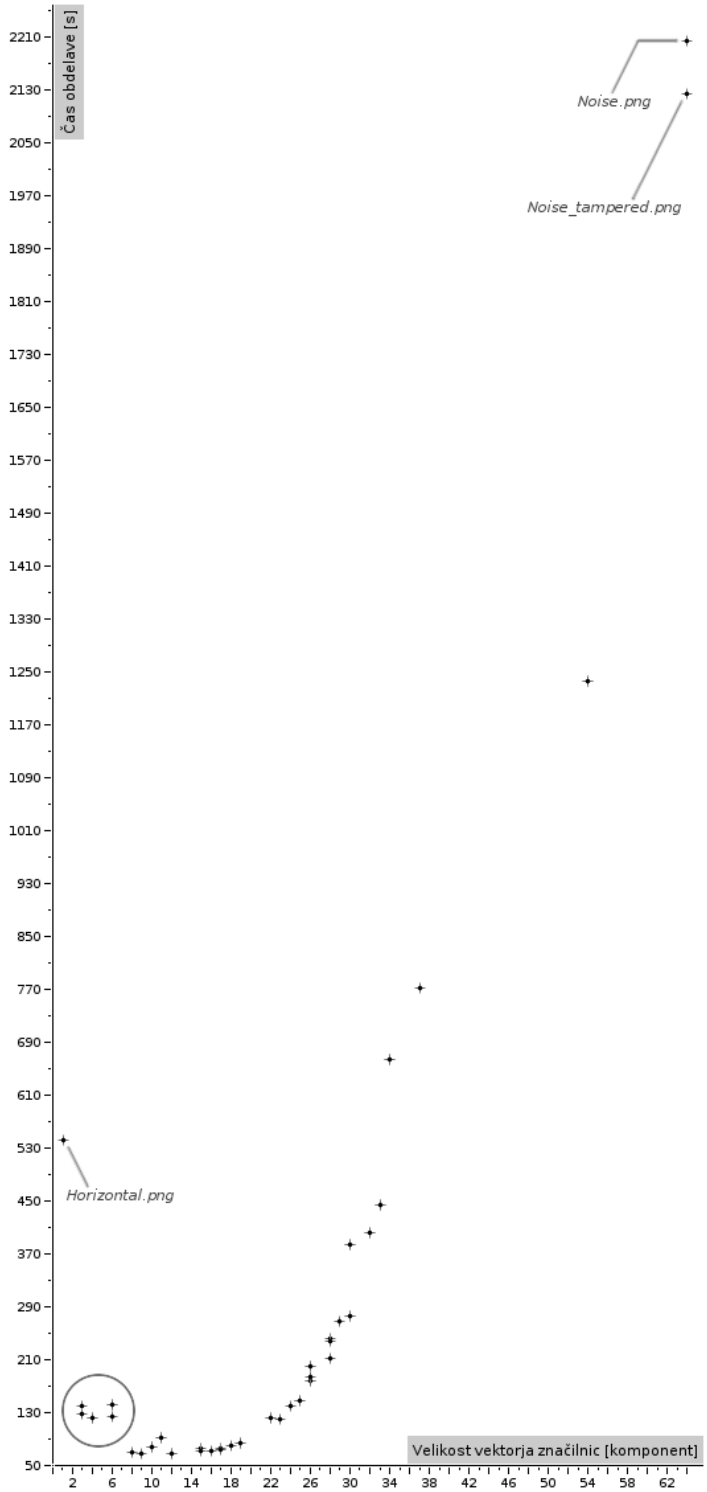
12 *Horizontal* je slika z mrežo vodoravnih črt, *Noise* slika s približkom belega šuma in *Noise_tampered* slika s približkom belega šuma in podvojeno regijo. Prikazane so na sliki 3.14.

13 Števila rekurzivnih klicev ne smemo enačiti z globino rekurzije. Medtem ko sta ti dve vrednosti pri operacijah na dvojiškem iskalnem drevesu enaki, je pri urejanju s porazdelitvami število rekurzivnih klicev ponavadi občutno večje od globine rekurzije.

Ta vrednost določa velikost zapisov poenostavljenih predstavitev blokov in s tem trajanje njihovih primerjav. Pogojena je s stopnjo podrobnosti v sliki in medsebojno podobnostjo oziroma različnostjo slikovnih izsekov, ki določajo bloke. Največjo velikost vektorja značilnic data umetno pridobljeni sliki s približkom belega šuma (*Noise*, *Noiste_tampered*). V osnovni preizkusni množici pa dajo največje vektorje značilnic slike *Brick* (54), *HanyFarid-hike* (34) in *Cambodia* (33), najmanjše pa slike *Pencils* (3), *Manca* (3) in *Boats* (4). Vse slike, omenjene v tem razdelku, so prikazane na sliki 3.14.

