Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Gregor Čepin

Iskanje podobnih enobarvnih fragmentov stenskih poslikav

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Zoran Bosnić
Somentor: prof. dr. Bogdan Filipič

Ljubljana 2013

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00103/2013
Datum: 10.04.2013

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

Kandidat: **GREGOR ČEPIN**

Naslov: **ISKANJE PODOBNIH ENOBARVNIH FRAGMENTOV STENSKIH POSLIKAV**

AUTOMATIC SEARCH FOR SIMILAR UNICOLORED WALL PAINTING FRAGMENTS

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

Arheologi se pri svojem delu srečujejo s problemom restavriranja starih stenskih poslikav. Kandidat naj v diplomski nalogi obravnava problem avtomatskega iskanja podobnih enobarvnih fragmentov in njihovega združevanja v skupine. V diplomskem delu naj primerja več načinov njihovega iskanja in združevanja, delovanje pristopov pa naj preizkusi na izbrani praktični zbirki fragmentov poslikav.

Mentor:

doc. dr. Zoran Bosnić

Somentor:

prof. dr. Bogdan Filipič

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Gregor Čepin, z vpisno številko **63090008**, sem avtor diplomskega dela z naslovom:

Iskanje podobnih enobarvnih fragmentov stenskih poslikav

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Zorana Bosnića in somentorstvom prof. dr. Bogdana Filipiča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 7. septembra 2013 Podpis avtorja:

Zahvaljujem se prof. dr. Bogdanu Filipiču, doc. dr. Zoranu Bosniću in mag. Tei Tušar za pomoč pri pripravi diplomskega dela. Zahvaljujem se tudi družini za podporo med študijem.

Kazalo

Povzetek

Stare stenske poslikave so skozi čas odpadle s sten in se razdrobile na fragmente. Sestavljanje fragmentov v prvotno obliko je pri delu s pravimi kosi ometa zahtevno in dolgotrajno opravilo. S primerno programsko opremo ga lahko opravimo v digitalni obliki. V tem diplomskem delu obravnavamo enega od problemov, ki se pri tem pojavi, to je avtomatsko iskanje podobnih enobarvnih fragmentov, ki bi spadali poleg trenutno sestavljenih vzorčastih fragmentov, in njihovo združevanje v skupine. Primerjamo več načinov iskanja podobnih fragmentov in načine združevanja podobnih fragmentov v skupine. Delovanje preizkusimo na fragmentih z arheološkega najdišča pri Celju, ki izvira iz rimskega obdobja.

Abstract

Old wall paintings had over time fallen from the walls and broken into fragments. Reassembly of fragments using real chunks of plaster is a challenging and time-consuming task. With appropriate software it can be done in digital form. In this thesis we discuss one of the challenges that arise in the process, that is automatic search for similar unicolored fragments, which can be placed next to the currently composed patterned fragments, and their clustering. We compare several methods of searching for and clustering of similar fragments. We test these methods on fragments found at archeological site near Celje which originates from the Roman period.

Poglavje 1

Uvod

Med izkopavanji na arheološkem najdišču iz rimskega obdobja na območju Celja so našli veliko število fragmentov stenskih poslikav. Fragmenti so deli ometa, ki so odpadli s sten in so ob odkritju neurejeno ležali na tleh. Restavratorji bi radi slike, ki so jih ti fragmenti nekoč tvorili, sestavili nazaj v prvotno obliko. S tem bi pripomogli k ohranjanju kulturne dediščine in dobili zanimiv vpogled v tedanji čas. Zaradi velikega števila fragmentov stenskih poslikav bi bil postopek ročnega sestavljanja zelo dolgotrajen in naporen. S fragmenti je potrebno rokovati zelo previdno, kar še dodatno upočasni postopek. Med sestavljanjem nimamo naenkrat vpogleda v vse preostale fragmente, saj jih je preveč, da bi imeli vse sočasno na dosegu. Vsak naslednji fragment, ki bi ga radi dodali že sestavljenim, je potrebno poiskati med množico preostalih fragmentov.

Za pohitritev procesa sestavljanja poslikav sta bili v preteklosti že razviti dve aplikaciji, ki omogočata sestavljanje slik poslikav v digitalni obliki. Prva se imenuje Pedius [5, 8] in je namenjena restavratorjem, druga pa se imenuje e-Pedius [2, 4] in je namenjena širši množici uporabnikov. Razvoj aplikacije e-Pedius je potekal na Institutu Jožef Stefan v sodelovanju z Zavodom za varstvo kulturne dediščine Slovenije. V okviru razvoja aplikacije e-Pedius je bilo izdelano tudi to diplomsko delo.

Tudi v primeru digitalnega sestavljanja slik se pojavi težava pri izvedbi

iskanja dodatnih fragmentov, ki bi spadali poleg trenutno sestavljenih. Fragmentov je ponavadi veliko, zato uporabnik ne more pregledati vseh preostalih fragmentov, ki bi lahko še spadali na sliko. Računalnik mu mora ponuditi tiste fragmente, ki jih po nekem algoritmu smatra za ustrezne. Običajno so to fragmenti, ki so podobni fragmentom, ki ležijo na robu trenutne postavitve. Kateri algoritem je najprimernejši za iskanje podobnih fragmentov, smo skušali ugotoviti v tem diplomskem delu. Fragmenti imajo določene lastnosti, ki jih vidimo v redkokaterih drugih slikah, zato je smiselno primerjati različne algoritme in izbrati najprimernejšega za to nalogo. Pri implementaciji aplikacije e-Pedius se srečujemo tudi s tehničnimi zahtevami za učinkovito delovanje. V ta namen morajo biti fragmenti predhodno združeni v skupine, najprimerneje je, če se podobni fragmenti nahajajo v isti skupini.

V tem diplomskem delu skušamo rešiti dve nalogi. Prva je izračun podobnosti med fragmenti. Tu je potrebno izbrati najustreznejši način merjenja razdalj med fragmenti. Z dobljenimi razdaljami lahko nato poiščemo najbolj podobne fragmente. Druga naloga je razvrstitev fragmentov v skupine glede na izračunane podobnosti. V posameznih skupinah želimo imeti medsebojno podobne fragmente, tako da so skupine čim bolj enotne. Za obe nalogi je znanih več algoritmov, zato jih želimo primerjati med seboj in izbrati tistega, ki bo dosegal najboljše rezultate.

Delo je v nadaljevanju strukturirano na naslednji način. V drugem poglavju predstavimo fragmente stenskih poslikav, njihov izvor, njihove najočitnejše lastnosti in razdelitev fragmentov na dve množici glede na način poslikave. Predstavljena sta projekt Pedius, s katerim se je začelo delo s fragmenti v digitalni obliki, in projekt e-Pedius, ki ponuja sestavljanje širši množici in je bil povod za to diplomsko delo. Opisano je ozadje razvoja aplikacije e-Pedius in zahteve zanjo, ki izvirajo tako iz samega problema sestavljanja fragmentov kot tudi iz tehničnih zmožnosti naprav, na katerih bo aplikacija delovala. Predstavljena so orodja, ki so bila uporabljena pri testiranju predstavljenih algoritmov, in grafični vmesnik, v katerem si lahko ogledamo, kateri fragmenti so bili izbrani kot podobni in kako so bili združeni v

skupine. V tretjem poglavju se osredotočimo na digitalni zapis slik, njihovo obdelavo in algoritme za ugotavljanje podobnosti med slikami. V četrtem poglavju so opisane metode za združevanje v skupine ter njihova uporaba v našem primeru. V petem poglavju dokumentiramo eksperimentalno ovrednotenje izbranih metod za iskanje podobnih fragmentov in njihovo združevanje v skupine. Opisani so postopki testiranja, ki smo jih izvajali za primerjavo različnih algoritmov, ter dobljeni rezultati. Sledijo še zaključne ugotovitve in opis nekaj možnosti za nadaljnje delo.

Poglavje 2

Problematika in orodja

2.1 Fragmenti stenskih poslikav

Stenske poslikave so nekdaj krasile stene zgradb. Sčasoma pa je omet odpadel s sten in se razdrobil na fragmente. Vzroki za to so lahko dotrajanje ometa, naravne nesreče ali pa so ljudje pri obnovah star omet zavrgli v odpadne jame. Kosi ometa so običajno debeli nekaj centimetrov, so raznovrstnih oblik in velikosti, zgornja ravna površina je poslikana, robovi so lahko odkrušeni in tako ob robu ponekod ni barve, temveč je viden siv omet. Nekaj primerov fragmentov stanskih poslikav je prikazanih na sliki 2.1.

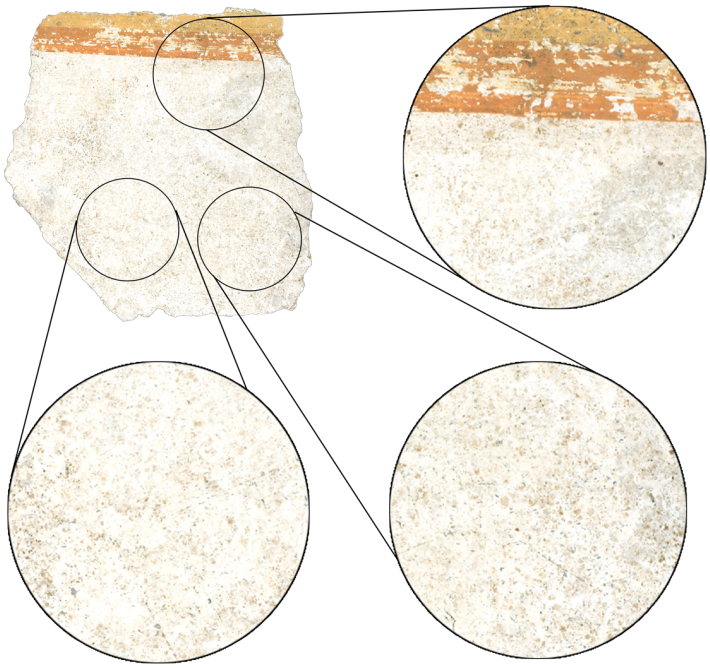
Ob arheoloških izkopavanjih fragmente ponavadi najdejo neurejeno ležati na tleh. Restavratorji fragmente očistijo in označijo z enoličnimi številkami za vodenje evidence in identifikacijo. Nato jih shranijo v zaboje in na varno mesto, kjer so zaščiteni pred pretiranim nadaljnjim razpadanjem. Nadalje fragmente sestavijo nazaj v prvotno razporeditev, kakršno so imeli, preden so odpadli s sten. S tem dobijo vpogled v motive, ki so jih prikazovale slike, ter tako prispevajo k razumevanju časa, v katerem so nastale. Sestavljanje poteka v peskovniku, ki je pripravljen za tovrstno delo. Pesek omogoča poravnavo zgornje poslikane površine fragmentov kljub različno debelemu ometu in neravni spodnji površini. V kolikor je fragmentov veliko, je tako delo dolgotrajno, saj je potrebno s fragmenti zaradi starosti in zgodovinske

Slika 2.1: Fragmenti stenskih poslikav

vrednosti ravnati zelo previdno. Tudi iskanje fragmentov, ki bi spadali poleg do takrat sestavljenih, je težavno, sej so fragmenti shranjeni v zabojih in restavratorji nimajo pregleda nad vsemi preostalimi fragmenti naenkrat.

