UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Borut Ogrinc

Vizualni sistem za iskanje in sledenje reklamnih panojev

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: 01520/2008
Datum: 15.10.2008

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

Kandidat: **BORUT OGRINC**

Naslov: **VIZUALNI SISTEM ZA ISKANJE IN SLEDENJE REKLAMNIH PANOJEV**

A VISION-BASED SYSTEM FOR DETECTING AND TRACKING OF ADVERTISEMENT BILLBOARDS

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Proučite problematiko vizualnega sistema za iskanje in sledenje reklamnih panojev v video posnetkih športnih prenosov. Načrtajte in implementirajte ustrezne algoritme, pri čemer se osredotočite na trenutno najsodobnejše postopke za detekcijo in sledenje. Analizirajte učinkovitost teh algoritmov ter jih eksperimentalno ovrednotite na ustreznih video posnetkih.

Mentor:
prof. dr. Aleš Leonardis

Fakulteta za računalništvo in informatiko
UNIVERZA V LJUBLJANI
2

Dekan:
prof. dr. Franc Solina

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Borut Ogrinc,

z vpisno številko 63020118,

sem avtor diplomskega dela z naslovom:

Vizualni sistem za iskanje in sledenje reklamnih panojev.

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom
  prof. dr. Aleša Leonardisa
- 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 22.1.2009

Zahvala

Zahvaljujem se mentorju, prof. dr. Alešu Leonardisu.

Kazalo

Terminologija

Izvorni pano je reklamni pano, ki se pojavlja v originalnem videu prenosa tekme.

Novi pano je reklamni pano, ki se vstavi v video na mestu izvornega panoja.

Primer panoja je slika izvornega panoja, ki se poda algoritmu za iskanje in zamenjavo panojev. Na osnovi primerov panoja algoritem išče izvorne panoje v videu.

Regija je del slike.

Ujemajoči regiji sta regiji, najdeni na različnih slikah istega predmeta, ki pokrivata isti del predmeta na slikah. Metodo za določanje ujemajočih regij smo uporabili pri detekciji panojev v videu. Iščemo ujemajoče regije primera panoja in panoja na sliki iz videa.

Ujemajoči točki sta točki na različnih slikah istega predmeta in predstavljata isto točko na predmetu.

RANSAC ali RANdom SAmple Consensus je metoda za robustno ocenjevanje matematičnega modela na podatkih, ki vsebujejo velike napake. Uporabili smo jo za določanje homografije ne podlagi parov ujemajočih točk.

Kovariantna regija je regija, ki je določena na sliki na tak način, da se spreminja skladno z nekim tipom transformacije. To pomeni, da regije določene na originalni in transformirani sliki, obsegajo iste dele predmeta na sliki — so ujemajoče.

Slikovni element je najmanjši del slike. Obsega podatek o lokaciji in intenziteti na tisti lokaciji.

Rob je meja v obliki črte, na kateri se intenziteta slike v eni smeri naglo spremeni.

Afina transformacija je transformacija, ki ohranja kolinearnost (točke, ki ležijo na premici pred transformacijo, bodo ležale na njej tudi po transformaciji) in razmerja razdalj (sredinska točka dela premice bo ostala sredinska tudi po transformaciji).

Povzetek

Predmet diplomske naloge je izdelava vizualnega sistema za iskanje in sledenje reklamnih panojev v videu, ki je del sistema za zamenjavo reklamnih panojev pri prenosih športnih dogodkov. V diplomi smo predstavili dosedanje raziskave na širšem področju in za temelj izbrali metodo, ki so jo uporabili G. Medioni, G. Guy, H. Rom in A. François ter predstavili v članku "Real-time billboard substitution in a video stream" [17]. Sistem predstavljen v članku smo nadgradili z najmodernejšimi metodami za iskanje in sledenje, razvili prototip ter ga testirali.

Iskanje panojev smo realizirali z iskanjem ujemajočih kovariantnih regij v videu in na primerih panojev. Postopek iskanja panoja poteka v naslednjih fazah: detekcija kovariantnih regij na panoju in na primeru panoja, izračun opisnikov regij, ujemanje opisnikov in ocenjevanje homografije glede na ujemajoče regije.

Preučili smo različne detektorje afino kovariantnih regij in opisnike regij. Na podlagi testov smo se odločili uporabiti detektor maksimalno stabilnih regij (MSER) in opisnik Shape Context. Določanje meje panoja smo realizirali z robustnim ocenjevanjem homografije med primerom panoja in panojem v videu. Uporabljeni sta bili metodi DLT (direct linear transformation) za izračun homografije in RANSAC (Random Sample Consensus) za robustnost kljub prisotnosti napačno določenih ujemajočih regij. Sledenje panojev smo realizirali s sledenjem regij panoja, za katerega smo uporabili metodo učinkovitega sledenja regij MSER.

Razvili smo prototip in testirali njegovo natančnost ter hitrost. Opravili smo test natančnosti določanja mej panojev, uspešnost sledenja regij MSER in časovne zahtevnosti posameznih komponent prototipa.

Posodobljen prototip sistema za iskanje in sledenje predstavlja dobro osnovo za nadaljnja raziskovanja na področju iskanja in zamenjave reklamnih panojev.

Ključne besede: Zamenjava reklamnih panojev, iskanje, sledenje, kovariantne regije, opisniki regij, homografija, RANSAC.

Abstract

The subject of the thesis is an implementation of a vision-based system for detecting and tracking of advertisement billboards, which is a part of a billboard replacement system for sport events broadcasts. In the thesis we present previous research concerning billboard detection, among which we choose a method proposed by G. Medioni, G. Guy, H. Rom and A. François in the article "Real-time billboard substitution in a video stream" [17]. We improved the mentioned system with state of the art methods for detection and tracking, developed a prototype of the system and tested its performance.

Billboard detection was performed by finding corresponding regions in video frames and reference billboard images. Process of billboard detection is divided into the following stages: detection of covariant regions in video frame and reference image, computing of region descriptors, matching the descriptors and estimating a homography given a set of matched regions.

We studied and tested different affine covariant region detectors and region descriptors. Based on testing we decided to use Maximally Stable Extremal Region (MSER) detector and Shape Context descriptor. Billboard boundaries were set by robust estimation of a homography between a reference billboard image and billboard in video frame. For homography estimation we used DLT (direct linear transformation) with RANSAC (Random Sample Consensus) for added robustness in cases of falsely matched regions. Efficient Maximally Stable Extremal Region (MSER) Tracking was used for billboard tracking.

We developed a prototype and tested its accuracy and speed. We tested the accuracy of billboard boundary estimation, proportion of correctly tracked regions and processor time spent on different tasks.

The developed updated system prototype for billboard detection and tracking is a good basis for future studies on the field of billboard detection, tracking and substitution.

Key words:

Billboard substitution, detection, tracking, covariant regions, region descriptors, homography, RANSAC.

Poglavje 1

Uvod

1.1 Motiv za izdelavo sistema za zamenjavo reklamnih panojev

Oglaševanje je v profesionalnem športu izredno pomembno. Na svetovnem prvenstvu v nogometu 2006 je FIFA s trženjem pridobila okoli 1,9 milijarde evrov prihodka, od uradnih sponzorjev pa še dodatnih 700 milijonov. Pomemben nosilec oglaševalskih sporočil so panoji ob robu igrišča. Oglaševanje na svetovnih prireditvah je smiselno le za multinacionalke, ker imajo le te dovolj velik delež potencialnih kupcev med gledalci, da je investicija upravičena. Letalski prevoznik Emirates Airlines je primer podjetja, ki je uporabilo tovrstno oglaševanje na svetovnem prvenstvu. Njihov oglas je dosegel potencialne kupce v državah, s katerimi je prevoznik imel vzpostavljene letalske povezave, v drugih državah ta oglas ni imel večjega vpliva. Oglaševalski panoji ob igrišču bi lahko bili bolje izkoriščeni, če bi gledalci na različnih koncih sveta, ob spremljanju prenosa, videli različne oglase. Uporaba sistema za zamenjavo oglasnih sporočil na panojih, bi omogočila, da oglas doseže izbrano publiko. Tako bi se povečalo zanimanje za tovrstno oglaševanje tudi med lokalnimi podjetji in s tem tudi zaslužek ob enaki površini oglasnega prostora.

Sistem za zamenjavo bi moral v vhodnem videu najprej razpoznati izvorne panoje, natančno določiti njihove meje in jih v videu zamenjati za nove. Za komercialno rabo, bi moral spremenjen video izgledati dovolj pristno, da gledalec ne bi opazil, da je prišlo do zamenjave. Druga zahteva je dovolj hitro delovanje sistema za prenose v živo.

Slika 1.1 prikazuje delovanje sistema na primeru prenosa tekme v tenisu, kjer je originalni oglas JPMorgan zamenjan z lokalnim NLB.

Slika 1.1: Primer uporabe sistema za zamenjavo panojev

1.2 Cilj naloge

V nalogi smo predstavili in izdelali vizualni sistema za iskanje in sledenje reklamnih panojev v videu, ki je del sistema za zamenjavo reklamnih panojev pri prenosih športnih dogodkov.

Predstavili smo dosedanje raziskave na področju iskanja in zamenjave panojev. Opisali smo dva sistema za beleženje pojavljanja panojev med prenosi tekme in metodo za zamenjavo panojev, ki so jo G. Medioni, G. Guy, H. Rom in A. François predstavili v članku "Real-time billboard substitution in a video stream" [17]. Slednjo smo vzeli za temelj in jo nadgradili z modernimi metodami.

Prvi cilj je bila posodobitev iskanja panojev. Problem smo rešili z iskanjem ujemanj med primeri panojev in panoji v videu. Najprej smo uporabili detektor za določanje kovariantnih regij. Tako določene regije so stabilne pri transformacijah v prostoru in so odporne na spremembe v osvetlitvi, kar zagotavlja, da bodo regije, detektirane na primeru panoja, obsegale ista območja kot tista na panoju v videu.

Da lahko regije panoja v videu povežemo z regijami primera panoja, potrebujemo način za primerjanje delov slike, ki ju regiji obsegata. Potrebujemo opis slike, ki bo primerljiv v različnih pogojih — torej da bo neodvisen od

oblike in pogojev osvetlitve regije. Opise regij nato primerjamo in določimo ujemajoče regije med primerom panoja in sliko iz videa.

Izbira detektorja in opisnika regij je ključna pri kvaliteti sistema, zato smo testirali uspešnost pravilnega določanja ujemajočih regij pri različnih kombinacijah detektorja in opisnika. Odločili smo se za kombinacijo detektorja maksimalno stabilnih ekstremnih regij in opisnika Shape Context.

Naslednja naloga je na podlagi ujemajočih regij določiti robove panoja v videu. Problem smo rešili z robustnim ocenjevanjem homografije med primerom panoja in panojem v videu. Pogosto se zgodi, da so ujemajoče regije napačno določene, kar pomeni, da so med pravimi pari ujemajočih točk tudi taki z zelo veliko napako. Za računanje homografije iz parov ujemajočih točk smo uporabili metodo DLT (direct linear transformation) v kombinaciji z metodo RANSAC (Random Sample Consensus), ki zagotavlja, da dobimo dober približek kljub prisotnosti napačnih parov.

Po detekciji panojev potrebujemo način za sledenje panojev v videu, ki bo dovolj hiter in bo deloval zanesljivo, kljub morebitnem delnem zakritju panoja. Izbrali smo metodo učinkovitega sledenja regij MSER.

Opisali smo omenjene metode in jih implementirali v prototipu sistema. Opravili smo test natančnosti določanja mej panojev, uspešnost sledenja regij MSER in časovne zahtevnosti posameznih komponent prototipa.

Na koncu smo kritično ovrednotili delo in nakazali ideje za prihodnje raziskave na tem področju.

1.3 Dosedanje delo na tem področju

1.3.1 Sistemi merjenja časa prisotnosti panojev v prenosu tekme

Obstajajo sistemi za zaznavanje prisotnosti posameznega panoja v prenosu, ki zagotavljajo bolj natančno obračunavanje cene tovrstnega oglaševanja. Sistema, ki sta primerna za obravnavo, sta opisana v člankih “Detection of on-field advertisement billboards from soccer telecasts” [25] in “Billboard advertising detection in sport TV” [3]. Sistema sta glede na funkcionalnost primerna za obravnavo problema, ki ga želimo v nalogi raziskati, le da ni dovolj poudarjena zahteva po natančnosti pri določanju lege panoja v videu.