- stopnja obstoječe urejenosti seznama poenostavljenih predstavitev blokov

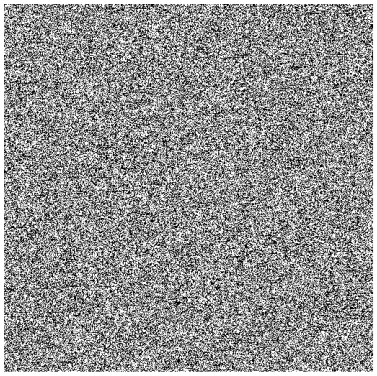
  Kot smo že omenili, medsebojna enakost ali izrazita neurejenost poenostavljenih predstavitev povzroči veliko število rekurzivnih klicev. Pri velikih vektorjih značilnic obe lastnosti izvirata iz vhodne slike (izrazita neurejenost pri slikah *Noise* in *Brick*). Pri zelo majhnih vektorjih značilnic lahko prvo lastnost tudi "pridobimo" in sicer s poenostavitvijo blokov (*Manca*, *Pencils*, *Boats*). Slika *Horizontal* že pred poenostavitvijo vsebuje medsebojno skoraj povsem enake bloke.

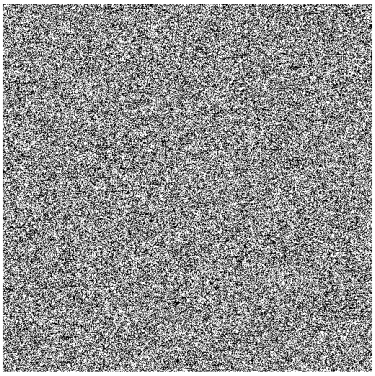


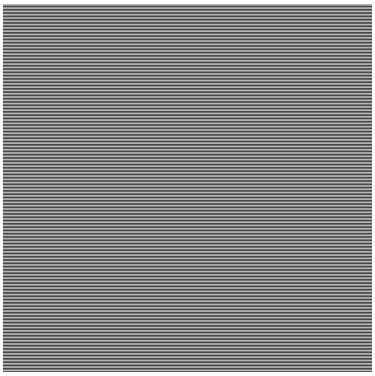
Slika 3.13: Odvisnost celotnega časa obdelave od velikosti vektorja značilnic.

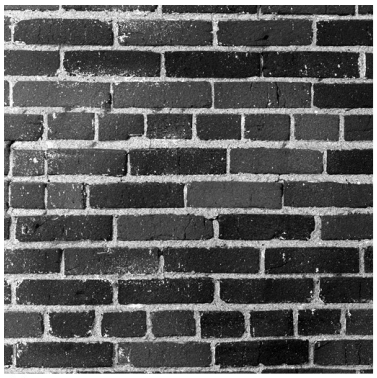
Graf odvisnosti na sliki 3.13 prikazuje, kako je čas odkrivanja podvojenih regij pogojen z velikostjo vektorja značilnic. Obkrožene točke na grafu pripadajo slikam, pri katerih smo medsebojno enakost blokov "pridobili" med poenostavitvijo blokov

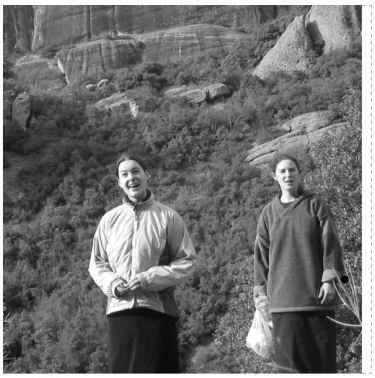
(*Manca*, *Pencils*, *Boats*, *Trnovo*, *ChopardCasran*). Pri teh celotni čas obdelave odstopa od splošnega trenda, ki je viden na grafu.


Noise

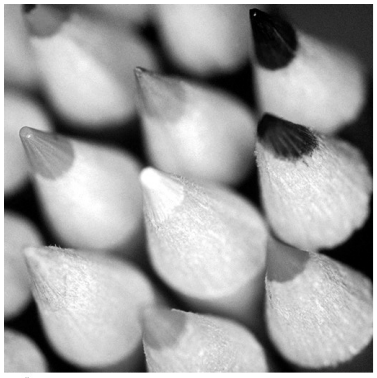

Noise_tampered


Horizontal


Brick


HanyFarid-hike


Cambodia


Pencils


Manca


Boats

(nadaljevanje slike na naslednji strani)

Slika 3.14: Podmnožica preizkusnih primerov, ki so bili uporabljeni za ugotavljanje učinkovitosti algoritma DRD.

Sklep o porabi časa

Vhodni podatki lahko torej znatno vplivajo na trajanje odkrivanja podvojenih regij z algoritmom DRD. Največji vpliv ima velikost vektorja značilnic, ki je najbolj odvisna od količine podrobnosti v preiskovani sliki.

Poglavje 4

Sklep in iztočnice za nadaljnje delo

Preizkusi, ki smo jih opravili za doseganje drugega in tretjega cilja iz uvodnega poglavja, nam omogočajo, da ovrednotimo algoritem DRD: uspešnost njegovega odkrivanja podvojenih regij in njegovo odpornost na izbrane postopke digitalne obdelave.