Med izkopavanjem na območju današnjega Celja so našli več kot 9000 fragmentov stenskih poslikav, ki izhajajo iz rimskega obdobja, iz takratne Celeie. Najdišče in pripadajoči restavratorski projekt so poimenovali Turška mačka. Zaradi velikega števila fragmentov najdenih v eni zgradbi ni znano, ali so bili vsi del stenskih poslikav v tej zgradbi ali pa je bila zgradba namenjena za odlaganje odpadnih kosov ometa in so jih tja odlagali med obnavljanjem drugih zgradb. Izkopavanja so potekala leta 1978. Od takrat so fragmente nekajkrat poskusili sestaviti v prvotno postavitev, a zaradi obsežnosti projekta sestavljanja niso nikoli dokončali.

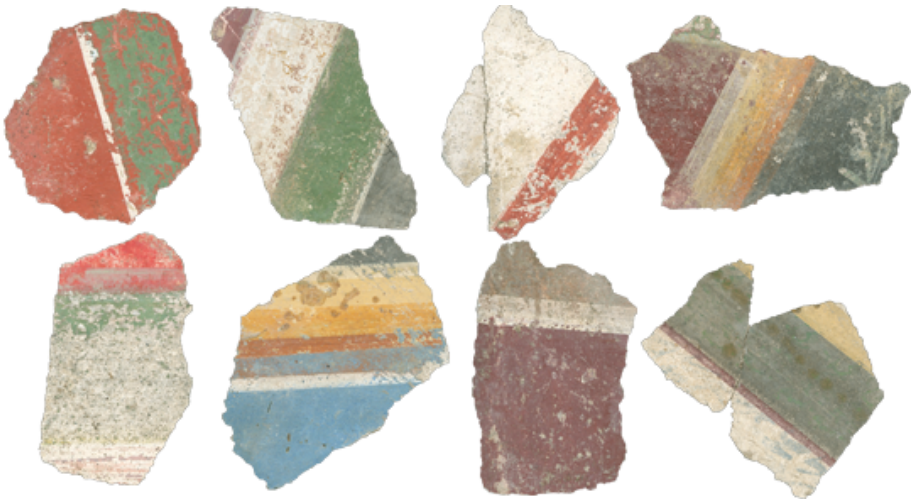
Na poslikavah fragmentov so dobro vidni zamišljeni vzorci in barve. Ob podrobnejšem ogledu je opaziti veliko manjših nepravilnosti, ki so posledica starosti fragmentov. Posamezna ploskev barve je lahko od blizu videti kot množica različno obarvanih pik, ki skupaj tvorijo barvo, kakršno zaznamo



Slika 2.2: Fragment s povečanimi območji

od daleč kot eno barvo. Na nekaterih delih je barva obrabljena in je viden siv omet, ponekod se pojavljajo razpoke in pike ometa. Na sliki 2.2 vidimo primer fragmenta s povečanimi nekaterimi območji.

Fragmente lahko glede na tip poslikave razdelimo v dve skupini, na tako imenovane vzorčaste in na enobarvne fragmente. Vzorčasti imajo narisan del vzorca, del črte ali pa so pobarvani z več barvami. Takšne fragmente lahko na podlagi podobnih grafičnih vzorcev ročno razvrstimo v skupine, fragmenti znotraj ene skupine pa bodo skupaj tvorili vzorec na končni sliki. Druga skupina fragmentov so enobarvni fragmenti, ki ne vsebujejo izrazitih vzorcev, so večinoma ene barve oziroma so bili ene barve ob poslikavi, sedaj pa zaradi vpliva časa vsebujejo tudi nekaj drugih odtenkov. Ročno jih ni bilo mogoče dodati k skupinam vzorčastih fragmentov, saj je težko določiti, v katero skupino spadajo. Na sliki 2.3 je prikazanih nekaj primerov vzorčastih fragmentov.



Slika 2.3: Primeri vzorčastih fragmentov

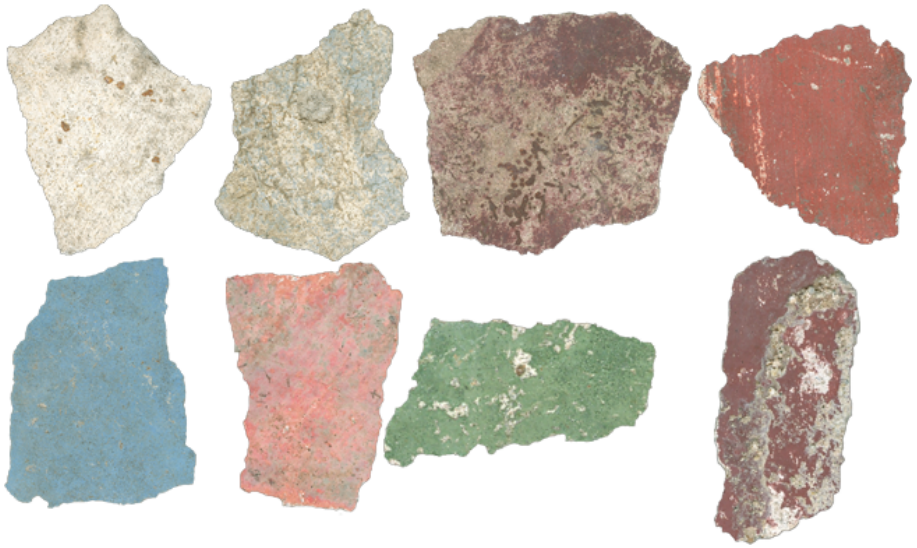
2.2 Enobarvni fragmenti

Enobarvni fragmenti ne vsebujejo le ene barve, pač pa, gledano od blizu, več odtenkov barve posameznih točk. Skozi čas so se na njih pojavile nepravilnosti, kot so razpoke, odrgnine, zbledela barva, sivi in beli predeli, kjer je barva odpadla in podobno. Kot enobarvni fragmenti so označeni predvsem zato, ker predvidevamo, da so bili v času poslikave zamišljeni kot enobarvni, danes pa niso več le ene barve. Na sliki 2.4 je prikazanih nekaj primerov enobarvnih fragmentov.

2.3 Projekt Pedius

V okviru projekta Pedius je bila v sodelovanju Zavoda za varstvo kulturne dediščine Slovenije in Instituta "Jožef Stefan" opravljena digitalizacija slik fragmentov in razvita aplikacija za njihovo sestavljanje na računalniku, namenjena restavratorjem [5].

Digitalizacijo fragmentov so opravili z optičnimi čitalniki, večje fragmente, ki so presegali površino optičnega čitalnika, pa so slikali z digitalnim foto-



Slika 2.4: Primeri enobarvnih fragmentov

aparatom. V tem primeru je na vsaki sliki zajeto tudi merilo, ki omogoča pretvorbo slik na enako povečavo, kot jo imajo slike, zajete z optičnim čitalnikom.

Nato so slike fragmentov digitalno obdelali s pomočjo za to pripravljene programske opreme. Ta avtomatsko odstrani ozadje slike, tako da je področje slike, kjer se ne nahaja fragment, prozorno. S programskim orodjem lahko restavratorji odstranijo dele slike, ki predstavljajo kose ometa ob robovih fragmentov, ki so nastali med lomljenjem fragmentov.

Aplikacija Pedius omogoča digitalno sestavljanje fragmentov na osebnih računalnikih. Restavratorji lahko fragmente premikajo in obračajo, fragmente, za katere so prepričani, da sodijo skupaj, lahko združijo v skupino in od tedaj naprej delajo z njimi kot z enim fragmentom. Tako si olajšajo zahtevnost sestavljanja in razdelijo delo na več manjših opravkov. Spreminjajo lahko oddaljenost pogleda, približajo in oddaljijo postavitev glede na količino podrobnosti, ki jih želijo videti. Preizkusijo lahko več različnih postavitev, v kolikor niso povsem prepričani, katera je pravilna, saj je premikanje fragmentov hitro in ni možnosti poškodb fragmentov. Sestavljene delne postavitve

lahko shranijo in kasneje nadaljujejo s sestavljanjem.

2.4 Projekt e-Pedius

2.4.1 Zasnova aplikacije

Projekt e-Pedius je nadaljevanje projekta Pedius. Cilj je izdelava aplikacije za sodobne digitalne naprave, predvsem tablične računalnike, ki bo približala sestavljanje fragmentov stenskih poslikav širši publiki [4, 2].

V aplikaciji bodo številni uporabniki na zabaven način, preko igre, lahko opravili sicer zelo obsežno in zamudno opravilo sestavljanja fragmentov, ter tako pomagali restavratorjem hitreje in učinkoviteje opraviti delo. Takšnemu pristopu pravimo množično izvajanje (angl. crowdsourcing). Tako lahko s pravim pristopom združimo zabavnost igre za uporabnika in koristnost opravljenega dela za avtorje aplikacije. Obenem dobijo uporabniki vpogled v sam proces dela strokovnjakov in izvedo veliko novega o določenem področju. V našem primeru sodelujejo pri procesu sestavljanja fragmentov, izvedo veliko o restavratorskem poklicu in zgodovini ter so vključeni v proces obnavljanja in ohranjanja kulturne dediščine. Restavratorji spremljajo napredovanje sestavljanja, ocenjujejo postavitve, ki so jih sestavili uporabniki, in med najboljšimi izberejo tiste, ki jih nato uporabijo v praksi, torej sestavijo prave fragmente v peskovniku.

Aplikacija e-Pedius deluje na tabličnih računalnikih z operacijskimi sistemi Android, iOS in Windows ter na računalnikih v spletnih brskalnikih, ki podpirajo standard HTML5. Namenjena je širši publiki in je brezplačno dostopna na spletnih tržnicah z aplikacijami vseh treh ponudnikov tabličnih operacijskih sistemov. Za dostop preko spletnega brskalnika ni potrebno namestiti aplikacije na lokalni disk, do nje dostopamo preko spleta na povezavi, ki jo najdemo na spletni strani projekta [4].

Aplikacija je zasnovana kot igra, v kateri uporabnik sestavlja stenske poslikave v postavitve, kakršne se mu zdijo smiselne. Med sestavljanjem lahko shrani trenutno postavitev in kasneje nadaljuje eno od njih. Nadaljuje lahko

tudi postavitve ostalih uporabnikov, tako da ni potrebno vsakemu uporabniku začeti s sestavljanjem od začetka, ampak lahko trud vloži v nadgradnjo dela nekoga drugega.