Algoritem, predstavljen v prvem članku, na sliki iz videa poišče zanimivo regijo tako, da določi premico, ki loči panoje in igralno površino. Glede na njo določi

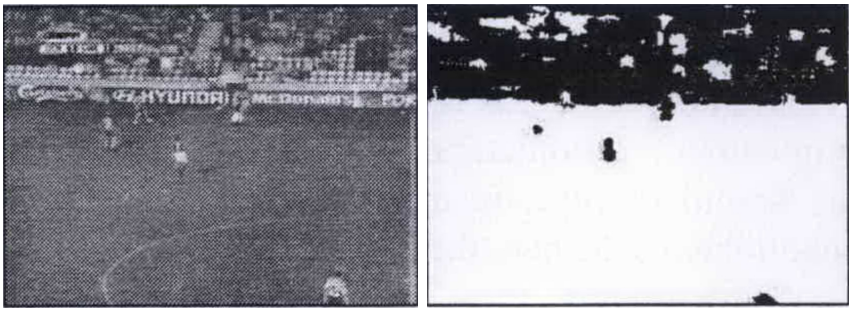
drugo premico, ki predstavlja zgornji rob panojev. Postopek določanja obeh je naslednji: sliko iz videa binariziramo na dva dela, na zeleni in nezeleni del, kjer zeleni predstavlja travnato površino, nezeleni pa ostalo. Nad binarizirano sliko (slika 1.2(b)) uporabimo Sobelov algoritem [6] za zaznavo robov in za njim, Houghov transform [7] za zaznavanje premic. Dobimo množico premic, ki predstavljajo meje med zeleno in nezeleno regijo — torej domnevne spodnje robove panojev (slika 1.2(c)). Nato premice zaznamo še na originalni sliki (slika 1.2(a)). Vsaki premici, ki predstavlja domnevni spodnji rob, izberemo premice, ki so njej vzporedne (slika 1.2(d)) in najbližjo izberemo za domnevni zgornji rob panojev.

Domnevni spodnji robovi in njim ustrezni zgornji določajo interesna območja (slika 1.3(a)). Ta transformiramo v pravokotnik (slike 1.3(b)) in na njih najdemo navpične robove z detektorjem Canny [4]. Navpične črte predstavljajo domnevne meje med panoji (slika 1.3(c)).

Naslednji korak predstavlja prepoznavo domnevnih panojev glede na primere panojev. Vsak domnevni pano primerjamo z množico primerov panojev. Za opis panoja uporabljamo opisnik v obliki vektorja dolžine 32, ki vsebuje srednje vrednosti in variance intenzitete in tona barv delov panoja. Pano se razdeli na 8 delov, na 2 vrstici in 4 stolpce. Pri primerjavi se izračuna evklidske razdalje med opisnikom posameznega domnevnega panoja in opisniki primerov panojev. Če je najmanjša razdalja pod določeno mejo, domnevni pano klasificiramo kot najbližji izmed primerov panojev.

Rezultati v obravnavanem primeru kažejo 95 odstotno zanesljivost. Pozitiven rezultat je definiran kot zaznavanje pravega panoja z določeno natančnostjo lokacije in kot nezaznavanje, če v posnetku iskani pano ni prisoten. [2] V članku “Billboard advertising detection in sport TV” [3] je zaznavanje zgornjega in spodnjega roba panojev doseženo z enakimi metodami, dodana jim je le hevristika za izločanje napačnih črt. Rezultat je zanimiva regija med zgornjim in spodnjim robom, ki se transformira v pravokotno regijo in binarizira. Regija je opisana kot histogram, v katerem višina stolpca predstavlja delež belih slikovnih elementov v enem stolpcu slikovnih elementov binarizirane regije. Število stolpcev histograma je enako številu slikovnih elementov v horizontalni smeri binarizirane regije. Deli histograma z vrednostmi blizu 0 ali 1 predstavljajo dele regije, kjer prevladuje ena barva, torej ne vsebujejo črk ali oblik. Taki deli predstavljajo domnevne meje med panoji. Panoje prepoznamo s primerjanjem histogramov primerov panojev in histogramov med verjetnimi mejami.

Algoritem je na primeru množice 200 slik dosegel 91 odstotno natančnost. Avtorji prvega članka menijo, da iskanje ujemanj s histogramom ni dovolj

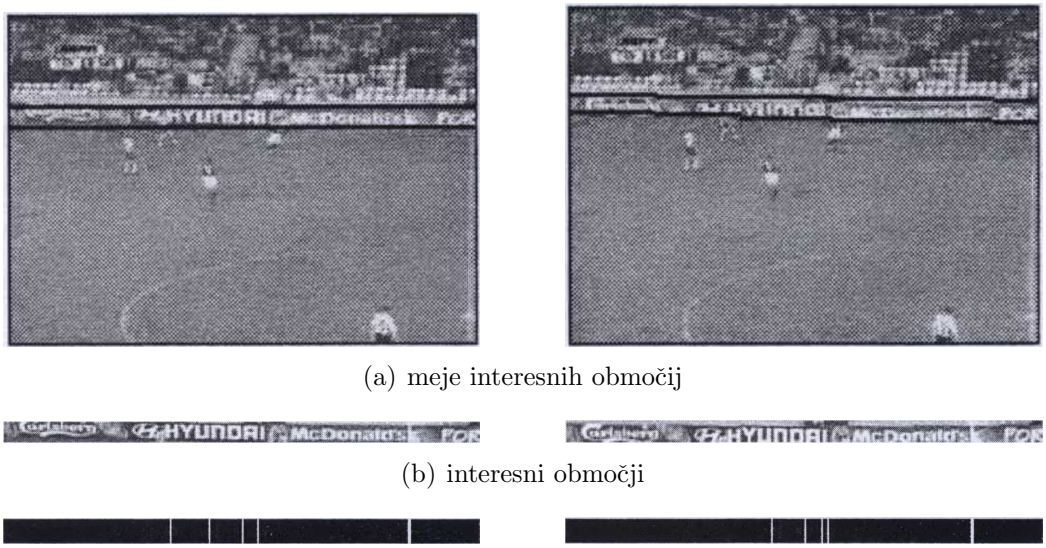


(a) originalna slika iz videa (b) binarizirana slika

(c) domnevna spodnja robova panojev

(d) domnevni zgornji robovi panojev vsakega izmed domnevnih spodnjih robov

Slika 1.2: Primer iskanja zgornjega in spodnjega roba panoja. Vir [25].



(a) meje interesnih območij

(b) interesni območji

(c) domnevne meje med panoji

Slika 1.3: Določanje mej med panoji. Vir [25].

natančno.

1.3.2 Sistem za zamenjavo panojev

Sistem, ki je najbližji našemu področju raziskovanja, so G. Medioni, G. Guy, H. Rom in A. François opisali v članku "Real-time billboard substitution in a video stream" [17]. Avtorji prikažejo implementacijo iskanja, sledenja in zamenjave panojev v realnem času. Sistemu podamo slike primerov panojev in označimo pare značilnih barv, ki se pojavljajo na posameznem panoju.

Pred procesiranjem videa, sistem na slikah primera panoja najde zanimive točke. Vsak trojček najdenih točk se vzame za bazo novega koordinatnega sistema in izračuna nove koordinate vseh točk glede na bazo. Nove koordinate točk se vstavijo v razpršeno tabelo, kjer so koordinate točke ključ in trojček, izbran za bazo, vrednost.

Ob procesiranju se sliki iz videa določi zanimive točke (slika 1.4). Točke, ki so preveč oddaljene od slikovnih elementov značilnih barv, se izloči, da algoritem ni preveč časovno zahteven. Izmed točk se naključno izbere tri, ki predstavljajo bazo novega koordinatnega sistema in pretvori koordinate točk v novi sistem. Pretvorjene koordinate točk sistem išče v razpršeni tabeli. Če koordinate najde v tabeli, njej pripadajoči bazi odda 1 glas. Če obstaja baza z dovolj glasovi, se izračuna afina transformacija med točkami baze in trojčkom

Slika 1.4: Zanimive točke na sliki iz videa. Vir [17].

točk, izbranih na sliki iz videa. Izračun afine transformacije je pravi, če so vse tri naključno izbrane točke na istem panoju. Opisani postopek večkrat ponovimo, da povečamo verjetnost izbire trojčka točk na panoju.

Sledenje je realizirano na enak način kot iskanje, le da se izvaja na zanimivi regiji, ki se izračuna na osnovi globalnega gibanja med zaporednima slikama.

V članku avtorji opišejo zamenjavo panojev v videu, a ne omenjajo problema delno zakritih panojev. V primeru delnega zakritja moramo z novim panojem prekriti le vidni del panoja, opisan postopek pa tega ne reši.

Po analizi virov je to edini članek, ki se direktno ukvarja s problemom iskanja, sledenja in zamenjave reklamnih panojev. Odločili smo se, da bomo na zasnovali sistem, ki bo temeljil na sistemu opisanem v članku in ga nadgradili z najnovejšimi metodami.

1.4 Kratek povzetek poglavij

V drugem poglavju predstavimo delovanja sistema. V tretjem poglavju testiramo kvaliteto različnih detektorjev in opisnikov regij ter ugotavljamo, katera kombinacija bi bila najbolj primerna. Četrto poglavje vsebujejo razlage metod, ki smo jih uporabili v implementaciji. V petem poglavju predstavimo delovanje prototipa za iskanje in sledenje panojev. V šestem poglavju podamo ugotovitve glede hitrosti in natančnosti implementacije. Sedmo poglavje prinaša kritičen pogled na implementacijo in pristop k delu ter ideje za nadaljnje raziskovanje.

Poglavje 2

Opis sistema za zamenjavo reklamnih panojev

Sistem pretvori izvorni video športnega prenosa v novi video z zamenjanimi panoji. Uporabnik mora najprej določiti množico izvornih panojev, ki jih namerava menjati. Uporabnik sistema izvorne panoje na igrišču fotografira in vnese v sistem. Nato za vsak tip izvornega panoja določi novi pano in vnese njegovo sliko v sistem.

Zamenjava panojev pri direktnih prenosih poteka v realnem času. Sistem na vhodu bere slike videa, v njih nadomesti iskane panoje z novimi in jih z majhnim časovnim zamikom zapiše na izhod. Glavne naloge sistema so določanje lokacije panojev v videu, sledenje panojev in zamenjava panojev z novimi.

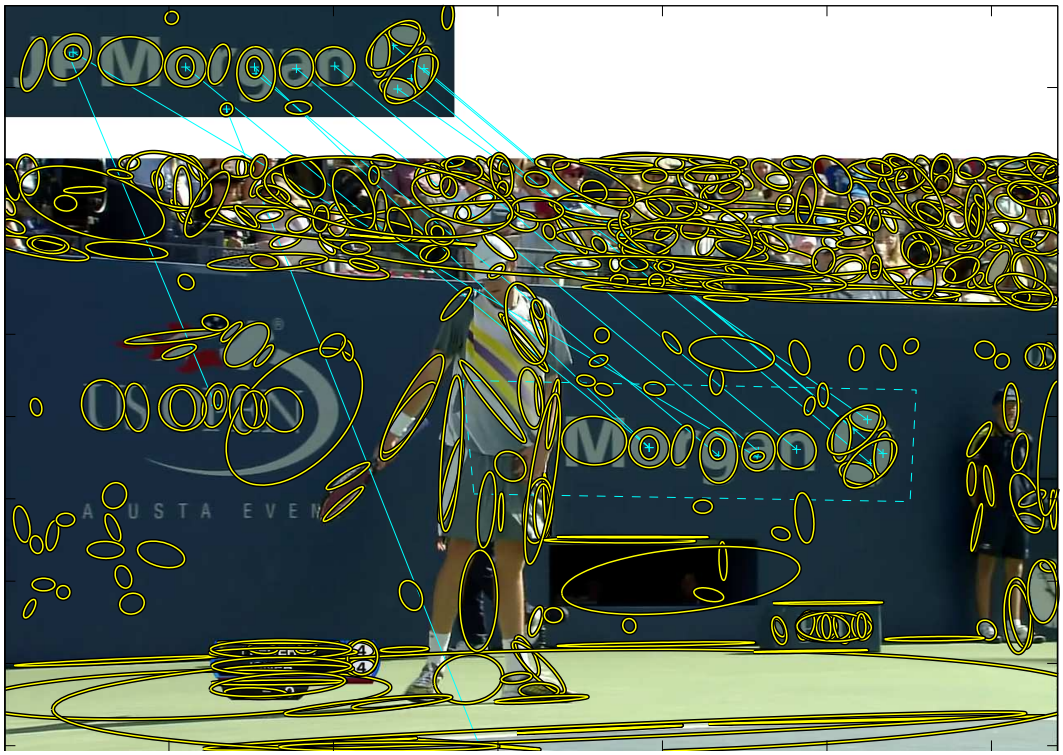
2.1 Določanje lokacije panojev v videu

Cilj komponente za iskanje je natančna določitev lokacije iskanih panojev v videu. Celoten postopek je ponazorjen na sliki 2.1. Na zgornjem delu slike je prikazana slika primera panoja, ki jo je uporabnik predhodno vnesel v sistem, na spodnjem delu pa slika iz videa, na kateri sistem najde podani pano.