Po izvedenih preizkusih sodeč algoritem uspe odkriti podvojene regije zadostne velikosti v ponarejenih slikah in takšne slike obravnavati kot ponaredke. S pravilno nastavitvijo parametra H_n, ki pa je odvisna od posamezne preiskovane slike, natančno označi meje podvojenih regij. Algoritem je robusten v tem smislu, da pod določenimi pogoji uspe pravilno obravnavati tudi primere, ki so bili po podvajauju regij podvrženi eni od treh operacij digitalne obdelave. Ti pogoji so odvisni od posamezne preiskovane slike in predstavljajo:

- za shranjevanje v izgubni zapis JPEG: doseganje ali preseganje določene kakovosti zapisa JPEG. Za preizkusne primere je povprečna vrednost tega praga znašala 80%.
- za dodajanje aditivnega belega šuma: doseganje ali preseganje določenega razmerja SNR. Za preizkusne primere je povprečna vrednost tega praga znašala 25 dB.
- za filtriranje s filtrom mediana: "ne-doseganje" določene velikosti polmera filtrirnega okna. Za preizkusne primere je povprečna vrednost tega praga znašala 2.

Če je dobra lastnost algoritma DRD robustnost, pa je njegova slabost ta, da so vrednosti parametrov odkrivanja (teh je skupaj 6) določene vnaprej in torej neodvisne od lastnosti preiskovane slike. Preizkusi so pokazali, da morajo biti za zadovoljivo pravilnost obravnave obeh vrst primerov (pozitivnih in negativnih), kar želimo doseči, vrednosti teh parametrov omejene z obeh strani. To pa pomeni, da na dovolj veliki populaciji slik ne moremo povečati uspešnosti algoritma DRD, ne da bi pri določanju teh vrednosti upoštevali lastnosti samih preiskovanih slik.

Možne nadgraditve in izboljšave algoritma DRD lahko razdelimo na tri sklope:

- razširitve algoritma v smeri bolj splošne uporabnosti,
- izboljšave algoritma v smeri večje natačnosti ter uspešnosti in
- pohitritev izvajanja programa, s katerim je izveden algoritem.

Razširitve algoritma

Za praktične namene bi morali najprej povečati prag omejitve velikosti za vhodne slike in odpraviti zahtevo po kvadratni obliki. Nadalje bi bilo koristno upoštevati predloga za izboljšanje uporabnosti algoritma, ki ju je Popescu navedel v [20].

Prva izboljšava uporabnosti bi omogočila odkrivanje podvojenih regij tudi v barvnih slikah. Tega bi se lahko lotili na dva načina:

- z obravnavo vsake barvne ravnine posebej

  S tem pristopom bi najprej obravnavali vsako barvno ravnino kot navadno sivinsko sliko in tako pridobili tri vmesne slike podvojitev. Potem bi s primerno operacijo določili "presek" vmesnih slik, ki bi nam predstavljal končno sliko podvojitev.

- s hkratno obravnavo vseh treh barvnih ravnin

  Po tej medoti bi preiskovano sliko razdelili na bloke tako, da bi v vektor, s katerim je predstavljen nek blok, za vsak slikovni element bloku pripadajočega slikovnega izseka zaporedoma zapisali vse tri njegove barvne komponente. Primerjavo blokov bi potem izvedli na povsem enak način kot pri osnovnem algoritmu.

Druga izboljšava bi omogočila odkrivanje regij, ki so bile izbrane v eni sliki in kopirane v drugo sliko. Ta problem je enostavno rešljiv z uvedbo podpore slikam pravokotne oblike in predhodno združitvijo para preiskovanih slik v eno samo.

Predlagani razšitivi seveda nista kakovostni izboljšavi, ki bi nam dali natančnejše rezultate.

Izboljšave algoritma

Kakovostno bi lahko algoritem izboljšali tako, da bi vrednosti parametrov odkrivanja sproti prilagajali glede na vhodne podatke. Pri tem bi lahko upoštevali različne lastnosti preiskovanih slik, kot na primer velikost vektorja značilnic, ki ga dobimo po metodi PCA, ali povprečje razmerja med signalom in šumom v celotni sliki.

Glede na karakteristike slik bi se lahko prilagodila:

- vrednosti parametrov b in ε.
- vrednost parametra H_n: za bolj šumne slike bi vrednost povečali.
- vrednost parametra H_f: za slike z majhnim vektorjem značilnic bi vrednost povečali in obratno.

Predlog možnega postopka za določanje ustrezne vrednosti parametru H_f je na primer podan v [2]. V istem delu je opisan tudi postopek, ki v enem koraku doseže boljši učinek, kot ponovljeno izvajanje erozije in dilatacije v koraku 6 algoritma DRD.