Ko uporabnik smatra postavitev za končano, jo lahko pošlje v ocenjevanje. Postavitve ocenjujejo uporabniki drug drugemu. Ob vstopu v del aplikacije za ocenjevanje postavitev se slika naključne postavitve prikaže na zaslonu. Uporabnik oceni postavitev z ustreznim številom zvezdic (od ena do pet) in prikaže se naslednja slika za ocenjevanje.

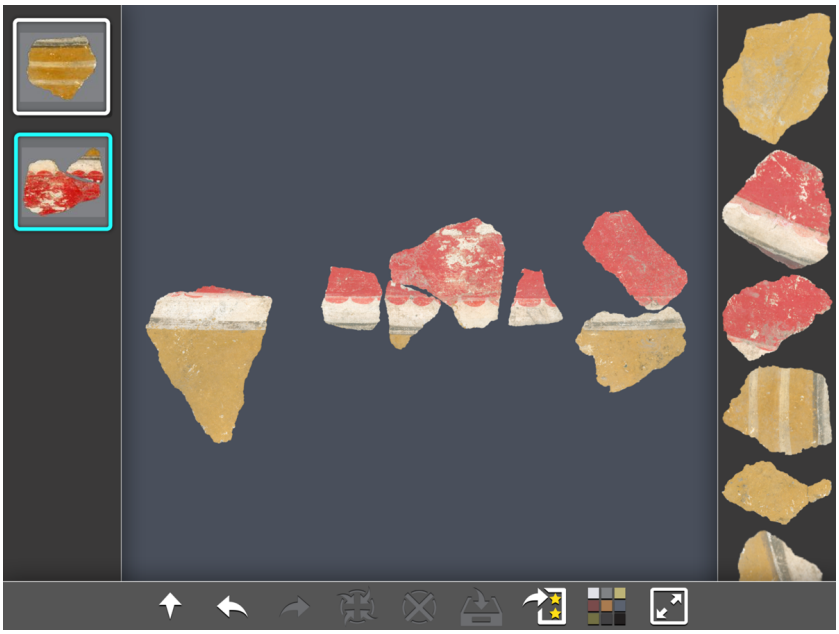
Uporabniki za opravljeno delo prejmejo točke in značke za posebne dosežke. Značke lahko prejmejo tako za sestavljanje postavitev kot tudi za ocenjevanje postavitev ostalih uporabnikov. Za postavitev, ki jo je uporabnik poslal v ocenjevanje, se izračuna povprečna ocena vseh ocen, ki jih je ta postavitev dobila. Število točk, ki jih prejme uporabnik za postavitev, je enako zmnožku povprečne ocene postavitve in števila fragmentov, ki so bili uporabljeni za postavitev. Za večje postavitve je uporabnik sorazmerno bolje nagrajen. V kolikor uporabnik nadaljuje postavitev drugega uporabnika, prejme število točk glede na število fragmentov, ki jih je dodal postavitvi. Uporabniki so s številom vseh zbranih točk uvrščeni na lestvico, kjer se lahko primerjajo z ostalimi, kar služi kot motivacija za nadaljnje delo.

Trenutno je najobsežnejši projekt, ki je na voljo v aplikaciji, projekt Turška mačka z več kot 9000 fragmenti. Poleg njega aplikacija ponuja še projekt Štirje letni časi, ki vsebuje štiri naloge. Namenjene so kot uvod v delo z aplikacijo. Prva naloga je najlažja in je namenjena spoznavanju sestavljanja fragmentov, ostale tri si sledijo po naraščajoči težavnosti in pripravijo uporabnika na zahtevnejši projekt Turška mačka. Te naloge so restavratorji že sestavili, tako da poznamo pravilne postavitve in lahko uporabniki nemudoma dobijo povratno informacijo o tem, kakšno oceno in koliko točk so dosegli. Restavratorji pripravljajo še tretji projekt, ki bo kmalu na voljo. V splošnem pa aplikacija ni vezana na konkretne projekte, kasneje bo mogoče dodati poljubno število novih projektov.

Pri projektu Turška mačka je zbirka fragmentov razdeljena na več na-

log. Restavratorji so vzorčaste fragmente ročno razdelili v skupine glede na podobne barve in vzorce. Z vzorčastimi fragmenti je smiselno začeti sestavljanje nove postavitve, saj zaradi vzorcev in črt, ki jih vsebujejo, lažje ugotovimo, kam spadajo. Vsaka skupina vsebuje takšne vzorčaste fragmente, ki bodo najverjetneje tvorili skupno sliko. S tem se celoten projekt deli na več lažje obvladljivih manjših nalog. Pred pričetkom sestavljanja uporabnik izbere eno od skupin vzorčastih fragmentov, s katero bi želel začeti sestavljanje. Vzorčasti fragmenti zadoščajo za postavitev osrednjega vzorca slike. Za dokončanje enobarvnih delov slike, na primer ozadja, mora biti na voljo učinkovit način iskanja enobarvnih fragmentov. Enobarvni fragmenti niso bili združeni v skupine skupaj z vzorčastimi, saj je težko določiti, v katero skupino spadajo. Do enobarvnih fragmentov se zato dostopa iz aplikacije med sestavljanjem, s pritiskom na za to namenjen gumb in izbiro območja, kateremu podobne enobarvne fragmente želimo.

Na zaslonu za sestavljanje večji del zaslona predstavlja površina, na kateri sestavljamo fragmente. Uporabnik lahko fragmente premika, jih zasuka, več fragmentov združi v skupino, oddalji ali približa pogled. Ob strani so v stolpcu prikazani vzorčasti fragmenti, ki jih uporabnik še ni uporabil. Z dotikom enega od teh fragmentov se ta fragment prenese na osrednji del zaslona, kjer poteka sestavljanje. Na spodnjem delu zaslon se nahaja orodna vrstica, v kateri lahko uporabnik spreminja barvo ozadja, razveljavi ali obnovi zadnje spremembe, shrani postavitev ali jo pošlje v ocenjevanje. Na voljo je tudi orodje, s katerim lahko poišče in izbere enobarvne fragmente. Ko izbere to orodje, določi položaj in polmer območja, kateremu bi rad našel podobne enobarvne fragmente. Vse točke znotraj tega območja, na katerih se nahajajo fragmenti, se upoštevajo kot točke, katerim podobne enobarvne fragmente bo algoritem iskal. Slika 2.5 prikazuje zaslonski posnetek aplikacije e-Pedius med sestavljanjem. Na njej vidimo eno od nalog iz projekta Turška mačka.



Slika 2.5: Posnetek zaslona za sestavljanje postavitev v aplikaciji e-Pedius

2.4.2 Tehnične podrobnosti aplikacije

Aplikacija e-Pedius za tablične računalnike je v večji meri implementirana v pripadajočih domorodnih programskih jezikih, za operacijski sistem Android v jeziku Java, za iOS v jeziku Objective C in za Windows v jeziku C#. Del aplikacije, ki omogoča sestavljanje fragmentov, pa je za zagotavljanje hitrega in odzivnega delovanja izdelan s knjižnico Cocos2d-x. To je programska knjižnica za razvoj iger, ki uporabljajo dvodimenzionalno grafiko. Napisana je v jeziku C++, ki je zelo razširjen programski jezik, tako da deluje na večini modernih operacijskih sistemov.

Za izris na zaslon uporablja grafični standard OpenGL. Ta deluje na večini operacijskih sistemov in za izris izkoristi grafično kartico naprave, kar pripomore k hitrejšemu delovanju. OpenGL izriše celotno vsebino zaslona večkrat na sekundo. Večkrat kot jo izriše, bolj tekoče je delovanje. Pri tem pa je omejitev čas, saj se lahko zgodi, da manj zmogljiva strojna oprema ali veliko število elementov na zaslonu za izris zahtevata več časa, posledično pa

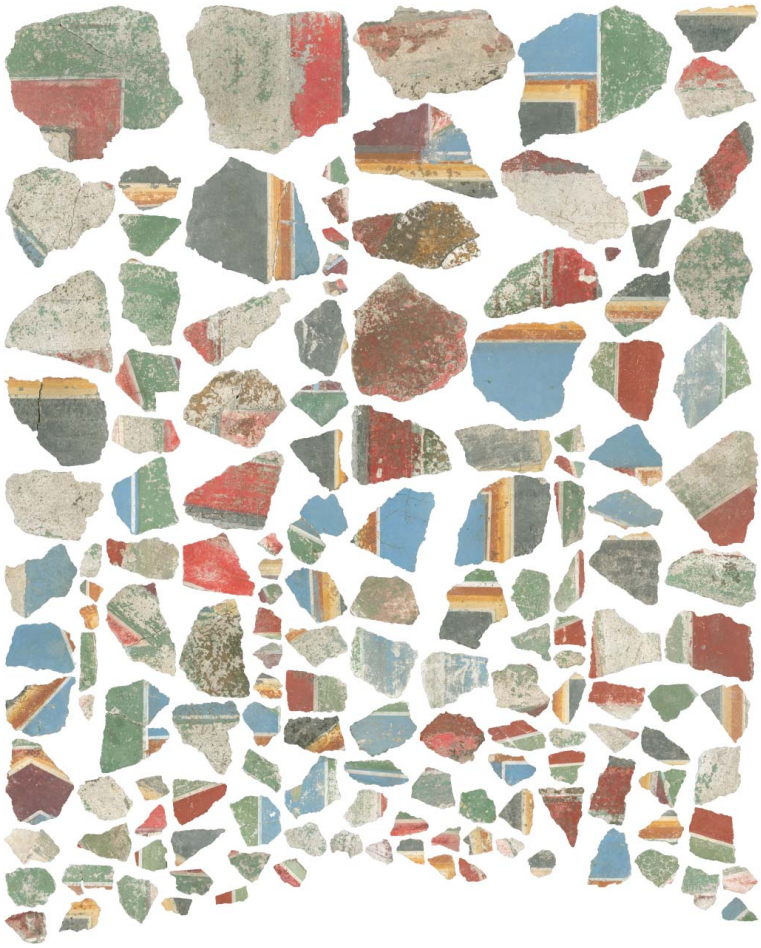
se število izrisov na sekundo zmanjša. Običajno je pri igrah zaželeno 60 izrisov na sekundo. Pri naši aplikaciji je ob daljšem sestavljanju lahko na zaslonu prikazanih zelo veliko slik fragmentov. Da je delovanje hitrejše, ne nalagamo vsake slike posebej, ampak slike že pri izdelavi aplikacije združimo v večjo sliko, imenovano zlepek sličic (angl. spritesheet). Zraven dodamo še datoteko, v kateri so zapisane koordinate in velikosti prvotnih slik. Med izrisovanjem OpenGL namesto obravnavanja vsake slike posebej le prestavlja področje zlepka sličic, ki ga nato izriše na zaslon. Ta lastnost grafičnega pogona OpenGL vpliva na samo strukturo aplikacije, saj ob iskanju podobnih enobarvnih fragmentov ni dovolj, da poiščemo podobne fragmente. Fragmente moramo predhodno združiti v skupine, čim bolj podobne skupaj, ter za vsako skupino pripraviti zlepek sličic. Ob uporabnikovem iskanju podobnih enobarvnih fragmentov najprej poiščemo skupino, ki je najbolj podobna označenemu področju na zaslonu, nato pa znotraj te skupine razvrstimo fragmente po podobnosti, tako da so najbolj podobni fragmenti postavljeni pri vrhu in najprej vidni.