Lokacijo določimo z naslednjim postopkom:

- določimo ujemanja med primerom panoja in panojem iz videa,
- na podlagi teh ujemanj izračunamo homografijo med primerom panoja in panojem iz videa,

- z ocenjeno homografijo preslikamo robove določene na primeru panoja v sliko iz videa. Preslikani robovi so na sliki 2.1 ponazorjeni s črtkanimi črtami.



Slika 2.1: Določanje lokacije panoja

Pri iskanju ujemanj med primerom panoja in panojem v videu smo soočeni z dvema problemoma: različna oblika (zaradi drugega zornega kota) in osvetlitev panoja iz primera ter panoja na sliki iz videa.

Problem različnih oblik bomo rešili s kovariantnimi regijami. Te regije so določene na sliki in se spreminjajo skladno s transformacijo slike — z drugimi besedami — obsegajo ista območja predmeta na sliki pri različnih transformacijah slike. Zaradi te lastnosti bomo lahko določili regije na primeru panoja in na sliki iz videa ter z njihovo primerjavo določili ujemanja regij med obema slikama.

Najnovejši pristop pri določanju kovariantnih regij so afino kovariantni detektorji [19]. Ti določijo regije, ki se spreminjajo skladno z afinimi transformacijami. Afina sprememba regije sicer ni točna pri projekcijski transformaciji,

(a) (b) (c) (d)

Slika 2.2: Postopek primerjave dveh regij

ki loči primer panoja in pano v sliki iz videa, a pri majhnih regijah je ta razlika dovolj majhna, da jo lahko zanemarimo.

Slika 2.2 prikazuje, kako afino kovariantne regije rešijo problem različnega zornega kota. Na sliki 2.2(a) je naslovnica knjige slikana z dveh zornih kotov. Na sliki 2.2(b) sta najdeni afino kovariantni regiji, ki obsegata črko 'g'. Regije v obliki elips transformiramo v kroge enakih velikosti. Na sliki 2.2(b) se najdeni regiji razlikujeta za afino transformacijo, na sliki 2.2(c) pa sta ti regiji transformirani v kroga, ki ju loči le še rotacija. Problem rotacije lahko rešimo tako, da regije rotiramo glede na prevladujočo smer gradienta intenzitete. Na sliki 2.2(d) sta prikazani enako usmerjeni regiji, z normaliziranimi vrednostmi slikovnih elementov.

Ko imamo določene afino kovariantne regije na sliki primera panoja in na sliki iz videa, lahko poiščemo ujemanja. Ujemanja določimo s pomočjo opisnikov regije [20]. Naloga opisnika je, da na podlagi slikovnih elementov, ki jih regija obsega, določi vektor, ki nam služi za primerjavo podobnosti vsebine regij. Opisniki lahko temeljijo na lastnostih slike kot so barva, intenziteta, porazdelitev robov, ipd. Dober opisnik je tak, ki bo ujemajočima regijama določil vektorja, med katerima je majhna razdalja, a bo hkrati razločen, tako da bosta različni regiji imeli različna opisa. Ujemanja med sliko primera panoja in sliko iz videa določimo z izračunom razdalj med opisniki regij primera panoja in opisniki regij slike iz videa. Tisti pari regij, katerih opisnika se malo razlikujeta, so izbrani za ujemajoče regije.

Regije, kot jih je določil prototip sistema, so prikazane s približki v obliki elips na sliki 2.1 in določene ujemajoče regije so povezane s črto.

Ko najdemo ujemajoče opisnike, in s tem ujemajoče regije, lahko poiščemo transformacijo. Transformacija se izračuna iz vsaj štirih parov ujemajočih točk. Za ujemajoče točke vzamemo težišča ujemajočih regij in preslikavo

izračunamo z metodo DLT (direct linear transformation) za ocenjevanje homografije [10]. Ker sistem pogosto napačno določi ujemajoče regije, je uporabljena še metoda RANSAC [8], s katero je mogoče najti natančno transformacijo, kljub morebitni prisotnosti napačnih parov.

Ko imamo transformacijo, preslikamo robove panoja s slike primera panoja na sliko iz videa. Tako dobimo območje panoja v videu.

2.2 Sledenje panojem

Ko smo pano našli, je najbolje, da mu sledimo, saj lahko z veliko mero verjetnosti predvidevamo, da se bo pano v naslednji sliki videa pojavil v bližini lokacije iz prejšnje slike. Sledenje je računsko manj zahtevno in tudi bolj zanesljivo.

2.3 Zamenjava izvornih panojev z novimi

V zadnjem koraku v sliki videa na mestu, kjer smo našli izvorne panoje, vstavimo nove panoje. V idealnem primeru, ko je izvorni pano popolnoma viden, je zamenjava enostavna. Vstavimo transformiran nov pano, prilagodimo njegovo osvetlitev bližnji okolici in zgladimo robove. Problem nastane, ko je pano delno zakrit. Taki primeri so zelo pogosti in težje rešljivi, ker zahtevajo veliko natančnost določanja meje med panojem in objektom, ki ga prekriva. Implementacija rešitve je preobsežna in je lahko predmet nadaljnjega dela.

Poglavje 3

Izbira detektorja in opisnika regij

Velik vpliv na kvaliteto sistema predstavlja izbira detektorja in opisnika regij. V poglavju smo opisali različne detektorje in opisnike, opravili primerjalne teste ter izbrali primerna na podlagi njihovih lastnostih in rezultatov testov.

3.1 Predstavitev detektorjev

Pri izbiri detektorjev smo se omejili na detektorje afino kovariantnih regij. Afino kovariantne regije se spreminjajo skladno z afino transformacijo slike.

Preučili smo:

- Harris afin detektor (angl. Harris-Affine),
- Hessov afin detektor (angl. Hessian-Affine),
- detektor maksimalno stabilnih ekstremnih regij (angl. Maximally Stable Extremal Region Detector),
- detektor regij na osnovi robov (angl. Edge-Based Region Detector),
- detektor regij na osnovi ekstrema intenzitete (angl. Intensity Extrema-Based Region Detector).

Harris afin detektor [18] za zanimive točke določi tiste, katerih okolica je podobna kotom. Za določanje take lastnosti uporabimo matriko drugih momentov (prikazano na formuli 3.1), ki predstavlja porazdelitev gradienta v okolici točke $\mathbf{x}$. L_a v formuli predstavlja odvod intenzitete v

smeri a, ki jo izračunamo z gausovim jedrom velikosti σ_D. Odvode nato zgladimo v okolici $\mathbf{x}$ z gausovim jedrom velikosti σ_I. Nato določimo podobnost okolice kotom. Formula 3.2 definira kriterij podobnosti kotu. Izbere se tiste točke, ki po tem kriteriju dosežejo dovolj velik lokalni maksimum. Tako izbrane točke so stabilne pri spremembah osvetlitve.

$$M = \mu(\mathbf{x}, \sigma_I, \sigma_D) = \begin{bmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{bmatrix} = \sigma_D^2 g(\sigma_I) * \begin{bmatrix} I_x^2(\mathbf{x}, \sigma_D) & I_x I_y(\mathbf{x}, \sigma_D) \\ I_x I_y(\mathbf{x}, \sigma_D) & I_y^2(\mathbf{x}, \sigma_D) \end{bmatrix} \tag{3.1}$$

$$c = det(\mu(\mathbf{x}, \sigma_I, \sigma_D)) - \alpha tr(\mu(\mathbf{x}, \sigma_I, \sigma_D)) \tag{3.2}$$

Nato določimo v okolici točke značilno velikost z operatorjem LoG [14].

Za določanje afine oblike v okolici točke se ponovno uporabi matriko drugih momentov. Obliko določimo z lastnima vrednostima matrike drugih momentov, ki predstavljata dve glavni vrednosti spremembe signala v okolici točke. Okolico iterativno transformiramo tako, da sta v transformirani okolici obe lastni vrednosti matrike drugih momentov dovolj enaki. Transformacija je kvadratni koren matrike drugih momentov M. Transformacija je zapisana v formuli 3.3. Zaporedje transformacij nam natančno določi obliko regije.

$$\mathbf{x'} = M^{1/2}\mathbf{x} \tag{3.3}$$

S podatki o lokaciji, velikosti in obliki lahko določimo regijo v obliki elipse.

Hessov afin detektor [18] je podoben Harris afinem, le da v prvi fazi zaznavanja zanimivih točk uporabi Hessovo matriko (formula 3.4). Postopek za določanje merila in oblike regije je enak. Določi regije, ki so svetlejše ali temnejše od svoje okolice.

$$H = H(\mathbf{x}, \sigma_I, \sigma_D) = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} = \begin{bmatrix} I_{xx}(\mathbf{x}, \sigma_D) & I_{xy}(\mathbf{x}, \sigma_D) \\ I_{xy}(\mathbf{x}, \sigma_D) & I_{yy}(\mathbf{x}, \sigma_D) \end{bmatrix} \tag{3.4}$$

Detektor MSER [16] zaznava maksimalno stabilne ekstremne regije. Ekstremne regije so povezane komponente pri določeni vrednosti binarizirane

slike. Omejene so z minimalno ali maksimalno vrednostjo slikovnih elementov, ki jih vsebujejo. Ekstremne regije so stabilne, če ostaja njihova velikost malo spremenjena pri spreminjanju vrednosti, pri kateri se slika binarizira. Tiste regije, pri katerih stabilnost doseže maksimum, so maksimalno stabilne.

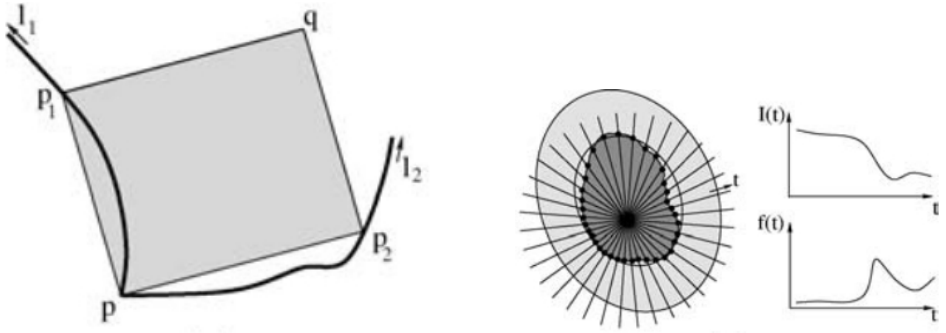
Za algoritem zaznavanja se uporabi algoritem za grajenje drevesa povezanih komponent. V drevesu je ekstremna regija starš drugi ekstremni regiji, če prva drugo direktno vsebuje.

Najdene regije so glede na okolico kontrastne in zato stabilne. Za razliko od prejšnjih dveh detektorjev (Harris afin in Hessov afin) MSER najde vse možne velikosti regij, saj ni omejen z iskanjem po vnaprej določenih merilih.