Pohitritev algoritma

Pri izvedba algoritma DRD v okviru te diplomske naloge nismo posvečali pretirane pozornosti časovni učinkovitosti algoritma, bolj so nas zanimale preglednost, prenosljivost in uspešnost odkrivanja. Hitrost izvajanja algoritma zato ni na zavidljivi ravni. V dodatku B so navedeni razlogi, zakaj smo se odločili za izvedbo v javi, ki je glavni "krivec" za počasnejše delovanje.

Za uspešno uporabo izvedbe algoritma v praksi bi bilo potrebno bodisi njeno javansko izvorno kodo prevesti v strojni jezik z uporabo ustreznih orodij bodisi algoritem izvesti v enem izmed programskih jezikov, ki so že po zasnovi prevedljivi. Primera takšnih jezikov sta vsem dobro znana C++ in C#. Z analizo in nadaljnjimi optimizacijami izvorne kode, na primer z uporabo **urejanja s kopico** (*heapsort*) namesto urejanja s porazdelitvami, bi bilo možno še dodatno pohitriti izvajanje programa. Nenazadnje bi lahko na ustrezno opremljenih sistemih izvajanje določenih delov algoritma izdatno pospešili z uporabo grafičnega procesorja, ki omogoča splošnonamensko uporabo.

Dodatek A

O metodi analize glavnih komponent

Metodo analize glavnih komponent (PCA) poznamo, glede na področje uporabe, pod več imeni: **diskretna Karhunen-Loèvejeva transformacija**[14] (*discrete Karhunen-Loève transform*), **Hotellingova transformacija** (*Hotelling transform*) in **pravi ortogonalni razcep** (*proper orthogonal decomposition*) [27]. To je linearna transformacija, ki preslika podatke oziroma njim pripadajoče vektorje iz višjerazsežnega prostora v nižjerazsežen prostor, pri čemer maksimizira varianco v podatkih in hkrati minimizira srednjo kvadratno oziroma rekonstrukcijsko napako. Metoda PCA torej izvaja operacijo **zmanjšanja razsežnosti** (*dimensionality reduction*) podatkov. Poenostavljeno povedano lahko zmanjšanje razsežnosti izvedemo tako, da v podatkih najprej poiščemo zakonitost oziroma notranjo strukturo, ki najbolje opisuje njihovo raznolikost. Potem na podlagi te strukture določimo poenostavljene predstavitve podatkov. V poenostavljenih predstavitvah so zajete bistvene značilnosti prvotnih podatkov, mednje pa sodijo tako medsebojne razlike kot tudi podobnosti.

PRIDOBITEV POENOSTAVLJENIH PREDSTAVITEV

Koraki postopka metode PCA, ki nas pripelje do poenostavljenih predstavitev podatkov, so naslednji:

1. Od vsake koordinate vsakega vhodnega vektorja $\vec{r}_i$ odštej povprečno vrednost te koordinate preko vseh vhodnih vektorjev, zbranih v matriki **R**:

$$\mathbf{R} = [\vec{r}_1 \quad \vec{r}_2 \quad \ldots \quad \vec{r}_{N_b}] \tag{A.1}$$

$$\vec{x} = \frac{\vec{r}_1 + \vec{r}_2 + \ldots + \vec{r}_{N_b}}{N_b} \qquad \mathbf{X} = [\vec{x} \quad \vec{x} \quad \ldots \quad \vec{x}]$$

$$\mathbf{P} = \mathbf{R} - \mathbf{X}$$

14 Pravzaprav se metoda PCA rahlo razlikuje od diskretne Karhunen-Loèvejeve transformacije pri merilih za vhodne podatke. Metoda PCA zahteva, da so povprečne vrednosti posameznih komponent podatkov (koordinat vektorjev) preko celotne vhodne množice enake 0, medtem ko pri Karhunen-Loèvejevi transformaciji tega pogoja ni.

Rezultat tega koraka je matrika prilagojenih vektorjev **P**. Povprečna vrednost vsake vektorske koordinate preko vseh prilagojenih vektorjev je enaka 0.

2. Za matriko prilagojenih vektorjev izračunaj kovariančno matriko:

$$\mathbf{K} = cov(\mathbf{P}) \tag{A.2}$$

3. Izračunaj **lastne vrednosti** (*eigenvalues*) λ_i in **lastne vektorje** (*eigenvectors*) $\vec{v_i}$ kovariančne matrike. Lastni vektorji in lastne vrednosti zadoščajo naslednji enačbi:

$$\mathbf{K}\vec{v_i} = \lambda_i\vec{v_i} \qquad i \in [1, \ldots, b] \tag{A.3}$$

4. Uredi lastne vektorje po velikosti pripadajočih lastnih vrednosti.

5. Za p največjih lastnih vrednosti izberi pripadajoče lastne vektorje in z njimi sestavi matriko **F**:

$$\mathbf{F} = [\vec{v_1} \quad \vec{v_2} \quad \ldots \quad \vec{v_p}] \tag{A.4}$$

Matriko **F** setavljeno iz izbranih lastnih vektorjev imenujemo **vektor značilnic** (*feature vector*).