Slike fragmentov že enega samega projekta zasedejo veliko prostora na disku. Aplikacija podpira sestavljanje fragmentov več projektov, kasneje se lahko dodajajo tudi novi projekti. Zato se skupine slik fragmentov oziroma pripadajoči zlepki sličic ne prenesejo skupaj z aplikacijo ob namestitvi, temveč se aplikacija med delovanjem poveže s spletnim strežnikom in prenese zlepke sličic, ki so v danem trenutku potrebni. Takšen način tudi omogoča, da se kasneje doda nov projekt, ne da bi bilo uporabniku potrebno posodobiti aplikacijo na svoji napravi. Primer zlepka sličic ene od skupin vzorčastih fragmentov se nahaja na sliki 2.6.

2.5 Uporabljena orodja

2.5.1 OpenCV

Za delo s slikami smo uporabljali odprtokodno knjižnico OpenCV, s katero smo si pomagali pri izgradnji histograma slike, pri obrezovanju slik fragmen-



Slika 2.6: Primer zlepka sličic skupine vzorčastih fragmentov

tov, pri rotiranju slik fragmentov in pri izdelavi pomanjšanih slik, ki smo jih nato uporabili v grafičnem vmesniku.

OpenCV velja za eno najbolj popularnih knjižnic za analizo slik in uporabo algoritmov računalniškega vida. Implementirana je v jeziku C++, kar omogoča hitro delovanje, na voljo pa sta tudi različici v jezikih Python in Java. Tudi pri uporabi slednjih jezikov se v ozadju izvaja koda C++, medtem ko imajo razvijalci na voljo vmesnik v obeh jezikih.

V knjižnici je na voljo veliko različnih funkcionalnosti, za nas pa so najbolj uporabne naslednje:

- funkcija za nalaganje slik iz datoteke v pomnilnik, ki omogoča tudi uporabo slik v zapisu PNG s prozornostjo, kar je pri nas zelo pomembno, saj prozoren del slike ne prikazuje fragmenta in ga ne upoštevamo pri izračunih barv;
- funkcija za shranjevanje slik, ki omogoča zapisovanje slik na trdi disk;
- funkcija za pretvarjanje slike med barvnimi modeli;
- funkcija za razdelitev slike na posamezne barvne kanale, po uporabi katere lahko vsak kanal posebej obdelamo, barvne kanale pa na koncu spet združimo v eno sliko;
- funkcija za izdelavo histogramov, ki izračuna histogram z želenim številom polj, podamo ji lahko tudi območje slike, s katerega naj izračuna histogram, kar nam pride prav, saj želimo histogram izračunati le iz dela, ki ni prozoren;
- funkcija za izris, ki omogoča, da dobljene histograme izrišemo in shranimo kot sliko;
- funkcija za združevanje več slik v skupno sliko, s katero lahko slike postavimo eno poleg druge vodoravno ali eno nad drugo navpično ter jih v takšni obliki shranimo v novo sliko;

- spreminjanje velikosti slike, pri čemer podamo želeno velikost slike in slika se sorazmerno poveča/pomanjša;

- obrezovanje slik, kjer podamo koordinate pravokotnega področja, ki ga želimo izrezati kot novo sliko. V našem primeru je to uporabno za obrezovanje prozornih delov slike. Vse slike fragmentov, ki so bile zajete v okviru projekta Pedius, imajo enako ločljivost, ne glede na dejansko velikost fragmenta. Zato imajo lahko poleg fragmenta še različno veliko prozorno območje. Odvečne prozorne točke slike lahko odstranimo in tako zmanjšamo velikost slike, ki gre v nadaljnjo obdelavo. To uporabimo tudi pri izdelavi manjših in obrezanih slik za grafični vmesnik, kjer si želimo čim bolje zapolniti prostor, ki je na voljo na zaslonu.

2.5.2 Weka

Weka je odprtokodna knjižnica, v kateri so implementirani mnogi algoritmi strojnega učenja [11]. Napisana je v programskem jeziku Java. Razvita je bila na univerzi Waikato na Novi Zelandiji. Uporablja se lahko kot grafični vmesnik, ki ne zahteva znanja programiranja, ali pa kot knjižnico, ki jo vključimo v program. V okviru diplomskega dela uporabljamo njene metode za združevanje podobnih elementov v skupine.

2.5.3 Orange

Orange je odprtokodna knjižnica za strojno učenje, ki je bila razvita na Fakulteti za računalništvo in informatiko v Ljubljani. Del nje je napisan v jeziku Python, računsko zahtevnejši deli pa so napisani v jeziku C++ [3]. Uporabljamo jo lahko kot programsko knjižnico, do katere dostopamo v Pythonu, ali pa kot grafični vmesnik, ki ne zahteva programiranja. V nekaterih implementacijah algoritmov se razlikuje od Weke, tako imamo možnost primerjave delovanja obeh in za vsak algoritem izberemo tisto orodje, ki se v danem primeru bolje obnese.

2.6 Grafični vmesnik

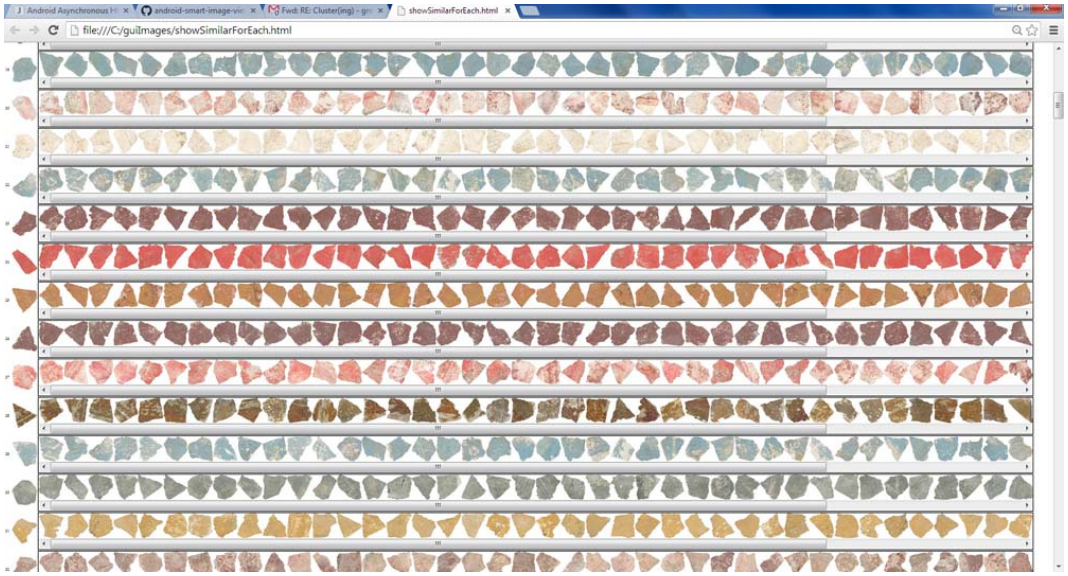
Želimo imeti pregled nad tem, kateri fragmenti so bili izbrani za podobne in kateri so bili združeni v skupine, saj lahko že s hitrim pregledom opazimo morebitna bistvena odstopanja.

Izdelali smo grafični vmesnik, v katerem ima vsak enobarven fragment svojo vrstico z njemu podobnimi fragmenti (način iskanja podobnosti le-teh opisujemo v tretjem poglavju). Izbran fragment je prikazan prvi v vrstici, ostali mu sledijo po padajoči podobnosti. Na zaslonu je prikazanih le prvih nekaj podobnih fragmentov, za ogled ostalih pa lahko celotno vrstico pomikamo v levo. Vrstice so razporejene ena nad drugo, tako da se lahko med njimi premikamo s koleščkom na miški in hitro opazimo, če so kje bistvena odstopanja.

Za ogled v skupine združenih fragmentov so slike ene skupine razporejene v več vrstic, v kolikor v eni ni dovolj prostora, tako da pomikanje v levo ni potrebno. V tem primeru so namreč vsi elementi skupine enako pomembni.

V restavratorskem projektu Turška mačka imamo 5172 enobarvnih fragmentov in če za vsakega prikažemo 50 najbolj podobnih fragmentov, je to skupaj več kot 250.000 slik. Slike fragmentov digitaliziranih v okviru projekta Pedius so ločljivosti 1716 $\times$ 2464 in zasedajo od približno 200 kB do 2 MB prostora, odvisno od velikosti fragmenta. Za zmanjšanje porabe pomnilnika smo najprej izdelali manjše verzije teh slik velikosti 150 $\times$ 150, kjer se fragment razteza po celotni sliki, s čim manj praznega prostora. Poleg tega namesto zapisa slik PNG uporabimo zapis JPG, ki omogoča manjšo porabo prostora v pomnilniku. Sicer zapis JPG ne podpira prozornega ozadja, katerega pa v tem primeru ne potrebujemo, saj si slike le ogledujemo. Vseh 5172 slik velikost 150 $\times$ 150 točk v zapisu JPG zaseda okrog 50 MB prostora. Tako lahko v grafičnem vmesniku prikažemo vse slike obenem, ne da bi nam zmanjkalo prostora v delovnem pomnilniku.