EBR [24] (edge-based region detector, sl. detektor regij na temelju robov) zaznava regije v obliki paralelogramov. Pri več merilih se z detektorjem Harris poišče kote in v okolici kota oba robova z detektorjem Canny[4]. Kandidati za regijo so množica paralelogramov, ki jih definirajo točka kota in ena točka na vsakem robu. Izmed paralelogramov se izbere takega, ki predstavlja afino kovariantno regijo. Prvi kriterij je razmerje med položajem točk na krivuljah, ki predstavljata rob. Površina med krivuljo in paralelogramom mora biti za obe točki enaka. Drugi kriterij je maksimalna vrednost določene funkcije intenzitete pod paralelogramom. Tako dobimo enega ali več paralelogramov, ki definirajo afino kovariantno regijo. Na sliki 3.1(a) je prikazana EBR regija, p je točka na kotu, p_1 in p_2 točki na robovih ter l_1 in l_2 krivulji, ki predstavljata rob. Slika je vzeta iz članka o primerjavi detektorjev regij[19].

IBR [24] (intensity extrema-based region detector, sl. detektor regij na temelju ekstrema intenzitete) pri različnih merilih zazna točke, v katerih obstaja ekstrem intenzitete. Na ekstremu izberemo polpremice, ki izvirajo v ekstremu in se širijo v vse smeri. Izračunamo vrednosti določene funkcije intenzitete po polpremici. Kjer ima ta funkcija ekstrem, je meja afino kovariantne regije. IBR regija je prikazana na sliki 3.1(b). Ob robu je prikazan graf intenzitete za izbrano črto in funkcije intenzitete v odvisnosti od razdalje od zaznane točke.

Metodo zaznavanja regij smo izbrali na podlagi teoretičnega znanja in na podlagi rezultatov testiranj njihovih algoritmov. Za testiranje smo uporabili enak pristop kot Mikolajczyk in ostali v članku “A Comparison of Affine Region



(a) EBR (b) IBR

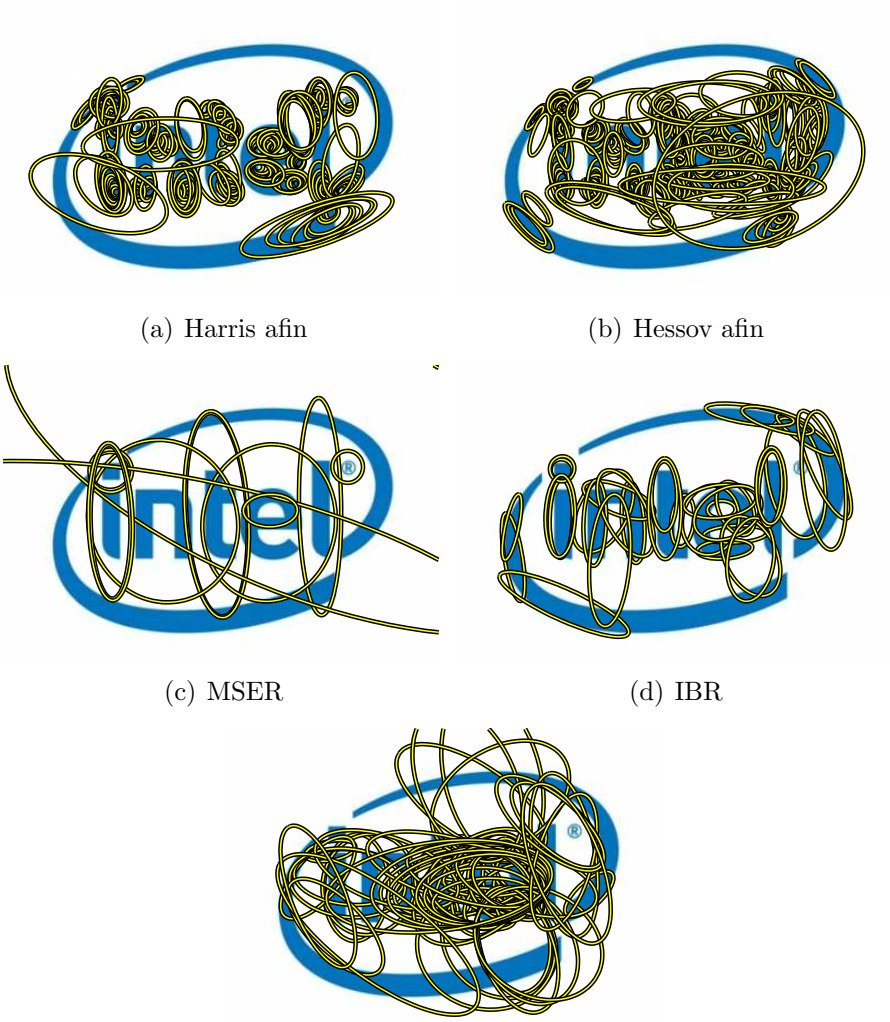
Slika 3.1: Primer EBR in IBR regije. Slika je vzeta iz članka o primerjavi detektorjev regij[19].

Detectors"[19], kjer so opravili primerjavo kvalitete detektorjev ob različnih transformacijah slike. Pri testiranju smo uporabili množico slik, ki predstavljajo reprezentativni vzorec panojev. Programi za izračun regij, izračun opisnikov in testiranje so plod dela omenjenih avtorjev in so dostopni na internetu [26].

3.2 Način testiranja detektorjev in kriteriji izbire detektorja

Opredelimo kriterije, po katerih ocenjujemo kvaliteto detektorjev. Imamo dve sliki istega predmeta, ki se razlikujeta z nam znano transformacijo. Na obeh slikah izračunamo regije in ugotavljamo ujemanje s to transformacijo. Definirajmo bolj natančno: imamo regijo $\mathcal{A}$ z originalne in regijo $\mathcal{B}$ s spremenjene slike. Sliki povezuje transformacija $\mathcal{H}$. Napako prekrivanja določimo z enačbo 3.5. Postopek je prikazan na sliki 3.3, kjer je predmet črka A. Okvir okoli črke predstavlja rob slike.

$$\epsilon = 1 - \frac{\mathcal{H}(\mathcal{A}) \cap \mathcal{B}}{\mathcal{H}(\mathcal{A}) \cup \mathcal{B}} < \epsilon_0 \tag{3.5}$$



(a) Harris afin

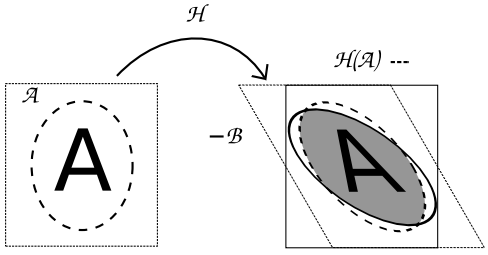
(b) Hessov afin

(c) MSER

(d) IBR

(e) EBR

Slika 3.2: Primerjava regij različnih detektorev



Slika 3.3: Testiranje detektorjev regij

Če je vrednost napake pod konstanto ϵ_0, označimo regiji kot *ujemajoči*. V testih ocenjujemo kvaliteto detektorjev po naslednjih kriterijih:

1. **ponovljivost**: to je število ujemajočih regij, deljeno s številom regij v skupnem območju na originalni ali spremenjeni sliki (upošteva se tisto z manj regijami). Iz stopnje ponovljivosti sklepamo, kolikšen delež regij iz primera panoja bo imel ujemajočo regijo v sliki iz videa. Visoka stopnja ponovljivosti, nam poleg pravilnejšega lociranja panoja, poveča hitrost iskanja homografije z RANSAC, kar poveča delež pravilnih ujemanj.
2. **Število ujemajočih regij**: Samo število regij je pomembno; če je regij premalo, lahko v koraku iskanja homografije dobimo slab približek, oziroma sploh ne moremo izračunati homografije.
3. **hitrost implementacije detektorja**: sistem potrebuje procesiranje videa v realnem času, zato je hitrost zelo pomembna.

Omeniti velja še nekaj podrobnosti glede testiranja. Metodi Harris afin in Hessov afin določita za regije elipse, metoda MSER različne oblike, ki so odvisne od intenzitete, metodi IBR in EBR pa določita regijo v obliki paralelograma. V testu se regijam, ki niso v obliki elipse, določi približek v obliki elipse. Poleg tega se velikost regij normalizira na konstantno velikost, saj bi sicer kriterij favoriziral velike regije, ker je pri velikih regijah delež prekrivanja večji.

3.3 Testna množica

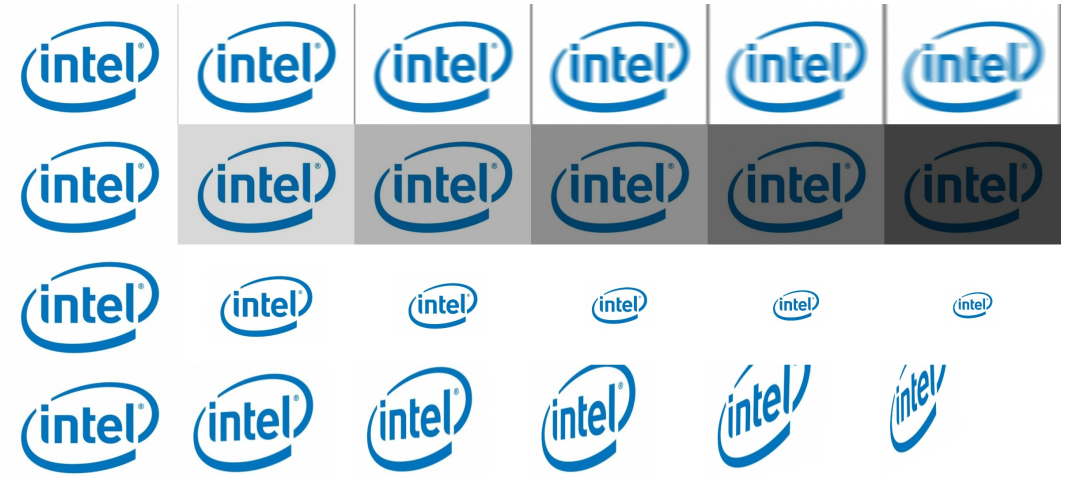
Testno množico sestavljajo slike različnih logotipov (slika 3.4). Izbrali smo logotipe, ki vsebujejo tako oblike kot črke.

Za vsako izmed testnih slik smo generirali 5 spremenjenih slik, razvrščenih po velikosti spremembe. Spremembe so naslednje:

- zameglitev zaradi gibanja v horizontalni smeri za 1%, 2%, 3%, 4% in 5% slike,
- zmanjšanje intenzitete na 0.85, 0.7, 0.55, 0.4 in 0.25 njihove prejšnje vrednosti,
- zmanjšanje slik na 2/3, 1/2, 2/5, 1/3 in 2/7 prejšnje velikosti,
- 5 projekcijskih transformacij - od najmanjše spremembe do največje.