6. Transponiraj vektor značilnic.

7. S transponiranim vektorjem značilnic z leve pomnoži matriko prilagojenih vhodnih vektorjev:

$$\mathbf{A} = \mathbf{F}^{\mathrm{T}} \times \mathbf{P} \tag{A.5}$$

Rezultat tega koraka je matrika izhodnih vektorjev. Razsežnost prostora izhodnih vektorjev je enaka p. Izhodni vektorji zajemajo bistvene značilnosti podatkov, ki so bili zapisani z vhodnimi vektorji.

OBNOVITEV PRVOTNIH PREDSTAVITEV PODATKOV

Pridobljene poenostavljene predstavitve podatkov ne vsebujejo vse informacijske vsebine izvirnih podatkov. Izvirnih blokov zato ne moremo natančno obnoviti, obnovimo pa lahko njihove približke in seveda velikost (razsežnost) zapisov, s katerimi so bili prvotno predstavljeni.

Za obnovitev približkov izvornih podatkov uporabimo naslednji postopek:

1. Z vektorjem značilnic z leve pomnoži matriko izhodnih vektorjev:

$$\mathbf{P}_{\mathrm{a}} = \mathbf{F} \times \mathbf{A} \tag{A.6}$$

2. Rezultatu prejšnjega koraka prištej matriko **X** z vektorji povprečnih vrednosti koordinat preko vseh vhodnih vektorjev $\vec{r_i}$. Matrika **X** ima vsebino, ki je bila uporabljena v formuli A.1:

$$\mathbf{R}_{\mathrm{a}} = \mathbf{P}_{\mathrm{a}} + \mathbf{X} \tag{A.7}$$

Dobljena matrika $\mathbf{R}_{\mathrm{a}}$ je matrika približkov vhodnih vektorjev.

UPORABA METODE PCA

Metoda PCA se običajno uporablja za ugotavljanje povezav in zakonitosti v podatkih ter druge analitične in primerjalne namene. Z njo lahko na primer ugotavljamo podobnosti med digitalnimi slikami. Ker so zapisi poenostavljenih predstavitev podatkov krajši od zapisov vhodnih podatkov, jo lahko uporabimo tudi za stiskanje podatkov. Takó metodo PCA uporabljajo nekateri izgubni algoritmi za stiskanje.

ČASOVNA ZAHTEVNOST METODE PCA

Metoda PCA je tako časovno kot prostorsko razmeroma zahtevna. Pri vhodni množici z n elementi (n slikovnimi bloki v primeru algoritma DRD), katerih razsežnost je enaka d (d slikovnih elementov v posameznem bloku v primeru algoritma DRD), je časovna zahtevnost metode enaka manjši od dveh vrednosti $O(n^3)$ in $O(d^3)$.

Težave s preveliko časovno zahtevnostjo metode PCA poskušajo raziskovalci zaobiti z zaporednim izvajanjem postopka na manjših podmnožicah vhodne množice [18], z učinkovitejšim iskanjem lastnih vektorjev [25, 10], pojavile pa so se tudi druge rešitve [1, 19].

Dodatek B

O razvojnem okolju in izvedbi

IZBIRA PROGRAMSKEGA JEZIKA

Opisani algoritem je izveden v programskem jeziku java. Za javo smo se odločili na podlagi ugotovljene perspektivnosti, prednosti in številnih pozitivnih ocen tega jezika. Med razvojem programa se je pokazalo, da je bila izbira jave pravilna zaradi vsaj dveh vidikov. Prvi je ta, da je to zelo čist in pregleden jezik, izvorna koda pa je lažje berljiva kot pri večini drugih objektnih in postopkovnih programskih jezikov. S tem smo olajšali morebitno ponovno uporabo kode in razširjanje programa. V ta namen smo se pri programiranju trudili karseda upoštevati priporočila za oblikovanje besedilnih datotek z izvorno kodo, vključno z dodajanjem smiselnih komentarjev na potrebnih mestih. Priporočila je objavilo podjetje Sun Microsystems, Inc. [26]. Drugi vidik je ta, da je program v celoti prenosljiv in takoj uporaben na vseh podprtih platformah.

RAZVOJNO OKOLJE

Brez integriranih razvojnih okolij si danes ne znamo več predstavljati razvoja aplikacij. Čeprav je program relativno majhen, smo se zaradi možnosti lažjega in hitrejšega razvoja odločili za uporabo javanskega razvojnega okolja. Izbrali smo brezplačno orodje JCreator LE podjetja Xinox Software. JCreator LE se učinkovito povezuje z okoljem Java 2 Platform Standard Edition Development Kit (JDK) in tako omogoča enostavnejši in hitrejši razvoj javanskih programov. Orodje podpira večino funkcij, ki jih najdemo v vseh integriranih razvojnih okoljih, na primer označevanje sintakse, postavitev profilov za prevajanje in izvajanje, napredno iskanje ter hierarhični prikaz paketov, razredov, spremenljivk in metod.