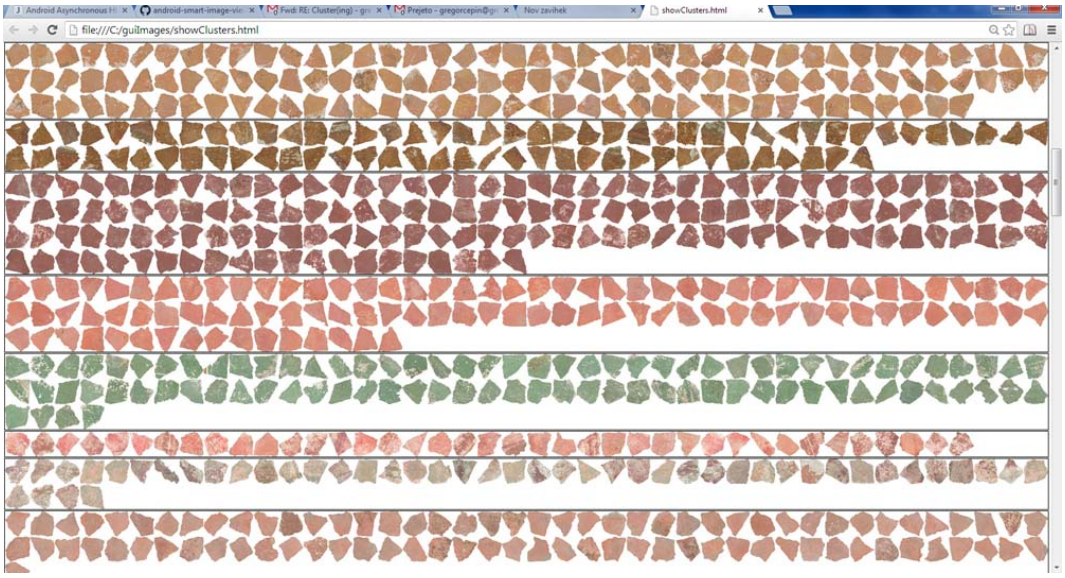
Vmesnik je napisan v jezikih HTML in Javascript. Pri prikazu podobnih fragmentov lahko prikažemo 500 vrstic s po 50 slikami, ob nekoliko počasnejšem delovanju pa lahko prikažemo tudi vseh 5172 vrstic. Pri pri-



Slika 2.7: Primer zaslona grafičnega vmesnika za ogled podobnih enobarvnih fragmentov

kazu skupin fragmentov ni podobnih težav in lahko vse skupine prikažemo obenem, saj se slika vsakega fragmenta pojavi le enkrat.

Slika 2.7 prikazuje zaslonski posnetek prikaza podobnih enobarvnih fragmentov. V povsem levem stolpcu so prikazani fragmenti, katerim iščemo podobne fragmente. Podobni fragmenti si sledijo po padajoči podobnosti v desno smer. Slika 2.8 prikazuje zaslonski posnetek prikaza skupin enobarvnih fragmentov.



Slika 2.8: Primer zaslona grafičnega vmesnika za ogled skupin enobarvnih fragmentov

Poglavje 3

Iskanje podobnih fragmentov

3.1 Barvni modeli

Barvni model predstavlja način zapisa barv slike v digitalni obliki. Za vsako barvo, ki jo želimo shraniti ali prikazati, moramo poznati številsko vrednost, ki predstavlja to barvo v digitalnem zapisu. S to vrednostjo lahko shranimo določeno barvo, ter jo kasneje uporabimo pri prikazu ali nadaljnji obdelavi slike. Pri tem običajno uporabimo nek obstoječ dogovor o načinu zapisa barv, imenovan barvni model. Barvna vrednost je lahko razdeljena na več številskih komponent, pri čemer souporaba vseh komponent predstavi končno barvo. Posamezno komponento imenujemo barvni kanal. Večina barvnih modelov ima tri barvne kanale, ne pa vsi. Na primer barvni model CMYK, ki se uporablja pri barvnem tiskanju, ima štiri barvne kanale, kadar zapišemo sliko v črno beli tehniki, pa zadostuje en barvni kanal. V tem diplomskem delu smo uporabljali tri barvne modele in sicer modele RGB, HSV in LAB [10]. Izbira barvnega modela je pomembna, saj vse izračune, ki jih opravimo na barvah slike, dejansko opravimo na številskih vrednostih, s katerimi so te barve zapisane. Različni barvni modeli ponujajo različne načine zapisa na videz enake barve. Različni načini zapisa barve se različno obnesejo pri različnih problemih.

Barvni model RGB (angl. Red, Green, Blue) predstavi barvo s tremi

barvnimi kanali, rdečim, zelenim in modrim. Model predpostavlja, da je mogoče vse barve, ki jih potrebujemo, ustvariti s kombinacijo teh treh barv. Vsak barvni kanal poda informacijo o tem, koliko te barve je v končni barvi. Je eden najbolj razširjenih barvnih modelov, uporablja se na primer za prikaz slike na zaslon, saj imajo običajno zasloni tri različne vire svetlobe, za vsak barvni kanal modela RGB enega.

Barvni model HSV (angl. Hue, Saturation, Value) je barvni model, v katerem je vsaka točka predstavljena s tremi vrednostmi: barvni odtenek (H), intenzivnost barve (S), ki pove, kje se nahaja izbrana barva med povsem belo in popolnoma čisto barvo, in svetlost (V), ki pove, kje med povsem črno in povsem čisto barvo se nahaja izbrana barva.

V barvnem modelu LAB vrednost L predstavlja svetlost (angl. lightness), vrednosti A in B (oznaki osi barvnih dimenzij) pa položaj na barvni ravnini, na kateri se nahajajo različni barvni odtenki. Model je bil ustvarjen z namenom, da bi posnemal človeško dojemanje barv. Tako imata dve barvi, ki se ljudem zdita podobni, majhno razliko vrednosti v modelu LAB.

Vsak barvni kanal ima v eni točki slike 256 možnih vrednosti. Različnih barv, ki jih lahko s posameznim barvnim modelom prikažemo je 256^3, saj imamo tri barvne kanale in so možne vse njihove kombinacije, kar pomeni, da lahko predstavimo 16,77 milijonov barv. Nekatere slike, kar velja tudi za naše slike fragmentov, pa imajo poleg informacije o barvi še podatek o prozornosti (angl. alpha). Običajno je možnih 256 vrednosti prozornosti, od 0, ki pomeni popolnoma prozorno, do 255, ki pomeni povsem neprozorno. V našem primeru vrednost prozornosti pove, ali se v neki točki slike nahaja fragment ali pa je v tisti točki prozorno ozadje. Če je vrednost prozornosti 0, je točka prozorna in v njej ne vidimo nobene barve, ne glede na vrednosti barvnih kanalov. V taki točki ni fragmenta in vrednosti barvnih kanalov te točke ne upoštevamo. Če je vrednost prozornosti 255, je točka neprozorna in v njej vidimo barvo, kakršna je določena z vrednostmi barvnih kanalov. Točke na robu fragmenta imajo neko vmesno vrednost prozornosti, vendar pa so te točke običajno sivkaste barve in jih ne bomo upoštevali, saj nam le

zmanjšajo natančnost točk, za katere je jasno določena barva.

3.2 Histogrami slik

Za slike fragmentov potrebujemo opis oziroma množico vrednosti, po katerih bomo primerjali fragmente med seboj. Te vrednosti morajo odražati lastnosti fragmenta. Pri enobarvnih fragmentih je najpomembnejša lastnost, po kateri se ti med seboj razlikujejo, barva fragmenta. Robovi fragmentov so preveč načeti, da bi si lahko pomagali z njihovo obliko, tudi ni pravih vzorcev na slikah fragmentov, s pomočjo katerih bi zaznali podobnost med enobarvnimi fragmenti s katero od drugih metod računalniškega vida.

Barvo fragmenta lahko predstavimo s histogramom. Histogram je oblika opisa slike, v kateri za vsako možno vrednost barvnega kanala preštejemo, kolikokrat se je ta pojavila na sliki. Najbolj podroben histogram ima 256 polj, kolikor je možnih vrednosti enega barvnega kanala. Z združevanjem sosednjih polj lahko dobimo tudi histograme z manjšim številom polj. Za izračun histograma slike se premikamo po vseh točkah slike in za vsako točko pogledamo, kolikšno vrednost ima posamezen barvni kanal. Za ena povečamo vrednost v polju histograma, ki šteje število ponovitev te vrednosti barvnega kanala. Za sliko zapisano z barvnim modelom s tremi barvnimi kanali, izračunamo tri histograme, za vsak barvni kanal po enega. Skupaj to znese $3 \times 256 = 768$ vrednosti, na podlagi katerih lahko nato izračunamo razdaljo do ostalih fragmentov.

Z izdelavo histograma sicer ne ohranimo vse informacije s slike, saj izgubimo informacijo o razporeditvi barv na sliki in o medsebojnih kombinacijah barvnih modelov. Tako imata sliki, ki sta sestavljeni iz točk enakih barv, a so te drugače razporejene, enaka histograma, prav tako imata lahko dve sliki z različnimi barvami točk enake histograme v primeru, da so barve sestavljene iz istih vrednosti barvnih kanalov, a so te drugače kombinirane med seboj.

Prednosti uporabe histograma so hiter izračun in hitra primerjava med histogrami, tako izračunane razdalje niso odvisne od rotacije slike, njene

velikosti ali razporeditve točk na sliki.

3.3 Normalizacija histogramov

Slike fragmentov so različnih velikosti, zato vsebujejo različno število neprozornih točk, iz katerih računamo histograme. Histogrami imajo zato različne vsote vseh polj. Pri primerjavi histogramov med seboj bi velikosti fragmentov na slikah vplivale na razdalje med njimi. Nas pa zanima predvsem podobnost po barvi fragmentov in ne njihovi velikosti. Z normalizacijo histogramov dosežemo, da histogrami predstavljajo delež posamezne vrednosti barvnega kanala na sliki in ne njenega absolutnega števila pojavitev. Za izračun normaliziranega histograma vrednost vsakega polja v histogramu delimo z vsoto vrednosti vseh polj v histogramu. Tako dobljen histogram ima vsoto vseh polj enako 1.

3.4 Mere razdalj med histogrami

Naš cilj je poiskati podobne fragmente, torej moramo poiskati podobne histograme, ki predstavljajo te fragmente. Ker je vsaka slika fragmenta opisana s tremi histogrami, jih najprej staknemo v en sam histogram. Nato podobne histograme iščemo tako, da z izbrano mero izmerimo razdaljo med dvema histogramoma, in tista histograma, ki imata najmanjšo medsebojno razdaljo, sta si najbolj podobna. Različne mere razdalje se za različne probleme različno dobro obnesejo, zato je pomembno, da jih preizkusimo na testnih primerih. Tako ugotovimo, katera mera najbolje deluje v našem primeru. Preizkusili smo nekaj najpogosteje uporabljenih mer razdalje, to so manhattanska razdalja, evklidska razdalja, razdalja χ^2, korelacija med histogramoma in Hellingerjeva razdalja [1, 6]. Formule za izračun omenjenih mer razdalj so zapisane v tabeli 3.1.

Tabela 3.1: Uporabljene mere razdalj med histogrami s pripadajočimi formulami

Mera razdalje	Formula za izračun
manhattanska razdalja	$d(a,b) = \sum_{i=0}^{n} \mid a_i - b_i \mid$
evklidska razdalja	$d(a,b) = \sqrt{\sum_{i=0}^{n}(a_i - b_i)^2}$
korelacija	$d(a,b) = \frac{\sum_{i=0}^{n}(a_i - a_{\text{avg}})(b_i - b_{\text{avg}})}{\sqrt{\sum_{i=0}^{n}(a_i - a_{\text{avg}})^2 \sum_{i=0}^{n}(b_i - b_{\text{avg}})^2}}$
χ^2	$d(a,b) = \sqrt{\sum_{i=0}^{n} \frac{(a_i - b_i)^2}{a_i}}$
Hellingerjeva razdalja	$d(a,b) = \sqrt{1 - \frac{1}{\sqrt{a_{\text{avg}} b_{\text{avg}} N^2}} \sum_{i=0}^{n} \sqrt{a_i \cdot b_i}}$

Poglavje 4

Združevanje v skupine

4.1 Motivacija za združevanje fragmentov

Za učinkovito delovanje aplikacije e-Pedius morajo biti slike fragmentov združene v zlepke sličic. Ob iskanju podobnih enobarvnih fragmentov se bo v aplikaciji naložil zlepek sličic, ki je najbolj podoben označenemu območju. Pomembno je, da so v zlepku sličic skupaj čim bolj podobni fragmenti. Prvi razlog za to je, da se bo histogram izbranega območja na zaslonu primerjal s povprečnimi histogrami skupin. V kolikor skupine ne vsebujejo podobnih fragmentov, bo povprečni histogram slabo predstavil vse fragmente v skupini. Drugi razlog je, da želimo, da je v prenesenem zlepku sličic čim več takšnih fragmentov, ki so podobni izbranemu območju.