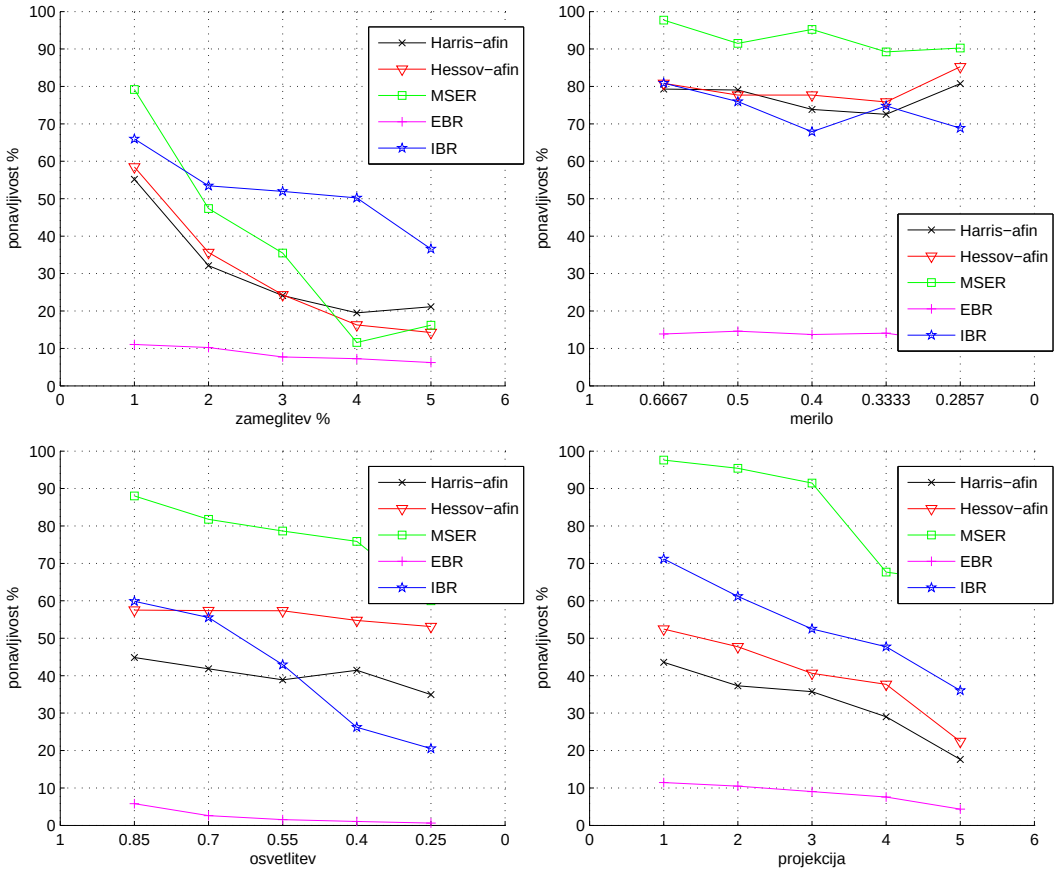
Originalnim in transformiranim slikam smo dodali gausov šum s srednjo vrednostjo 0 in varianco 0,01.Primer spremembe enega izmed logotipov je prikazan

Slika 3.4: Primeri logotipov uporabljenih za testiranje detektorjev regij



Slika 3.5: V testu uporabljene transformacije ene izmed testnih slik

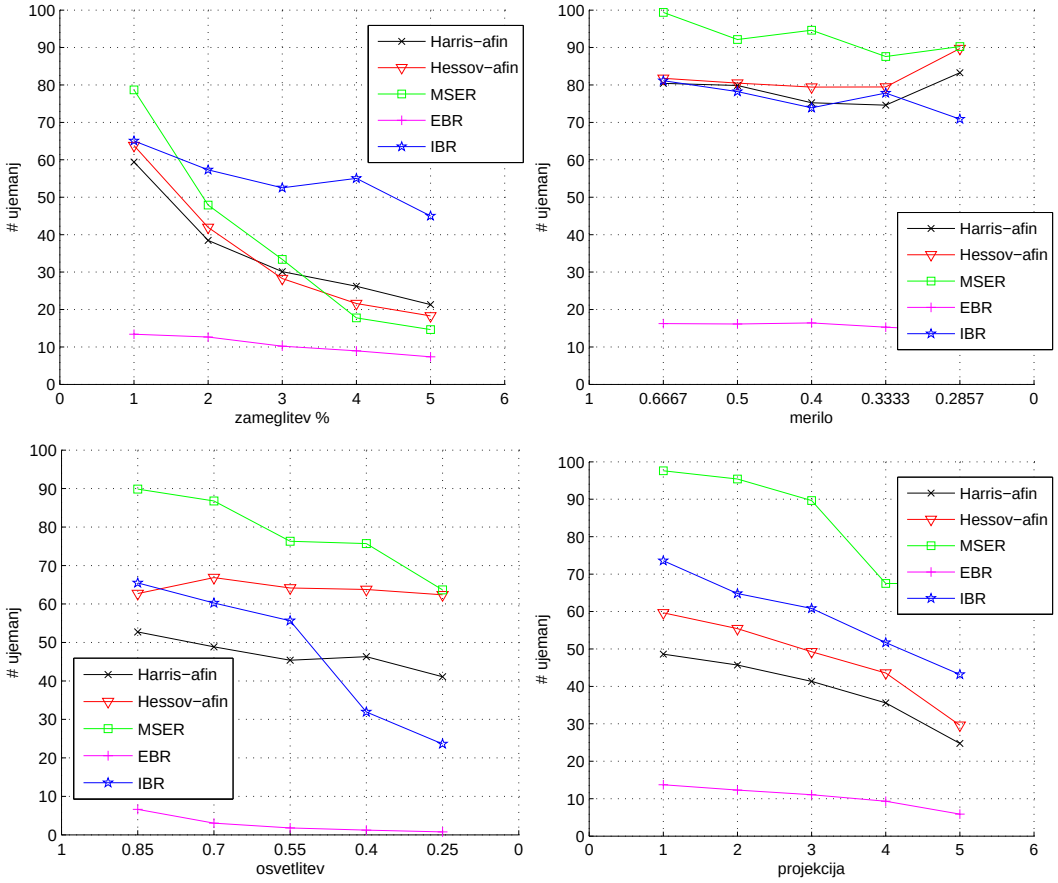
na sliki 3.5.



Slika 3.6: **Ponovljivost** pri različnih spremembah slike glede na izbrani detektor. Parameter ϵ_0 je nastavljen na 40%. Graf prikazuje srednjo vrednost ponovljivosti v odvisnosti od velikosti spremembe slike.

	čas izvajanja (s)
Harris afin	0.86
Hessov afin	0.36
MSER	0.08
IBR	1.86
EBR	3.34

Tabela 3.1: Hitrosti detektorjev na sliki logotipa Intel, velikosti 800x600



Slika 3.7: **Število ujemajočih regij** pri različnih spremembah slike glede na izbrani detektor. Parameter ϵ_0 je nastavljen na 40%. Graf prikazuje srednjo vrednost števila ujemajočih regij v odvisnosti od stopnje spremembe slike.

3.4 Predstavitev opisnikov

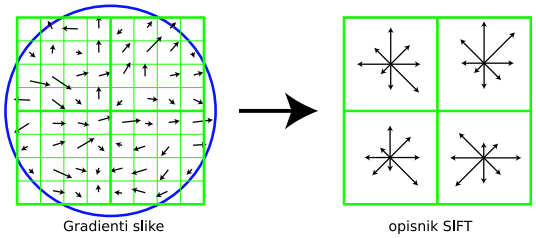
Opisnik regije služi hitremu primerjanju regij med sabo. Iz dela slike, ki jo pokriva regija, se glede na tip opisnika izračuna vektor, ki regijo opiše. Podobnost dveh regij se nato izračuna z razdaljo med njunima vektorjema. Izbirali smo med naslednjimi opisniki in metodami primerjanja regij:

- SIFT,
- Shape context,
- histogram lokacije in orientacije (angl. Gradient location-orientation histogram),
- PCA-SIFT,
- Spin image,
- usmerjeni filtri (angl. Steerable Filters),
- kompleksni filtri (angl. Complex Filters).
- invariantni momenti (angl. Moment invariants),
- križna korelacija (angl. Cross correlation).

Opisali bomo nekatere izmed njih, opredelili kriterije kvalitete opisnikov in opravili teste.

Opisnik SIFT [15] regijo opiše s histogramom lokacije in orientacije gradienta. Regijo v obliki elipse transformiramo v krog določene velikosti ter ga rotiramo glede na prevladujočo smer orientacije gradienta v regiji. V točkah izračunamo gradient in ga utežimo z gausovo funkcijo, s σ velikosti polovice radija kroga, kjer imajo največjo težo točke v sredini kroga. Tako je opis regij manj občutljiv na napake pri določanju regije.

Nato z mrežo razdelimo regijo na območja. Na območjih izračunamo histogram smeri gradienta, kjer ponovno bolj upoštevamo točke v bližini centra območja. Histogram ima 8 razredov orientacije gradienta. Postopek je prikazan na sliki 3.8. Na levi je prikazan gradient v točkah in delitev na območja, na desni pa histogrami orientacij gradienta za posamezna območja. Histograme vseh območji lahko predstavimo v vektorju dolžine $8n$, kjer je n število območij. Vektor normaliziramo na dolžino 1. S tem je opisnik odporen na spremembe v kontrastu.



Slika 3.8: Opisnik SIFT

Opisnik GLOH (Gradient location-orientation histogram) ima podobno zasnovo kot opisnik SIFT. Regijo razdelimo z mrežo v logaritmično-polarnih koordinatah, s tremi območji v smeri oddaljenosti od izhodišča in 8 območij glede na kot. Območje najbližje izhodišču ni deljeno. Tako dobimo 17 območij na katerih izračunamo histograme gradienta. Histogrami imajo 16 razredov smeri orientacije gradienta. Tako dobimo vektor dimenzije 272. Dimenzijo opisnika nato zmanjšamo na 128 z analizo glavnih komponent [12].

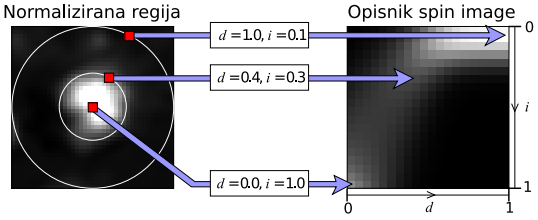
Opisnik Shape Context temelji na metodi primerjanja oblik Shape Context [2]. Predstavlja ga histogram lokacij točk na robovih in orientacije roba v točkah. Površina regije se razdeli na 9 območij določenih z mrežo v logaritmično polarnih koordinatah. Razdalja od izhodišča razdeli v 3 območja in kot na 4 območja, kjer so 4 območja najbližja izhodišču združena v eno. Vpliv točke na histogram je utežen z velikostjo gradienta v tej točki. Tak pristop favorizira bolj izrazite robove in dalje boljše rezultate.

PCA-SIFT [11] je opisnik gradienta v smeri x in y izračunan na točkah mreže 39x39. Dobljen vektor dolžine 3.042 se reducira z analizo glavnih komponent[12] na vektor dolžine 36.

Opisnik spin image je histogram lokacije in intenzitete. Intenziteta je razdeljena v 10 razredov, lokacija pa je razdeljena na pet kolobarjev s centrom v središču regije. Na sliki 3.9 je prikazana normalizirana regija na levi in njen histogram na desni.

3.5 Kriteriji izbire opisnika

Dobra izbira opisnika je tista, za katero bosta ujemajoči regiji imeli tudi podobna opisnika. Vzemimo za ujemajoča opisnika tista, ki sta si najbližja po



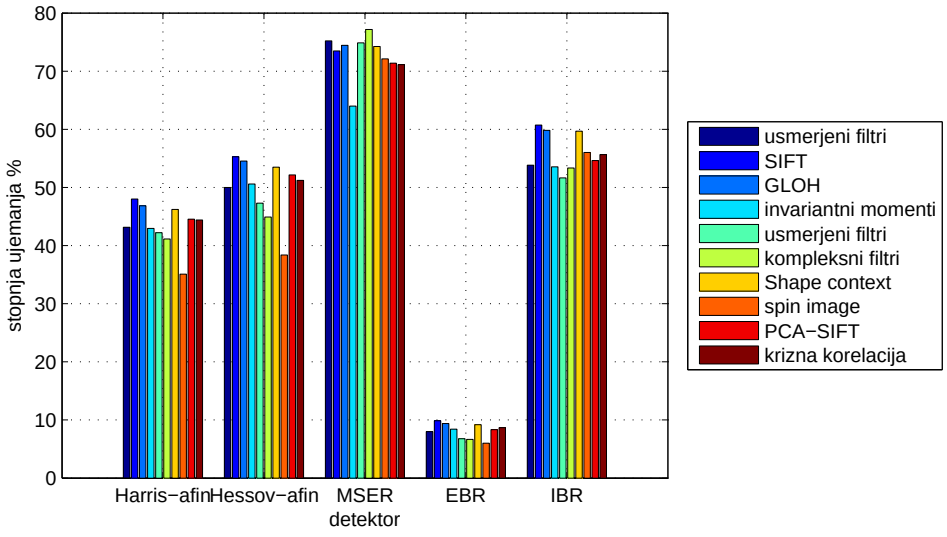
Slika 3.9: Opisnik spin image

evklidski razdalji. Če imata ujemajoči regiji ujemajoča opisnika, imamo pravilno ujemanje.

V testih bomo uporabili naslednji kriterij:

Stopnja pravilnega ujemanja - število pravilnih ujemanj deljeno s številom regij v skupnem območju na originalni ali spremenjeni sliki (izbrana je slika z manj regijami).