PREGLEDNOST IZVORNE KODE

Razvoj programa je potekal z upoštevanjem principa modularnosti in z zagotavljanjem preglednosti izvorne kode. Zaradi preobsežnosti bi bila izvedba z

izvorno kodo v enem samem javanskem paketu nepregledna. Izvorna koda aplikacije je tako razdeljena na štiri glavne in dva pomožna paketa, v katerih so smiselno združeni objekti in metode, ki pomensko spadajo skupaj:

- *main*

  Paket vsebuje glavni program z definicijami vseh konstant in naštevnih podatkovnih tipov. Tu so definirani glavna izvajalna nit in metode za sprotne izpise, pisanje v dnevniško datoteko, pomoč pri razhroščevanju, pretvorbo med enorazsežnimi in dvorazsežnimi koordinatami blokov in slikovnih elementov in za druge pomožne namene.

- *fileio*

  Paket vsebuje vse metode za branje in zapisovanje datotek, razen metod za pisanje v dnevniško datoteko. Slednje se uporabljajo v vseh glavnih paketih, zato so definirane v paketu *main*. Metode definirane v paketu *fileio* pa so namenjene branju datoteke s parametri, branju vhodne slike, zapisovanju izhodnih slik, branju paketne datoteke s seznamom slik in zapisovanju ter branju začasnih datotek s pari blokov.

- *drdperformer*

  Paket vsebuje definicije vseh matrik, drugih struktur in metod, ki se uporabljajo v glavnem algoritmu, vključno s tvorbo izhodnih slik. Poleg tega so v njem še definicije dvojiškega iskalnega drevesa in metod uporabljenih na njem ter definicije metod algoritma **urejanja s porazdelitvami** (*quicksort*).

- *gui*

  V tem paketu so definirani razredi, konstante, spremenljivke in metode, uporabljene pri izvedbi grafičnega uporabniškega vmesnika programa.

- *coordinates*

  Paket vsebuje metode za pretvorbo dvorazsežnih koordinat podatkovnih objektov (koordinat slikovnih elementov znotraj bloka, koordinat blokov znotraj slike) v enorazsežne in obratno. Enorazsežne koordinate so potrebne za zapisovanje v polje, ki hrani slikovne elemente vseh slikovnih blokov.

- *drdexception*

  Paket vsebuje definicijo razreda *DRDException*, ki je razširitev standardnega javanskega razreda *Exception*, namenjenega obravnavi napak med izvajanjem.

PRILAGODLJIVOST

Pri razvoju aplikacije smo posebno pozornost posvečali prilagodljivosti. Vrednosti vseh številskih in drugih parametrov nismo uporabili neposredno, pač pa smo jih določili s pomočjo konstant. V praksi so te konstante predstavljene z javanskimi spremenljivkami s stalno vsebino. Vsebina spremenljivk se lahko popravi pred ponovnim prevajanjem izvorne kode. Na ta način je zagotovljena visoka

prilagodljivost in lažja uporaba aplikacije pri nadaljnjem raziskovanju in nadgrajevanju. Imena objektov in metod, komentarji, zapisi v dnevniških datotekah ter oznake in izpisi v obeh uporabniških vmesnikih so v angleškem jeziku.

VGRAJENA UPORABNIŠKA VMESNIKA

Aplikacija je zasnovana tako, da omogoča tako interaktivno delo preko grafičnega uporabniškega vmesnika, kot tudi delo preko ukaznega uporabniškega vmesnika in zaporedno obdelavo zbirke slik s pomočjo lastnih ali standardnih (sistemskih) paketnih datotek. Interaktivno delo je prvenstveno namenjeno predstavitvi delovanja aplikacije in prikazu izvajanja glavnega algoritma. Ukazni vmesnik in možnost paketne obdelave sta namenjena lažjemu preizkušanju na množicah slik in morebitni kasnejši uporabi v praksi. V tem režimu dela se vsi izhodni podatki shranjujejo samodejno. Za sledenje izvajanju osnovnega algoritma in lažje odkrivanje napak smo v aplikacijo vgradili funkcionalnost za beleženje izvajanja sklopov algoritma in posameznih metod.

VHOD IN IZHOD

Vhodni podatki, ki jih aplikacija prebere, so naslednji:

- preiskovana slika

  To je slika, za katero bo algoritem preveril pristnost. V grafičnem režimu jo interaktivno izberemo v pogovornem oknu, v ukaznem režimu pa jo izberemo z navedbo argumenta v ukazni vrstici. V program je vgrajena podpora za branje slikovnih zapisov PNG in JPEG.

- parametri odkrivanja

  Aplikacija vedno prebere vrednosti parametrov iz posebne tekstovne datoteke, če ta obstaja, sicer pa uporabi privzete vrednosti. Pri obravnavi posameznih slik se mora datoteka s parametri nahajati v istem imeniku kot preiskovana slika. Pri uporabi aplikaciji lastnih paketnih datotek so za obdelavo vseh slik navedenih v posamezni paketni datoteki uporabljeni isti parametri. V teh primerih pa se mora datoteka s parametri nahajati v istem imeniku kot paketna datoteka.