Prednost združevanja fragmentov v skupine je, poleg hitrejšega izrisa na zaslon z uporabo zlepkov sličic, tudi v hitrejšem iskanju podobnih fragmentov [7]. Fragment, kateremu bi radi poiskali podobne fragmente, tako primerjamo le s centri skupin in fragmenti znotraj najbližje skupine, namesto z vsemi fragmenti. Trenutno imamo 5172 enobarvnih fragmentov. Če bi računali razdalje do vseh fragmentov, bi morali opraviti 5172 takšnih izračunov. Če so ti združeni v 50 skupin, je povprečna velikost skupine približno 103. V tem primeru najprej opravimo 50 izračunov razdalj do skupin, ter nato še v povprečju 103 izračune razdalj do fragmentov znotraj skupine, kar je skupaj

153 izračunov razdalj. To je bistveno manj ob 5172 izračunov razdalj in pripomore k hitrejšemu delovanju aplikacije. Lahko pa tak postopek najde suboptimalno rešitev, saj so predstavniki skupin povprečeni histogrami in je lahko prva izbira skupine napačna.

Najpogosteje uporabljena algoritma za združevanje v skupine sta hierarhično združevanje in metoda voditeljev. Ta dva algoritma smo preizkusili na našem problemu.

4.2 Hierarhično združevanje v skupine

Pred pričetkom hierarhičnega združevanja v skupine moramo imeti izračunane vse medsebojne razdalje med elementi, ki jih želimo združevati. Hierarhično združevanje v skupine deluje na naslednji način. V prvem koraku so vsi elementi postavljeni vsak v svojo skupino, torej je toliko skupin, kot je elementov. Nato v vsakem koraku združi dve, v tistem trenutku najbolj podobni skupini. Ta postopek ponavljamo, dokler nista v zadnjem koraku združeni zadnji dve skupini in dobimo eno veliko skupino, v kateri so zajeti vsi elementi [9]. Kot mero razdalje med elementi smo v našem primeru uporabili Hellingerjevo razdaljo, ki se je izkazala za najuspešnejšo pri primerjavi mer razdalje.

Možnih je več načinov računanja razdalj med skupinami. Te nam omogočajo izračun razdalje med dvema skupinama, ne glede na to, koliko elementov ti dve skupini vsebujeta. Načini računanja razdalj med skupinami so naslednji:

- najkrajša povezava (angl. single linkage): kot razdalja med skupinama je uporabljena najkrajša razdalja med dvema elementoma, ki se nahajata vsak v eni od skupin;
- najdaljša povezava (angl. complete linkage): kot razdalja med skupinama je uporabljena najdaljša razdalja med dvema elementoma, ki se nahajata vsak v eni od skupin;

- povprečna povezava (angl. average linkage): kot razdalja med skupinama je uporabljena povprečna razdalja med vsemi kombinacijami elementov v obeh skupinah;
- Wardova metoda: v vsakem koraku združi tisti dve skupini, pri katerih bo po opravljeni združitvi najmanjša vsota kvadratov razdalj od centra nove skupine do vseh njenih elementov.

Hierarhično združevanje kot rezultat vrne hierarhično drevo (angl. dendrogram), v katerem je podana hierarhija povezovanja elementov v skupine. Na eni strani hierarhičnega drevesa so vsi primeri vsak v svoji skupini, na drugi so vsi primeri združeni v eno skupino. Razporeditev v skupine, kakršno običajno iščemo, je nekje vmes. Dokončno razporeditev v skupine določimo z izbiro mesta, na katerem presekamo drevo in dobimo skupine, kot so razporejene na tistem mestu. Postopek izgradnje hierarhičnega drevesa opravimo enkrat, nato pa lahko večkrat izbiramo, kje odrežemo drevo in s tem dobimo različno število skupin.

4.3 Metoda voditeljev

Združevanje v skupine z metodo voditeljev (angl. k-means clustering) je druga metoda združevanja, ki smo jo uporabili. Pred začetkom združevanja podamo število skupin k, ki jih želimo. Postopek združevanja je naslednji. Najprej naključno izbere k elementov iz množice vseh elementov in vsakega dodeli v svojo skupino. Tako imamo k skupin s po enim elementom. Nato ponavlja naslednji postopek:

- vsak element v zbirki priredi najbližji skupini, glede na razdaljo med elementom in centrom skupine;
- izračuna center oziroma povprečno vrednost za vsako skupino, glede na elemente, ki jih ta vsebuje.

Ta postopek ponavlja, dokler se kakšen element še premakne med skupinami [9]. Dobimo k skupin, v katere so združeni elementi.

Poglavje 5

Eksperimentalno ovrednotenje

5.1 Metodologija

5.1.1 Ugotavljanje uspešnosti iskanja podobnih fragmentov

Predstavili smo različne barvne modele in mere za merjenje razdalje med histogrami. Pri izgradnji histograma lahko izberemo različno število polj v histogramu. Možnih je veliko kombinacij zgoraj naštetih možnosti, v naši aplikaciji pa bi radi uporabili tisto, ki bo najbolje delovala.

V naši zbirki je 5172 enobarvnih fragmentov, za katere pa nimamo podatkov, kateri so si med seboj podobni. Če bi želeli ocenjevati uspešnost delovanja algoritmov na teh fragmentih, bi morali vsakemu fragmentu ročno poiskati najbolj podobne fragmente. Vendar bi bilo to zelo zamudno opravilo. Ker pa so fragmenti naključnih oblik in velikosti, jih za potrebe ocenjevanja digitalno razdelimo na manjše fragmente. Za te manjše fragmente pa poznamo nekaj njihovih sosedov, to so fragmenti, s katerimi so tvorili skupni prvotni fragment.

S slike vsakega fragmenta smo najprej odrezali prozorni del, tako da se je fragment dotikal vseh štirih robov slike. Nato smo ga razdelili na štiri enako velike dele, v dve vrstici in dva stolpca. Tako smo dobili manjše fragmente,

za katere poznamo sosede. Za vsak fragment poznamo dva njegova soseda, s katerima ima skupno eno stranico, ima pa vsak manjši fragment še dva soseda na zunanjih stranicah, ki ju ne poznamo. Ker smo fragmente razrezovali na manjše dele z ravnimi pravokotnimi črtami, moramo upoštevati, da realni primeri niso tako lepo razkosani, saj sosednja fragmenta nimata vedno povsem enake oblike roba in je zato med dvema fragmentoma še nekaj praznega prostora. Tak način testiranja je primeren le, če fragmente primerjamo le po barvi fragmenta, ne pa tudi po obliki ali prostorski porazdelitvi barve.

Tako smo dobili testno množico, na kateri lahko preizkušamo delovanje algoritma. Izbrati moramo še način ocenjevanja. Zaželeno je, da so sosednji fragmenti ocenjeni kot najbolj podobni, a obenem vemo, da ima vsak manjši fragment še vsaj dva soseda, ki ju ne poznamo. Ta soseda sta lahko povsem upravičeno uvrščena pred soseda, ki ju poznamo, zato smo pri ocenjevanju dali točke algoritmu, če je sosednji fragment uvrstil vsaj med prvih deset najbolj podobnih fragmentov. Vsak manjši fragment ima dva soseda, s katerima ima skupno stranico, ostane pa še tretji sosed, ki je v razrezu diagonalno od njega in se z njim stika le v eni točki, a smo zaradi ohranjanja preprostosti vse tri sosede obravnavali enakovredno. Za uvrstitev fragmenta na k-to mesto, kjer je $k = 1, \ldots, 10$, je algoritem prejel $11 - k$ točk. Testiranje smo opravili na 1000 naključno izbranih fragmentih, ki smo jih razdelili na štiri dele. Vseh uporabljenih manjših fragmentov je bilo tako 4000. Vseh možnih doseženih točk je $(10 + 9 + 8) \cdot 4000 = 108.000$ in jih dosežemo v primeru, ko so vsi trije sosedje vseh manjših fragmentov na prvih treh mestih. Kot mero uspešnosti iskanja podobnih fragmentov uporabimo delež doseženih točk od vseh možnih.

5.1.2 Ugotavljanje uspešnosti združevanja v skupine

Od vsake skupine pričakujemo, da vsebuje podobne fragmente. Uspešnost združevanja v skupine smo ocenjevali tako, da smo za vsak fragment v skupini preverili, koliko njegovih najbolj podobnih fragmentov je v skupini. Tako ocenjevanje smo uporabili za primerjavo velikosti skupin in tipov povezav

med skupinami pri hierarhičnem združevanju oziroma uporabljenih razdalj pri metodi voditeljev.

Testiranje poteka tako, da za vsak fragment v skupini preverimo, koliko njegovih najbolj podobnih fragmentov je v isti skupini z njim. Število najbolj podobnih k, za katere podelimo točke, lahko poljubno izberemo. Ker bodo v aplikaciji podobni enobarvni fragmenti prikazani ob strani zaslona in si želimo, da so najbolj podobni prikazani pri vrhu seznama in bodo vidni brez uporabnikovega premikanja po seznamu ali z zelo malo premikanja, smo se odločili za k izbrati vrednost 10. Sicer je število enobarvnih fragmentov, ki so naenkrat vidni na zaslonu, odvisno tudi od njihove oblike, velikosti in zasuka. Običajno je to število 5 do 6. Za prisotnost k-tega najbolj podobnega fragmenta v njegovi skupini, kjer je $k = 1, \ldots, 10$, prejme algoritem $11 - k$ točk. Skupaj je možnih $5172 \cdot (10 + 9 + \cdots + 1)$ točk, to je 284.460. Kot mero uspešnosti združevanja fragmentov v skupine uporabimo delež doseženih točk od vseh možnih. Pri takem ocenjevanju moramo biti pozorni na vpliv velikosti skupin na rezultat, saj je pri večjih skupinah večja verjetnost, da se v skupini nahajajo podobni fragmenti. Cilj je najti razvrstitev v skupine z največjim deležem doseženih točk, katere velikost največje skupine ne presega največje velikosti zlepka sličic.