Testiranje je opravljeno na enakem naboru slik in sprememb slik kot pri testu detektorjev. Ker je kvaliteta opisnika odvisna od regije, ki jo opisuje, je v testu prikazana stopnja ujemanja opisnikov glede na tip detektorja regij. Na sliki 3.10 so rezultati povprečne stopnje ujemanja za srednjo težavnost spremembe (stolpec 4 v sliki 3.5).



Slika 3.10: Stopnja pravilnega ujemanja glede na izbiro detektorja in opisnika

3.6 Sklep

Iz grafov na sliki 3.6 in 3.7 je razvidno, da se je detektor MSER po obeh kriterijih obnesel najbolje pri spremembi osvetlitve, pri spremembi merila in pri projekcijski transformaciji. Slabo se je obnesel pri zameglitvi. To je bilo pričakovano, ker v spremenjeni sliki ni bilo več kontrastnih robov. Regije so zato postale manj stabilne, lahko je prišlo tudi do združevanja regij.

Po hitrosti implementacije je najboljši detektor MSER. Čas izvajanja je vsaj 5x krajši kot pri ostalih algoritmih.

Iz grafa na sliki 3.10, se vidi, kako dobro se obnesejo kombinacije detektor-opisnik na izbranih primerih. Najbolje so se obnesle kombinacije z MSER detektorjem. Te kombinacije so imele za okoli 20% boljši rezultat.

V prototipu smo uporabili detektor MSER in opisnik Shape Context. Slednji sicer ni imel najboljšega rezultata, a je za najboljšim opisnikom zaostajal le za 3%. Izbrali smo ga, ker je primeren prav za opis enostavnih oblik in črk[1, 2].

Poglavje 4

Teoretične osnove

V poglavju so podrobneje opisane metode uporabljene pri določanju lokacije in sledenju panoja. Opisali smo *maksimalno stabilne ekstremne regije*, zapisali njihovo definicijo in lastnosti. Opisali smo postopek grajenja *drevesa ekstremnih regij*, kriterij stabilnosti in *določanje maksimalno stabilnih regij* v drevesu. Predstavili smo metodo za primerjanje oblik Shape Context in *opisnik Shape Context*, ki na njej temelji. Opisali smo naš pristop pri določanju ujemajočih regij na podlagi njihovih opisnikov. Nato smo omenili metodo DLT, ki iz parov ujemajočih regij določi homografijo med primerom panoja in panojem v videu. Opisali smo še metodo RANSAC, ki pripomore k bolj robustni oceni homografije.

4.1 MSER

Neformalni opis regij MSER lahko podamo s primerom. Denimo, da imamo črno-belo sliko, ki jo binariziramo pri vseh vrednostih intenzitete $\theta = \{0..255\}$. Deli slike pod vrednostjo θ bodo črni, ostali pa beli. Če začnemo pri vrednosti $\theta = 0$, imamo celotno binarizirano sliko bele barve. Ko se vrednost parametra θ veča, se začnejo na binarizirani sliki pojavljati črni deli. Črne dele imenujemo ekstremne regije. Tiste ekstremne regije, ki malo spreminjajo obliko, kljub spremembi parametra θ, so lahko maksimalno stabilne ekstremne regije. Te regije imajo dobre lastnosti, ki jih lahko uporabimo pri iskanju ujemajočih delov primerov panojev in izvornih panojev v slikah videa.

Podrobnejša definicija slike, regij, ekstremnih regij in maksimalno stabilnih regij je zapisana v tabeli 4.1. Pojmi so definirani enako, kot jih definirajo Matas in ostali v originalnem članku o MSER [16].

Slika: Slika I je preslikava $I : \mathcal{D} \subset \mathbb{Z}^2 \rightarrow \mathcal{S}$. V implementaciji uporabljamo črno-bele slike, torej je $\mathcal{S}$ urejena množica $\mathcal{S} = \{0..255\}$. Definirana je relacija sosed $A \subset \mathcal{D} \times \mathcal{D}$, kjer sta $p, q \in \mathcal{D}$ v relaciji pAq, ko in natanko ko $\sum_{i=1}^{d} |p_i - q_i| \leq 1$.

Regija: Regija $\mathcal{Q}$ je povezana podmnožica $\mathcal{D}$, torej za vsak $p, q \in \mathcal{Q}$ obstaja zaporedje $p, a_1, a_2, .., a_n, q$, za katere velja $pAa_1, a_1Aa_2, .., a_{n-1}Aa_n, a_nAq$.

Zunanja meja regije $\partial\mathcal{Q} = \{q \in \mathcal{D} \setminus \mathcal{Q} : \exists p \in \mathcal{Q} : qAp\}$, torej je zunanja meja $\partial\mathcal{Q}$ od $\mathcal{Q}$ množica slikovnih elementov, ki so sosednji vsaj enemu slikovnemu elementu iz $\mathcal{Q}$, a niso v $\mathcal{Q}$.

Ekstremna regija $\mathcal{Q} \subset \mathcal{D}$ je regija, za katero velja za vsak $p \in \mathcal{Q}, q \in \partial\mathcal{Q} : I(p) > I(q)$ (omejena z nižjimi vrednosti slikovnih elementov) ali $I(p) < I(q)$ (omejena z višjimi vrednostmi slikovnih elementov).

Maksimalno stabilna ekstremna regija. Naj bo $\mathcal{Q}_1, .., \mathcal{Q}_{i-1}, \mathcal{Q}_i, ..$ zaporedje gnezdenih regij ($\mathcal{Q}_i \subset \mathcal{Q}_{i+1}$). Ekstremna regija $\mathcal{Q}_{i^*}$ je maksimalno stabilna, če in samo če $q(i) =\mid \mathcal{Q}_{i+\Delta} \setminus \mathcal{Q}_{i-\Delta} \mid / \mid \mathcal{Q}_i \mid$ ima lokalni minimum pri i^* ($\mid . \mid$ pomeni moč množice). $\Delta \in \mathcal{S}$ je parameter metode.

Tabela 4.1: Definicije pojmov o MSER

4.1.1 Dobre lastnosti maksimalno stabilnih ekstremnih regij

Regije se ohranjajo pri enakomerni spremembi osvetlitve. Če se intenziteta poveča za neko vrednost, to ne bo vplivalo na stabilnost regij, torej bodo po spremembi iste regije maksimalno stabilne. Poleg tega algoritem zazna regije ne glede na njihovo velikost. Gradnja drevesa ekstremnih regij poteka od najmanjših regij velikosti enega slikovnega elementa do največje, ki obsega celotno sliko. To je prednost pred metodami, ki so omejene na določene velikosti zaznanih regij, denimo Harris-afin ali Hessov-afin detektor, kjer poteka detekcija na določenih merilih.

Implementacija detektorja regij MSER je hitra, saj za določanje regij obstaja hiter, skoraj linearen algoritem ($\mathcal{O}(n \log \log n)$, kjer je n število slikovnih elementov). V primerjavi z ostalimi detektorji je bil detektor MSER vsaj 10x hitrejši.

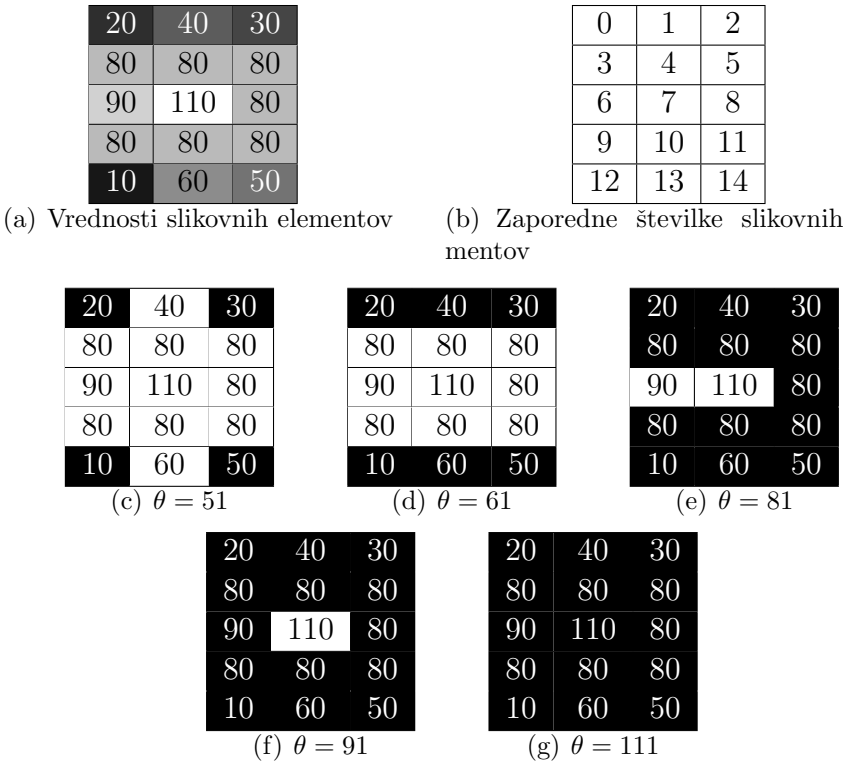
Regije so še posebej primerne za obravnavan problem, ker se v oglaševanju uporabljajo preproste oblike in črke. Navadno se uporabljata dve barvi, ki sta kontrastni. V tem primeru bodo najdene regije na panoju zelo stabilne in s tem odporne na spremembe v osvetlitvi. Poleg tega bodo ponavadi obsegale, v primerjavi z drugimi pristopi iskanja kovariantnih regij, večja območja, kot so na primer cele črke ali deli logotipa. To je vidno na sliki 3.2, ker so primerjani približki regij različnih detektorjev. Na slikah primerov panojev bo malo regij, kar bo olajšalo primerjanje. Regije bodo nudile tudi zadosti informacije za računanje opisnikov.

4.1.2 Drevo ekstremnih regij

Regije MSER se določi tako, da se zgradi drevo ekstremnih regij slike. Ekstremne regije so v drevesu razporejene tako, da je ekstremna regija $\mathcal{P}$ v drevesu starš ekstremne regije $\mathcal{Q}$, če $\mathcal{P}$ direktno vsebuje $\mathcal{Q}$, torej ne obstaja kakšna druga ekstremna regija, ki bi vsebovala $\mathcal{Q}$ bila vsebovana v $\mathcal{P}$.

Razporeditev ekstremnih regij v drevesu je ponazorjena na slikah 4.1 in 4.2. Slike, na kateri določimo ekstremne regije, je prikazana z vrednostmi njenih slikovnih elementov na sliki 4.1(a). Vse njene ekstremne regije so za različne vrednosti θ prikazane na slikah 4.1(c)-(g) in njihov razpored v drevesu ekstremnih regij na sliki 4.2. V drevesu so ekstremne regije označene z zaporednimi številkami slikovnih elementov, ki regiji pripadajo.

Za grajenje lahko uporabimo algoritem za grajenje drevesa komponent [21]. Za opis algoritma je potrebno definirati pojem povezane komponente. Pove-

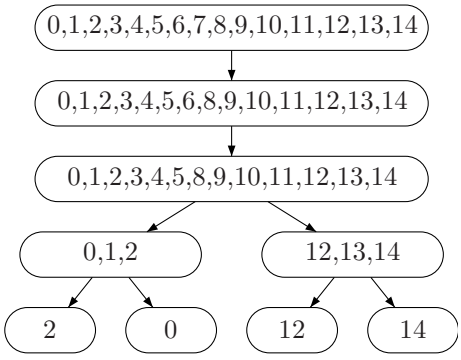


(a) Vrednosti slikovnih elementov (b) Zaporedne številke slikovnih elementov

(c) $\theta = 51$ (d) $\theta = 61$ (e) $\theta = 81$

(f) $\theta = 91$ (g) $\theta = 111$

Slika 4.1: Prikaz ekstremnih regij



Slika 4.2: Drevo ekstremnih regij

zana komponenta je v našem primeru del slike, ki obsega povezane (preko relacije sosed) slikovne elemente iste barve. Povezane komponente predstavljajo vozlišča drevesa. Ekstremna regija omejena z maksimalno vrednostjo intenzitete θ, vsebuje povezane komponente, ki vsebujejo slikovne elemente vrednosti od 0 do $\theta - 1$.