- aplikaciji lastna paketna datoteka s seznamom slik za preiskovanje

  V ukaznem režimu lahko z navedbo ustrezne opcije določimo, naj aplikacija izvede postopek preverjanja pristnosti na seznamu slik, čigar imena so zapisana v posebni tekstovni datoteki, tako imenovani aplikaciji lastni paketni datoteki. Imena lahko vsebujejo absolutne ali relativne poti. Pot do datoteke je poljubna, datoteko s parametri odkrivanja pa aplikacija pričakuje v istem imeniku.

Izhodni podatki, ki jih aplikacija zapiše, so naslednji:

- slika podvojitev

  To je slika, na kateri so v dveh sivih odtenkih označene regije oziroma območja

na sliki, za katere algoritem domneva, da so podvojena. Program sliko podvojitev shrani v zapisu PNG.

- označena slika

  To je kombinacije preiskovane slike in slike podvojitev. Domnevno podvojena območja so obarvana s prosojno kontrastno barvo. Program označeno sliko shrani v zapisu PNG.

- dnevniška datoteka

  To je tekstovna datoteka, ki pripada posamezni preiskani sliki. V njej so zapisani osnovni podatki o preiskani sliki, vrednosti parametrov, ki so se uporabili v algoritmu, ocena pomnilnika, ki ga bodo zasedle podatkovne strukture s slikovnimi podatki, osnovne statistike o blokih preiskane slike in imena izhodnih datotek.

- sprotni izpisi

  Izpisi na zaslon vsebujejo informacije o napredovanju izvajanja glavnega algoritma. Prikazujejo se v obeh režimih dela. Stopnja podrobnosti izpisov je nastavljiva.

UČINKOVITOST

Za doseganje potrebne učinkovitosti izvedbe je bilo potrebno med razvojem programa odpraviti nekaj problemov.

Zagotavljanje časovne učinkovitosti

Pri zgodnjem preizkušanju začetnih korakov algoritma DRD se je pokazalo, da bi bila lastna izvedba matričnih operacij odločno prepočasna ne samo pri uporabi v praksi, pač pa že pri osnovnem preizkušanju za potrebe diplomskega dela. Zato smo za delo z matrikami uporabili paket Matrix Toolkits for Java (MTJ). MTJ je javanska knjižnica, izdana pod licenco LGPL, z zelo učinkovito izvedbo metod za izvajanje osnovnih in zahtevnejših operacij nad matrikami. Med njimi so na primer iskanje korenov linearnih enačb, izračun lastnih vektorjev in lastnih vrednosti ter določanje singularnih matrik. Knjižnica je v praksi povsem zadostila kriteriju pričakovane hitrosti izvajanja.

Zagotavljanje prostorske učinkovitosti

Pri zahtevnejši digitalni obdelavi slik hitro naletimo na problem premajhnega bralno-pisalnega pomnilnika. Običajno k temu najbolj pripomore vmesna obdelava večjih oziroma večdimenzionalnih matrik.

Izvedba algoritma DRD preseže porabo 750 MB pomnilnika že pri obdelavi sivinskih slik velikosti 512×512 elementov in tipičnih vrednostih parametrov odkrivanja. To se zgodi med sestavljanjem množice kandidatov. Za izogibanje ostranjevanju in s tem dodatni upočasnitvi izvajanja smo prostorsko zahtevne korake (gradnja seznama

kandidatov, izločanje nepravih kandidatov) naredili večfazne; vsaka faza naenkrat obdeluje le del podatkov, medtem ko se celotna zbirka podatkov hrani samo na disku.

Seznam slik

Seznam tabel

Literatura

[1] M. Artač, M. Jogan, A. Leonardis, "Incremental PCA for On-line Visual Learning and Recognition", v zborniku *16th International Conference on Pattern Recognition*, Quebec City, Canada, avgust 2002, zv. 3, str. 781–784.

[2] M. K. Bashar, K. Noda, N. Ohnishi, H. Kudo, T. Matsumoto, Y. Takeuchi, "Wavelet-Based Multiresolution Features for Detecting Duplications in Images", v zborniku *IAPR Conference on Machine Vision Applications*, Tokyo, Japan, maj 2007, str. 264–267.

[3] J. A. Bloom, M. L. Miller, "Informed Detection Revisited", *Digital Watermarking*, Berlin, Germany: Springer Berlin / Heidelberg, 2005, str. 29–41.

[4] R. H. Chan, F. R. Lin, K. M. Yeung, "A Frequency Domain Based Watermarking Scheme with Spatial Repetition Coding", v zborniku *5th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando, USA, julij 2001, str. 35–40.

[5] T. K. Das, S. Maitra, "A Robust Block Oriented Watermarking Scheme in Spatial Domain", *Information and Communications Security*, Berlin, Germany: Springer Berlin / Heidelberg, 2002, str. 184–196.