5.2 Rezultati

5.2.1 Rezultati primerjav mer razdalje med histogrami

Dobljeni rezultati pri testiranju različnih kombinacij barvnih modelov, velikosti histogramov in razdalj med histogrami so prikazani v tabeli 5.1. Največji odstotek možnih točk je bil dosežen pri uporabi Hellingerjeve razdalje, barvnega modela HSV in številu polj v histogramu 256. Le malenkost slabši rezultat je bil dosežen pri uporabi histograma s 128 polji. Manjše število polj pa pomeni hitrejše računanje podobnosti in manjšo porabo pomnilnika pri primerljivih rezultatih. Zato je smiselno v praksi uporabljati histograme s 128 polji. V rezultatih so opazni tudi določeni vzorci. Pri uporabi barvnega

modela HSV in histograma z 256 polji dobimo najboljše rezultate pri vseh preizkušenih merah razdalje. Uporaba barvnega modela RGB je pri vseh merah razdalje najslabša s precej občutno razliko, medtem ko barvni model LAB ne zaostaja veliko za modelom HSV. Med merami razdalje se najslabše odreže evklidska razdalja, medtem ko je Hellingerjeva v vseh primerih najboljša.

Tabela 5.1: Rezultati za različne mere razdalje, barvne modele in velikosti histogramov, razvrščeni po uspešnosti (v % doseženih možnih točk)

Mera	HSV256	HSV128	LAB256	HSV64	LAB128	LAB64	RGB128	RGB256	RGB64
Hellingerjeva	60,1	59,7	58,4	58,6	58,1	57,2	48,5	49,1	49,1
χ^2	56,8	56,5	55,8	55,6	55,5	54,8	46,7	47,2	47,7
mahnattanska	55,9	55,2	54,2	53,6	53,6	52,1	44,5	44,5	44,5
korelacija	53,0	51,9	51,5	49,9	50,8	48,9	42,9	42,9	42,7
evklidska	48,9	48,7	48,3	48,2	48,0	47,2	42,9	42,9	42,8

5.2.2 Rezultati primerjav algoritmov združevanja v skupine

Tabela 5.2 prikazuje rezultate pri hierarhičnem združevanju v skupine. Uporabljene povezave med skupinami so: povprečna povezava, najdaljša povezava, najkrajša povezava in Wardova metoda. Prikazani so rezultati za od 40 do 100 skupin s koraki po 20. Prva vrednost v polju tabele je dosežen odstotek vseh možnih točk, v oklepaju sledi podatek o največji velikosti skupine pri uporabljenem združevanju v skupine.

Hierarhično združevanje v skupine z uporabo najkrajše povezave med skupinama sicer doseže visok odstotek podobnih fragmentov skupaj v skupinah, a velikost največje skupine je okrog 5000 fragmentov, torej obsega skoraj vse fragmente, medtem ko imajo ostale skupine le po en fragment. Takšna naloga očitno ni primerna za uporabo te povezave.

Če ne upoštevamo primerov z najkrajšo povezavo, najvišji odstotek doseže

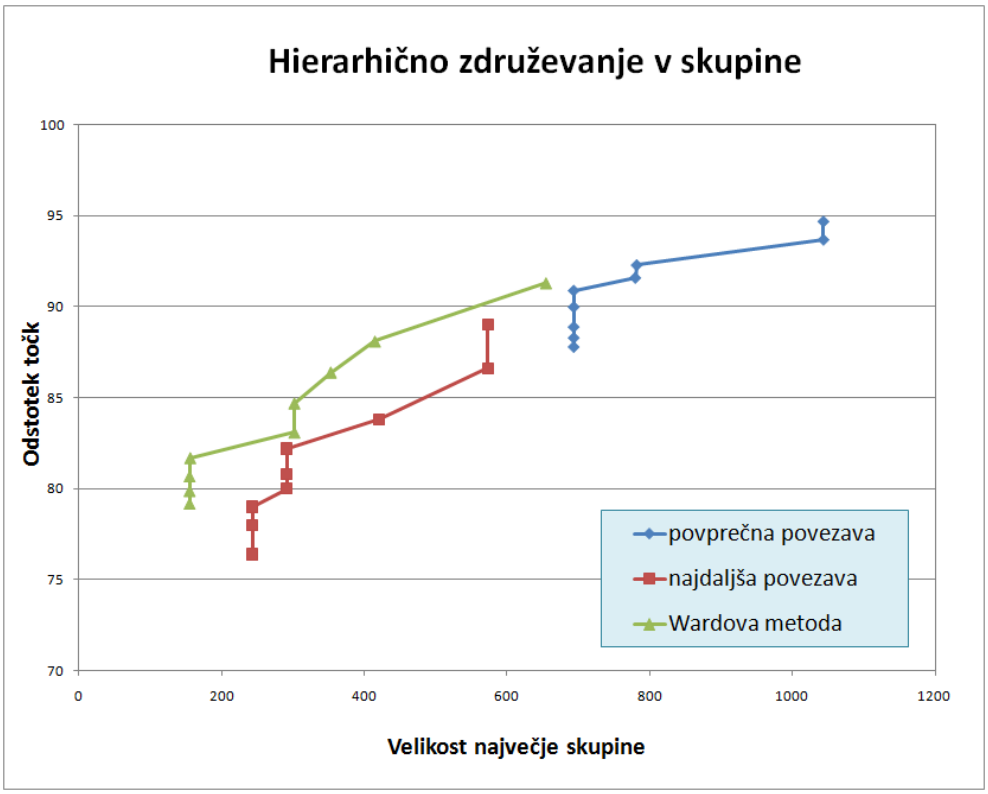
Tabela 5.2: Rezultati testiranja hierarhičnega združevanja v skupine. Podani so odstotki doseženih možnih točk, v oklepajih so velikosti največjih skupin

Način združevanja	Število skupin			
	40	60	80	100
povprečna	92,3% (782)	90,9% (694)	88,9% (694)	87,8% (694)
najdaljša	83,8% (421)	80,8% (291)	79% (243)	76,4% (243)
najkrajša	99,1% (5121)	98,7% (5098)	98,2% (5070)	97,8% (5051)
Wardova	86,4% (353)	83,1% (302)	80,7% (155)	79,2% (155)

povprečna povezava, sledi Wardova metoda in nato najdaljša povezava. Vendar pa so bili rezultati doseženi pri različno velikih največjih skupinah.

Dosežke pri uporabi različnih povezav med skupinami ob hierarhičnem združevanju v skupine primerjamo na sliki 5.1. Na grafu je prikazano število doseženih odstotkov točk na osi y in velikosti največje skupine na osi x. Velikosti skupin so od 20 do 100, s koraki po 10. Rezultati doseženi z uporabo najkrajše povezave zaradi prevelikih velikosti največjih skupin niso prikazani. V tem primeru hierarhično združevanje z uporabo najdaljše povezave dosega pri primerljivih velikostih največjih skupin nižji odstotek točk kot pri uporabi Wardove metode. Pri uporabi povprečne povezave pa so velikosti največjih skupin v vseh primerih prevelike, saj se začnejo pri približno 700 elementih, kar je preveč za naše potrebe. Pri 60 skupinah je velikost največje skupine 694 elementov, kar predstavlja kar 13% vseh elementov. Najboljša metoda hierarhičnega združevanja po naših kriterijih je z uporabo Wardove metode, ki doseže dobre rezultate pri majhni največji velikost skupin.

Pri metodi voditeljev smo primerjali različne načine merjenja razdalj med centrom skupine in elementi. Običajno se uporabljata manhattanska ali evklidska razdalja. Preizkusili smo še uporabo Hellingerjeve razdalje, saj bo ta razdalja uporabljena tudi za iskanje najbližje skupine za novo sliko fragmenta. Dobljeni rezultati za razporeditve s 40 do 100 skupinami s koraki po 20 so zbrani v tabeli 5.3. Prva vrednost je odstotek doseženih možnih točk, v

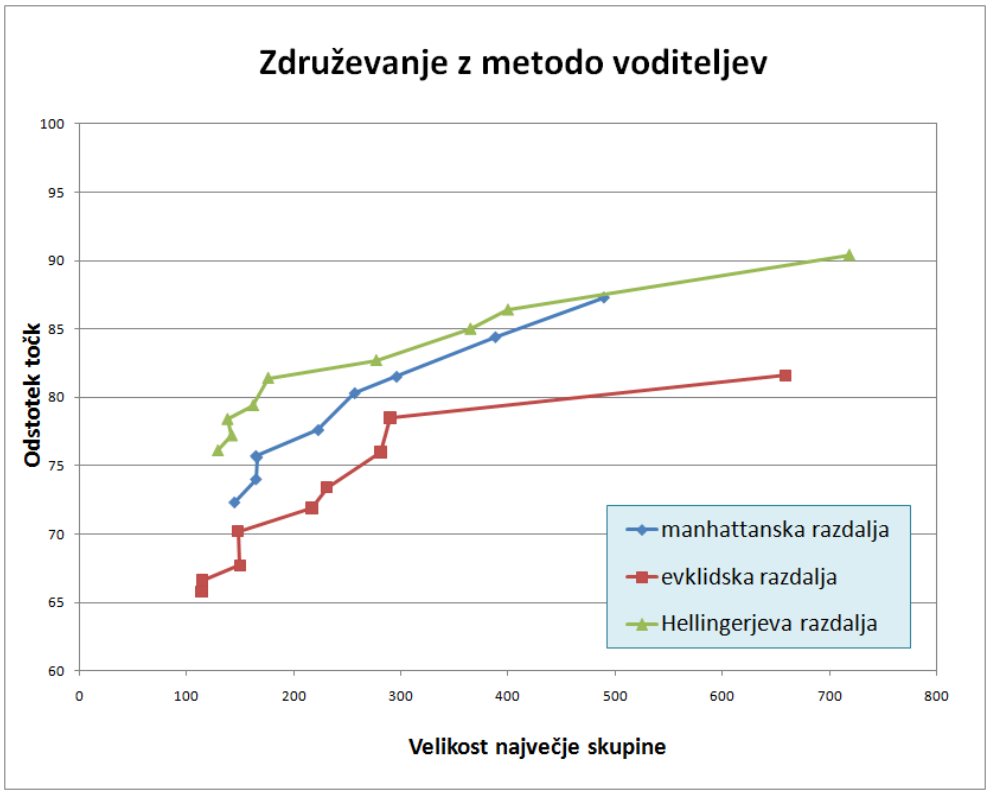


Slika 5.1: Odstotek točk glede na velikost skupin pri različnih načinih hierarhičnega združevanja v skupine

oklepaju pa je zapisana velikost največje skupine, dobljene po tem postopku.