Drevo gradimo od spodaj navzgor tako, da obiskujemo slikovne elemente slike v zaporedju po njihovi intenziteti. V obravnavanem primeru so uredijo po padajočem vrstnem redu. Primeren algoritem je "sortiranje s štetjem" z zahtevnostjo $\mathcal{O}(n)$.

Algoritem se uporablja za sortiranje števil v primerih, ko imamo omejeno število različnih vrednosti števil v seznamu. Algoritem deluje tako, da ugotavlja, kolikokrat se posamezna vrednost pojavlja v množici števil. S tem določimo lokacije posameznih vrednosti v urejenem seznamu.

V primeru implementacije ne gre za sortiranje števil, ampak urejanje indeksov slikovnih elementov po njihovi vrednosti. V prvem prehodu čez slikovne elemente preštejemo slikovne elemente vseh vrednosti. Nato izračunamo lokacije slikovnih elementov določene vrednosti v urejenem seznamu. V primeru, ko razporejamo slikovne elemente po naraščajočem vrstnem redu, bodo slikovni elementi, ki imajo vrednost 0 na začetku seznama, tisti z vrednostjo 1 se začnejo pri lokaciji, ki je za eno večja od števila slikovnih elementov vrednosti 0 in tako dalje.

Pri drugem prehodu čez slikovne elemente vpisujemo indekse v urejen seznam. Vsakič, ko dodamo indeks v seznam, povečamo naslov slikovnih elementov te vrednosti v urejenem seznamu za 1.

Ko imamo urejen seznam indeksov slikovnih elementov, začnemo graditi drevo ekstremnih regij omejenih z maksimalno intenziteto.

V prvem koraku vsem slikovnim elementom določimo, da so svoja povezana komponenta in svoje drevo.

Za vsak slikovni element vemo, kateremu delnemu drevesu in vozlišču (povezani komponenti) trenutno pripada. V opisu bomo vozlišče v korenu drevesa, kateremu pripada slikovni element x označili kot $root(x)$. Podrobnosti o hitri implementaciji so opisane v članku o drevesu povezanih komponent [21].

V vsakem naslednjem koraku obiščemo slikovni element p in za vsak že obiskani sosednji slikovni element q (torej takega, ki ima večjo vrednost) naredimo naslednje:

1. pogledamo, če sta p in q v istem drevesu, če sta, gremo na naslednji sosednji slikovni element, sicer:

2. pogledamo, če predstavljata $root(p)$ in $root(q)$ povezano komponento z

isto vrednostjo slikovnih elementov, ju združimo v isto vozlišče, sicer:

3. predstavljata $root(p)$ in $root(q)$ povezani komponenti, kjer ima prva nižje vrednosti slikovnih elementov. $root(q)$ postane sin $root(p)$.

Ko algoritem obišče vse slikovne elemente, se konča. Dobimo drevo povezanih komponent, ki pa je popolnoma enako kot drevo ekstremnih regij, saj poddrevo, ki ima za koren povezano komponento z vrednostjo slikovnih elementov t, predstavlja ekstremno regijo, ki je omejena z vrednostjo $t+1$ in vsebuje vse povezane komponente poddrevesa.

Ko gradimo drevo, za potrebe našega algoritma, računamo še lastnosti regij. Te lastnosti so parametri elipse, ki je približek regije, višina in širina pravokotnika, ki uokvirja regijo in povprečna vrednost slikovnih elementov regije. Vsi ti parametri se določijo za ekstremne regije velikosti enega slikovnega elementa in se spreminjajo, ko se povezane komponente in drevesa združujejo. Tako zagotovimo, da je izračun lastnosti regij enako računsko zahteven, kot je gradnja samega drevesa povezanih komponent.

Gradnja drevesa je na primeru ponazorjena v dodatku.

4.1.3 Določanje maksimalno stabilnih regij

Ko imamo zgrajeno drevo ekstremnih regij, moramo določiti, katere izmed ekstremnih regij so maksimalno stabilne. Pogoj za maksimalno stabilnost smo omenili v tabeli 4.1. Za vsako ekstremno regijo $\mathcal{Q}_i$ moramo poznati velikosti njej ustreznih regij $\mathcal{Q}_{i+\Delta}$ in $\mathcal{Q}_{i-\Delta}$. Regija $\mathcal{Q}_{i+\Delta}$ je v najmanjša ekstremna regija, ki vsebuje ekstremno regijo $\mathcal{Q}_i$ in vsebuje slikovne elemente vrednosti, ki je večja za vsaj Δ od največje vrednosti slikovnih elementov regije $\mathcal{Q}_i$ (enačba 4.2).

$$I_{max}(\mathcal{R}) = argmax\{I(p) : p \in \mathcal{R}\} \tag{4.1}$$

$$\mathcal{Q}_{i+\Delta} = argmax\{|\mathcal{R}| : \mathcal{R} \supset \mathcal{Q}_i, I_{max}(\mathcal{R}) \geq I_{max}(\mathcal{Q}_i) + \Delta\} \tag{4.2}$$

Regija $\mathcal{Q}_{i-\Delta}$ pa je največja ekstremna regija, ki je vsebovana v ekstremni regiji $\mathcal{Q}_i$ in $\mathcal{Q}_i$ vsebuje slikovne elemente vrednosti vsaj za Δ večje vrednosti od slikovnih elementov regije $\mathcal{Q}_{i-\Delta}$ (enačba 4.3).

$$\mathcal{Q}_{i-\Delta} = argmax\{|\mathcal{R}| : \mathcal{R} \subset \mathcal{Q}_i, I_{max}(\mathcal{R}) \leq I_{max}(\mathcal{Q}_i) - \Delta\} \tag{4.3}$$

Ti regiji z algoritmom najdemo tako, da obiščemo vse ekstremne regije in se pri vsaki sprehodimo po drevesu navzgor. Denimo, da smo na regiji $\mathcal{R}_0$, starš

je $\mathcal{R}_1$, starš $\mathcal{R}_1$ je $\mathcal{R}_2$ in tako dalje. Če je regija $\mathcal{R}_i = \mathcal{Q}_i$, potem je enačba 4.4 pogoj, da je regija $\mathcal{R}_0$ njen $\mathcal{Q}_{i-\Delta}$:

$$I_{max}(\mathcal{R}_0) \leq I_{max}(\mathcal{R}_i) - \Delta < I_{max}(\mathcal{R}_1) \tag{4.4}$$

Prva neenačba nam zagotovi, da je med $\mathcal{R}_0$ in $\mathcal{R}_i$ razlika vsaj enaka Δ, druga pa, da ne obstaja druga regija na poti, ki bi po vrednosti bila oddaljena za več ali enako Δ, a manj kot $\mathcal{R}_0$. Poleg tega ima lahko regija $\mathcal{R}_i$ več potomcev, ki zadostijo enačbi 4.4. Izbrati moramo tistega med njimi, ki ima največjo površino.

Podobno velja za $\mathcal{Q}_{i+\Delta}$. Iščemo $\mathcal{Q}_{i+\Delta}$ za $\mathcal{R}_0 = \mathcal{Q}_i$. Zadostni pogoj za to je enačba 4.5

$$I_{max}(\mathcal{R}_i) \leq I_{max}(\mathcal{R}_0) + \Delta < I_{max}(\mathcal{R}_{i+1}) \tag{4.5}$$

Za vsako regijo izračunamo stabilnost po forumi iz tabele 4.1.

Vse ekstremne regije, ki so lokalni minimum, so tudi MSER. V začetku označimo vse ekstremne regije kot MSER. Za vsako ekstremno regijo $\mathcal{R}_0$ pogledamo, če je starš $\mathcal{R}_1$ v grafu bolj stabilen, če je, $\mathcal{R}_0$ ni MSER, če pa je starš manj stabilen, $\mathcal{R}_1$ ni MSER.

Lokalni minimumi so v grafu lahko zelo pogosti in regije se lahko prekrivajo. Izmed dveh preveč podobnih regij je smotrno odstraniti manjšo. Poleg tega odstranimo tudi vse premajhne in prevelike regije. Majhne regije so premalo opisne, prevelike regije so v našem primeru deli igralne površine ali ozadje panojev, kar ne pripomore k iskanju posameznih panojev. Slika 4.3 predstavlja s prototipom najdene približke regije na primeru panoja.

Slika 4.3: Približki regij MSER na primeru panoja

4.2 Opisnik Shape Context

Namen opisnika je, da za dano regijo dobimo invarianten opis, ki je uporabljen pri iskanju ujemajočih regij. Pri implementaciji smo uporabili opisnik Shape

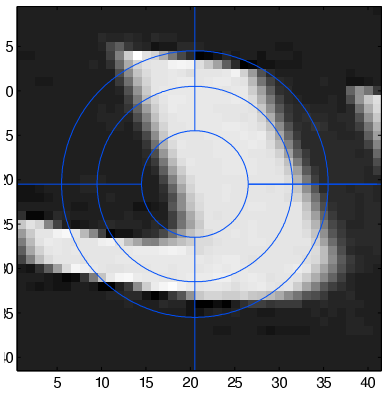
context, ki je nastal na podlagi metode Shape context [2] za primerjanje oblik objektov.

Na objekt gledamo kot na množico točk, na robovih silhuete objekta. Te dobimo tako, da na sliki najdemo robove in po njih enakomerno razporedimo točke. Nato izberemo iz te množice eno točko. Položaj drugih do nje nam opiše obliko objekta. S tem dobimo natančen opis oblike. Opis ni ustrezen za primerjanje oblik, zato relativne položaje točk opišemo s histogramom. Relativen položaj ostalih točk se razporedi v omejeno število območij, ki so enakomerno razporejena v logaritmično polarnih koordinatah z izhodiščem v dani točki. Torej je večja natančnost razporejanja v bližini dane točke. Območja ustvarijo mrežo čez objekt. Če želimo, da je opisnik invarianten na rotacijo, nastavimo orientacijo mreže po tangenti na krivuljo silhuete v tisti točki. Nato za vsako točko izračunamo histogram porazdelitve množice robnih točk po prostoru glede na izbrano točko. Definiramo še ceno ujemanja dveh histogramov z enačbo 4.6.

$$C_{ij} \equiv C(p_i, q_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)} \tag{4.6}$$

Denimo, da imamo dve obliki, za obe izračunane histograme vseh točk, nato moramo točke prve oblike povezati z drugo tako, da bo skupna cena povezav minimalna. Če je točk ene izmed oblik manj, ji dodamo navidezne točke s konstantno ceno povezave. Nato poiščemo optimalno dodelitev točk ene oblike točkam druge oblike. Cena vseh povezav nam pove stopnjo razlikovanja obeh oblik.

Deskriptor regij se razlikuje od originalne ideje.



Slika 4.4: Normalizirana regija z območji opisnika shape context

Najprej moramo določiti regijo, na kateri bomo računali opisnik. Lahko

bi vzeli kar približek regije v obliki elipse, vendar bi tako lahko izgubili pomembne oblike, ki so prisotne ravno na robovih regije MSER. Primerneje je vzeti nekajkrat večjo elipso. Povečano regijo v obliki elipse transformiramo v krog in ga normaliziramo na velikost, ki je primerna za opisnik. Preden ga zmanjšamo, ga zgladimo z Gaussovim filtrom z jedrom velikosti trenutna regija deljeno z normalizirana regija. Da bo opisnik invarianten na rotacijo, zmanjšano okroglo regijo zavrtimo glede na prevladujočo smer orientacije gradienta, izračunanega v osrednjem delu kroga. Deskriptor bo, za razliko od originalne ideje, vseboval še informacijo o gradientu. Lokacijo bomo določili z devetimi območji in gradient s štirimi, torej imamo 36 kategorij gradienta in lokacij. Krog se razdeli v 9 območij v logaritemsko polarnih koordinatah - površino razdelimo na 3 kose glede na razdaljo od centra in zunanja 2 še na 4 kose glede na kot. Slika 4.4 predstavlja razdelitev regije. Za smer gradienta določimo 4 orientacije (horizontalno, vertikalno in po diagonalah). Na regiji se uporabi detektor robov Canny[4]. Vsaka točka pripada eni izmed 36 kategorij. Za vsako kategorijo se izračuna vsota velikosti gradienta v točkah, ki ji pripadajo.