[6] B. Eckel, *Thinking in Java*, 3. izdaja, Upper Saddle River, USA: Prentice Hall PTR, 2003.

[7] B. Fisher, S. Perkins, A. Walker, E. Wolfart. (2002, avgust). *Hypermedia Image Processing Reference*. Dostopno na spletnem naslovu[15]: http://www.dsi.unive.it/~atorsell/Hipr.pdf

[8] J. Flusser, "Moment Invariants in Image Analysis", v zborniku *World Academy of Science, Engineering and Technology*, Helsinki, Finland, februar 2006, zv. 11, str. 196–201.

[9] L. Guohui, W. Qiong, T. Dan, S. Shaojie, "A Sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on DWT and SVD", v zborniku *IEEE International Conference on Multimedia and Expo*, Beijing, China, julij 2007, str. 1750–1753.

[10] G. Henry, D. Watkins, J. Dongarra, "A Parallel Implementation of the Nonsymmetric QR Algorithm for Distributed Memory Architectures", *SIAM*

15 Spletni naslovi, ki so navedeni v seznamu literaturnih virov, so bili zadnjič preverjeni marca 2009.

Journal on Scientific Computing, št. 1, zv. 24, str. 284–311, 2002.

[11] A. Langille, M. Gong, "An Efficient Match-based Duplication Detection Algorithm", v zborniku *Third Canadian Conference on Computer and Robot Vision*, Quebec City, Canada, junij 2006, str. 64.

[12] Z. J. Lee, S. W. Lin, S. F. Su, C. Y. Lin, "A Hybrid Watermarking Technique Applied to Digital Images", *Applied Soft Computing*, št. 1, zv. 8, str. 798–808, 2008.

[13] E. T. Lin, E. J. Delp, "A Review of Fragile Image Watermarks", v zborniku *Multimedia and Security Workshop at ACM Multimedia '99*, Orlando, USA, oktober 1999, str. 25–29.

[14] B. Mahdian, S. Saic, "Blind Methods for Detecting Image Fakery", v zborniku *42nd Annual IEEE International Carnahan Conference on Security Technology*, Prague, Czech Republic, oktober 2008, str. 280–286.

[15] B. Mahdian, S. Saic, "Detection of Near-Duplicated Image Regions", *Computer Recognition Systems 2*, Berlin, Germany: Springer Berlin / Heidelberg, 2007, str. 187–195.

[16] P. Meerwald, A. Uhl, "A Survery of Wavelet-Domain Watermarking Algorithms", v zborniku *SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents III*, San Jose, USA, januar 2001, str. 505–516.

[17] A. M. Namboodiri, A. K. Jain, "Multimedia Document Authentication Using On-Line Signatures as Watermarks", v zborniku *SPIE, Security, Steganography, and Watermarking of Multimedia Contents VI*, San Jose, USA, januar 2004, str. 653–662.

[18] K. Nishino, S. K. Nayar, T. Jebara, "Clustered Blockwise PCA for Representing Visual Data", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, št. 10, zv. 27, str. 1675–1679, 2005.

[19] T. Oyama, S. Karungaru, S. Tsuge, Y. Mitsukura, M, Fukumi, "Incremental Learning Method of Simple-PCA", *Knowledge-Based Intelligent Information and Engineering Systems*, Berlin, Germany: Springer Berlin / Heidelberg, 2008, str. 403–410.

[20] A. C. Popescu, *Statistical Tools for Digital Image Forensics*, Dartmouth College, Hanover, New Hampshire, USA, tehniško poročilo, TR2005-531, december 2004.

[21] A. C. Popescu, H. Farid, "Statistical Tools for Digital Forensics", *Information Hiding*, Berlin, Germany: Springer Berlin / Heidelberg, 2005, str. 127–147.

[22] L. I. Smith. (2002, februar). A Tutorial on Principal Components Analysis. Dostopno na spletnem naslovu: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

[23] V. Solachidis, I. Pitas, "Circularly Symmetric Watermark Embedding in 2-D DFT Domain", *IEEE Transactions on Image Processing*, št. 11, zv. 10, str. 1741–1753, 2001.

[24] L. Tian, S. Kamata, "Near-Duplicate Detection Using a New Framework of Constructing Accurate Affine Invariant Regions", v zborniku *VISUAL2007 – 9th*

International Conference on Visual Information Systems, Shanghai, China, julij 2007, str. 61–72.

[25] Z. Zeng, X. Lin, T. Y. Li, "An Efficient Parallel Homotopy Algorithm for Nonsymmetric Eigenvalue Problems with O(N) Running Time", v zborniku *5th SIAM Conference on Parallel Processing for Scientific Computing*, Houston, USA, marec 1991, str. 29–33.

[26] Različni avorji. (1999). Code Conventions for the Java Programming Language. Sun Microsysytems, Inc. Dostopno na spletnem naslovu: http://java.sun.com/docs/codeconv

[27] Različni avtorji. (2009). Wikipedia. Wikimedia Foundation, Inc. Dostopno na spletnem naslovu: http://www.wikipedia.org