Tabela 5.3: Rezultati testiranja združevanja v skupine z metodo voditeljev. Podani so odstotki doseženih možnih točk, v oklepajih so velikosti največjih skupin

Mera razdalje	Število skupin			
	40	60	80	100
manhattanska	81,5% (296)	77,6% (223)	75,6% (166)	72,3% (145)
evklidska	76% (281)	71,9% (217)	67,7% (150)	65,8% (114)
Hellingerjeva	85% (365)	81,4% (176)	78,4% (138)	76,1% (129)



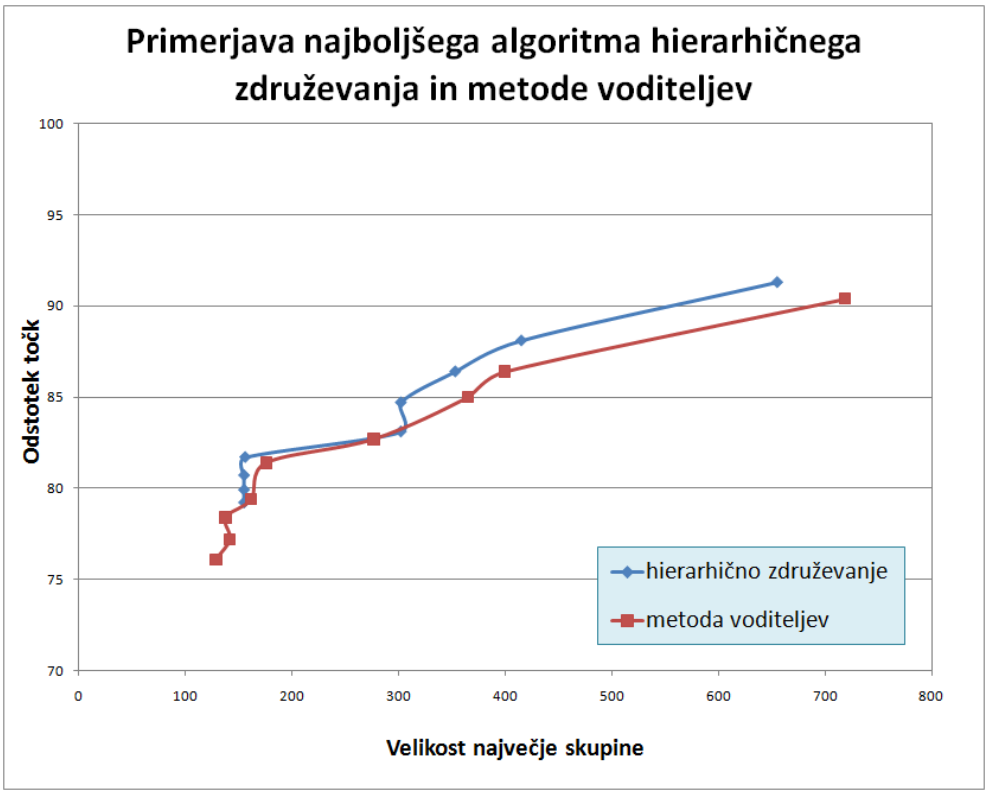
Slika 5.2: Odstotek točk glede na velikost skupin pri združevanju v skupine z metodo voditeljev

Na sliki 5.2 primerjamo dosežene odstotke točk v odvisnosti od velikosti največje skupine za različne mere razdalje, uporabljene pri metodi voditeljev.

Pri združevanju z metodo voditeljev najboljše rezultate dosežemo z uporabo Hellingerjeve razdalje. V kolikor uporabimo Hellingerjevo razdaljo, je tudi zagotovljeno, da so vsi elementi ob kasnejšem uvrščanju v skupine uvrščeni v tisto skupino, v kateri se nahajajo, saj uporabimo enako razdaljo pri gradnji skupin, kot tudi kasneje pri izbiri skupin.

5.2.3 Izbira najboljšega algoritma

Na sliki 5.3 primerjamo najboljši algoritem za hierarhično združevanje, to je z uporabo Wardove metode, in najboljši algoritem za združevanje z metodo



Slika 5.3: Primerjava hierarhičnega združevanja z Wardovo metodo in združevanja z metodo voditeljev z uporabo Hellingerjeve razdalje

voditeljev, to je z uporabo Hellingerjeve razdalje. Hierarhično združevanje se odreže nekoliko bolje, saj dosega višji odstotek točk.

Moramo pa upoštevati še eno lastnost problema iskanja najbližje skupine za dani fragment. Ko bomo imeli skupine sestavljene, bomo za vsako izračunali povprečni histogram ter sestavili zlepek sličic. Ko bomo dobili v poizvedbo novo sliko in izračunali njen histogram, bo najbližja skupina izbrana na podlagi najbolj podobnega histograma skupine. Združevanje z metodo voditeljev zagotavlja, da za vse fragmente velja, da jim je izmed vseh povprečnih histogramov skupin najbližji ravno povprečni histogram njihove skupine. To je zagotovljeno s tem, da se v prvem koraku metode voditeljev vsak element vključi v tisto skupino, katere center skupine mu je najbližji.

Center skupine pa se izračuna ravno kot povprečni histogram skupine. Medtem ko hierarhično združevanje te lastnosti ne zagotavlja in se tudi v praksi izkaže, da so nekateri elementi v skupinah, katerih povprečna vrednost jim ni najbližja. Če želimo takšno razporeditev uporabiti v našem primeru, moramo fragmente prestaviti v skupino, ki jim je najbliže, ter nato ponovno izračunati centre skupin, ter postopek ponavljati dokler prihaja do sprememb. Ta postopek pa je enak postopku metode voditeljev. Izkaže se, da so rezultati tako popravljenih skupin v vseh primerih, ne glede na to, s katero razporeditvijo smo začeli, zelo podobni med seboj in tudi prvotnemu združevanju z metodo voditeljev z uporabo Hellingerjeve razdalje. Tako je najbolj smiselno v praksi uporabiti metodo voditeljev s Hellingerjevo razdaljo.

Poglavje 6

Sklep

6.1 Zaključne ugotovitve

Pri prvi nalogi, ki smo si jo zadali v tem diplomskem delu, to je poiskati najustreznejši način iskanje podobnih fragmentov, se je najbolje obnesel postopek z uporabo Hellingerjeve razdalje v kombinaciji z barvnim modelom HSV in histogramom z 256 polji za vsak barvni kanal. Le malenkost slabše rezultate smo dobili pri uporabi histograma s 128 polji, manjše število polj pa pomeni hitrejše računanje podobnosti in manjšo porabo pomnilnika pri primerljivih rezultatih. Zato je smiselno uporabljati histograme s 128 polji.

Druga naloga je bila poiskati najboljši način za združevanje fragmentov v skupine. Pri tej nalogi se je izkazalo, da je najprimernejša metoda metoda voditeljev z uporabo Hellingerjeve razdalje med centrom skupine in fragmenti.

Z uporabo izbranih metod bomo lahko v aplikaciji e-Pedius dodali funkcionalnost iskanja podobnih enobarvnih fragmentov in zagotovili visoko hitrost delovanja aplikacije ter s tem dobro uporabniško izkušnjo.

Pričakujemo, da se bo z uporabo aplikacije sestavilo najdene fragmente stenskih poslikav v prvotno obliko, tako pri aktualnem projektu Turška mačka kot tudi pri prihajajočih novih projektih.

6.2 Nadaljnje delo

Pri predstavitvi slik fragmentov z barvnima modeloma HSV in LAB bi lahko uporabili različne uteži za različne barvne kanale, saj ti niso nujno enako pomembni. Na primer razlika med barvnimi odtenki je morda pomembnejša kot razlika med intenziteto barv. S testiranjem različnih obtežitev bi lahko poiskali najbolj primerne uteži.

Z izidom aplikacije e-Pedius lahko pričakujemo pridobitev novih informacij o fragmentih, saj nam vsaka postavitev, ki jo uporabniki sestavijo ali ocenijo, pove nekaj o uporabljenih fragmentih. Iz sestavljenih postavitev uporabnikov lahko ugotovimo, kateri fragmenti so bili največkrat postavljeni skupaj. Ti so si verjetno med seboj podobni in bi zato lahko služili za ocenjevanje uspešnosti iskanja podobnih fragmentov. Shranjevali bi lahko tudi informacije o tem, katere enobarvne fragmente so v določenem primeru uporabniki izbrali s seznama ponujenih. Za te lahko sklepamo, da so podobni označenemu območju in to informacijo uporabimo pri nadaljnjem izpopolnjevanju algoritmov.

Obravnavanje enobarvnih fragmentov v aplikaciji e-Pedius pa bi lahko nadgradili z lastno metodo, ki bi poleg barvnih histogramov upoštevala tudi razporeditve barv na fragmentih oziroma njihove teksture.

Literatura

[1] S. Cha. "Comprehensive survey on distance/similarity measures between probability density functions", *International Journal of Mathematical Models and Methods in Applied Sciences*, let. 1, št. 4, str. 300–307, 2007.

[2] G. Čepin, T. Tušar, B. Filipič. "Iskanje podobnih enobarvnih fragmentov pri množičnem sestavljanju stenskih poslikav". Zbornik 22. mednarodne Elektrotehniške in računalniške konference ERK 2013 (ur. B. Zajc, A. Trost), zv. B, str. 73–76, 2013.

[3] J. Demšar, B. Zupan, G. Leban, T. Curk. "Orange: from experimental machine learning to interactive data mining", v zborniku: Knowledge Discovery in Databases: PKDD 2004 (ur. J.-F. Boulicaut in sod.), Lecture Notes in Computer Science 3202, str. 537–539, 2004.

[4] e-Pedius: Podpora množičnemu sestavljanju fragmentov stenskih poslikav. Dostopno na: `http://e-pedius.si/`

[5] B. Filipič, M. Mlakar, E. Dovgan, T. Tušar. "Razvoj sistema za računalniško podprto evidentiranje in sestavljanje fragmentov stenskih poslikav", Zbornik 14. mednarodne multikonference Informacijska družba IS 2011, zv. A (ur. M. Bohanec in sod.), str. 45–48, 2011.

[6] Histogram Comparison. OpenCV documentation. Dostopno na: `http://docs.opencv.org/doc/tutorials/imgproc/histograms/histogram_comparison/histogram_comparison.html`

[7] S. Krishnamachari, M. Abdel-Mottaleb. "A scalable algorithm for image retrieval by color", v zborniku: 1998 International Conference on Image Processing, ICIP-98 (ur. B. Werner), zv. 3, str. 119–122, 1998.

[8] Pedius: Sistem za evidentiranje in sestavljanje fragmentov stenskih poslikav. Dostopno na: `http://dis.ijs.si/ci/pedius/`

[9] P. Tan, M. Steinbach, V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[10] M. Tkalčič, J. F. Tasič. "Colour spaces - perceptual, historical and applicational background", v zborniku: The IEEE Region 8 EUROCON 2003, Computer as a Tool (ur. B. Zajc, M. Tkalčič), str. 304–308, 2003.

[11] I. H. Witten, E. Frank, M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*, 3. izdaja, Morgan Kaufmann, 2011.