4.3 Ujemanje opisnikov

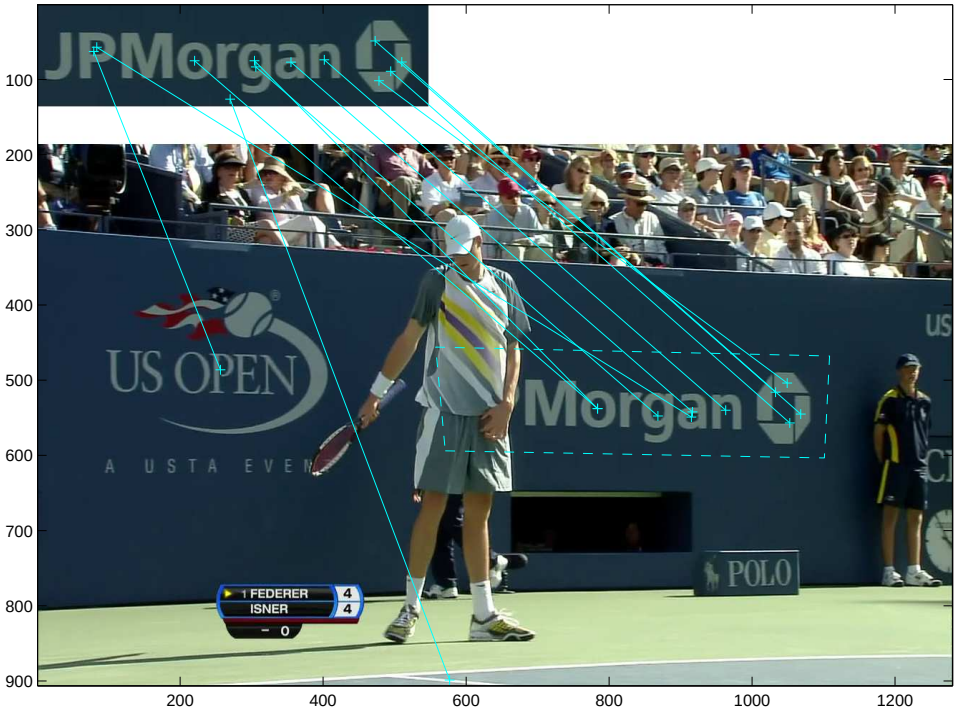
V naslednjem koraku skušamo določiti čim več parov ujemajočih regij, torej tiste, ki obsegata isti del panoja na sliki primera panoja in isti del izvornega panoja z videa. Treba je določiti kriterij, po katerem bomo na podlagi izračunanih opisnikov določili pare.

V implementaciji določamo pare tako, da za vsako regijo na sliki primera panoja izračunamo evklidske razdalje med njenim opisnikom in opisniki vseh regij iz slike videa. Med računanjem se hranita dve najmanjši razdalji. Izberemo tiste pare, pri katerih je najmanjša razdalja za določen faktor manjša od druge najmanjše. Ta pogoj dobro omeji izbiro napačnih parov, vendar ni najustreznejši, ko je v videu prisotnih več enakih panojev.

Drugi kriterij za izbiro para je dovolj majhna razdalja med opisnikoma, ne glede na to ali sta najbližja. Meja razdalje je postavljena na nekajkrat povečano povprečno razdaljo med opisniki že izbranih parov. Tako rešimo problem enakih panojev v videu.

4.4 Ocenjevanje homografije in metoda RANSAC

Naslednji korak je določanje območja panoja na sliki iz videa. Območje določimo tako, da ocenimo homografijo med primerom panoja in sliko panoja iz videa. Homografija je linearna projekcijska transformacija, ki slika točke $x \in \mathbb{R}^2$ v točke $x' \in \mathbb{R}^2$. Homografijo smo iz parov ujemajočih točk ocenili z metodo DLT[10] na normaliziranih točkah x in x'. Za določitev homografije potrebujemo najmanj štiri pare točk, več točk nam bolje oceni homografijo. Na sliki 4.5 so prikazane ujemajoče točke, kot jih določi prototip. Okvir okoli panoja v videu so s homografijo preslikane meje slike primera panoja.



Slika 4.5: Ujemajoče točke in najdena homografija

Homografije pa ne moremo direktno izračunati iz točk, ki smo jih določili za ujemajoče, ker obstaja verjetnost, da so med njimi pari, ki ne povezujejo ujemajočih regij. Še več, zgodi se lahko, da se na trenutni sliki v videu iskani pano pojavi na dveh mestih in obstajata dve homografiji.

Problem rešimo z metodo RANSAC. RANSAC (RAndom SAmple Consensus) je iterativna metoda, ki oceni parametre matematičnega modela na

podatkih, ki lahko vsebujejo velik delež napak. Na naključno izbranem vzorcu ocenimo model. Nato s cenitveno funkcijo na celotni množici podatkov ocenimo kvaliteto modela. Omenjen postopek ponavljamo, dokler ne dobimo modela, za katerega je dovolj verjetno, da je pravi.

Algoritem 1 Ocenjevanje homografije z RANSAC

Vhodni podatki: U, s, k_{max}, P, θ
Izhodni podatki $\mathcal{H}$

$K = inf$
$k \Leftarrow 0$
$i_{max} \Leftarrow 0$
while $k < K \wedge k < k_{max}$ **do**
  $S \subset U, |S| = s$
  $\mathcal{H}_S \Leftarrow DLT(S)$
  $i \Leftarrow C(U, \mathcal{H}_S, \theta)$
  if $i > i_{max}$ **then**
    $\mathcal{H} \Leftarrow \mathcal{H}_S$
    $i_{max} \Leftarrow i$
    $w \Leftarrow i/N$
    $K \Leftarrow \log(1 - P)\ \log(1 - w^s)$
    $k \Leftarrow k + 1$
  end if
end while
return $\mathcal{H}$

Potek ocenjevanja homografije z metodo RANSAC je predstavljena v razdelku Algoritem 1. Potrebujemo vhodne podatke, na katerih ocenjujemo model, to so v našem primeru pari ujemajočih točk $U = (x_1, x_1'), .., (x_i, x_i'), .., (x_N, x_N')$. Model, ki ga ocenjujemo je homografija med točkami $\mathcal{H}(x) = x'$. Funkcija $DLT()$ nam iz vzorca parov točk oceni homografijo H. Dobljeni model ocenjujemo na vseh parih točk s funkcijo $C(U, \mathcal{H}, \theta) = n$, kjer je n število parov iz U v skladu s homografijo H. Par je v skladu s homografijo, ko je razdalja med s $\mathcal{H}$ preslikanim x in x' manjša kot θ (enačba 4.7).

$$C(U, \mathcal{H}, \theta) = \#\{(x, x') \in U, |x' - \mathcal{H}(x)|_2 < \theta\} \tag{4.7}$$

Ostane nam še odločitev, kolikokrat ponoviti naključno izbiranje in ocenjevanje modela. Model je dobro ocenjen, če smo vsaj enkrat izbrali vzorec s samimi pravilnimi pari, torej pari, ki predstavljajo ujemajoči regiji. Označimo delež pravilnih parov v množici U z w. S s označimo velikost vzorca in z N

število vseh parov. Za približek verjetnosti, da smo v K poskusih izbrali vzorec s samimi dobrimi pari, izberemo P (enačba 4.8). Iz enačbe 4.8 v enačbi 4.9 izrazimo število poskusov.

$$P = 1 - (1 - w^s)^K \tag{4.8}$$

$$K = \frac{\log(1 - P)}{\log(1 - w^s)} \tag{4.9}$$

Algoritmu podamo množico parov U, najvišje dovoljeno število iteracij k_{max}, velikost vzorca s, verjetnost izbire pravega vzorca P in najvišjo napako θ, pri kateri par še označimo, da je v skladu z dano s homografijo. Vsako iteracijo izberemo vzorec S (vrstica 5) in na njem z metodo DLT ocenimo homografijo $\mathcal{H}_S$ (vrstica 6). S funkcijo $C()$ (enačba 4.7) preštejemo število točk v skladu s homografijo $\mathcal{H}_S$ (vrstica 7). Najboljšo najdeno homografijo hranimo v $\mathcal{H}$ in največje število parov v skladu s to homografijo v i (vrstici 9 in 10). Če smo našli najboljšo homografijo, spremenimo tudi oceno števila potrebnih korakov, ki ga izračunamo po enačbi 4.9. Algoritem se zaključi, ko preteče vsaj K ali k_{max} korakov.

Z manjšo spremembo lahko izboljšamo rezultate. Namesto shranjevanja najboljše homografije, shranjujemo pare točk, ki so v skladu z najboljšo homografijo. Ko se algoritem zaključi, na vseh shranjenih parih točk z metodo DLT ocenimo homografijo. Nato z ocenjeno homografijo preslikamo meje primera panoja in določimo lokacijo panoja v videu.

4.5 Metoda učinkovitega sledenja regij MSER

Ker bi bilo iskanje panojev v vsaki sliki računsko prezahtevno, smo uporabili metodo učinkovitega sledenja regij MSER [5]. Implementacija je zelo podobna iskanju regij MSER, vendar je v primerjavi z iskanjem na istem območju 2 do 4-krat hitrejša. Poleg hitrosti se izboljša tudi zanesljivost.

V začetku sledenja potrebujemo množico regij MSER, ki jih želimo slediti in vektorje njihovih lastnosti. V vektorju so naslednje lastnosti:

- srednja vrednost sivin na območju regije,
- velikost regije,
- težišče regije,
- stabilnost regije,

- višina ter širina pravokotnega okvira regije.

Na trenutni sliki zgradimo drevo ekstremnih regij in izračunamo njihove lastnosti. Zahtevnost algoritma se ne poveča, lastnosti se lahko izračunajo v $\mathcal{O}(n \log \log n)$. Vsaki regiji MSER se najde ustrezno ekstremno regijo v drevesu, tako da se izračuna evklidsko razdaljo med vektorjem lastnosti iskane regije in vektorji lastnosti vseh ekstremnih regij v drevesu. Izbere se regiji, ki sta si po vektorjih lastnosti najbližji.

Sledenje je v primerjavi z iskanjem hitrejše in bolj zanesljivo. Algoritem pohitrimo tako, da gradimo drevesi ekstremnih regij le na območju, kjer pričakujemo, da se bo regija, ki ji sledimo, pojavila. V obravnavanem primeru je to območje za faktor povečan pravokotni okvir okoli panoja, prestavljen v smeri gibanja panoja v videu. Dodatno lahko pohitrimo algoritem tako, da sledimo le pozitivne ali le negativne ekstremne regije, če je enih ali drugih dovolj, da lahko določimo natančen okvir okoli panoja. Reklamni panoji imajo dobro lastnost, da ponavadi prevladujejo regije le enega tipa, odvisno od barve v ozadju in barv črk ali oblik. Poleg tega se lahko pri grajenju drevesa omejimo na določen interval sivin. Denimo, da sledimo regijam, ki vsebujejo slikovne elemente z vrednostmi, ki ne presegajo 100, celoten interval sivin pa sega od vrednosti 0 do 255. Če se osvetlitev med sosednjimi slikami v videu ne spreminja veliko, lahko algoritem omejimo le na slikovne elemente do vrednosti nekaj čez 100. Prihranek je odvisen od deleža slikovnih elementov, ki jih ni bilo potrebno obiskati. Rezultat algoritma s tako optimizacijo je množica dreves ekstremnih regij.

Poleg pohitritve je sledenje tudi bolj zanesljivo. Lahko se zgodi, da regija MSER iz predhodne slike videa ni več maksimalno stabilna v trenutni sliki. V takem primeru sledenje daje boljše rezultate kot iskanje, saj ni omejeno le na maksimalno stabilne regije. Na slikah 4.6 se določi regije MSER. Na slikah 4.7 pa se regije določi v sliki (a) in se jim sledi v sliko (b) ter (c). Nekatere črke na sliki 4.6(b) in 4.6(c) ne predstavljajo več maksimalno stabilnih regij, zato jih iskanje ne zazna, a jim lahko sledimo. Primer je iz članka o MSER sledenju [5].

(a) (b) (c)

Slika 4.6: MSER iskanje

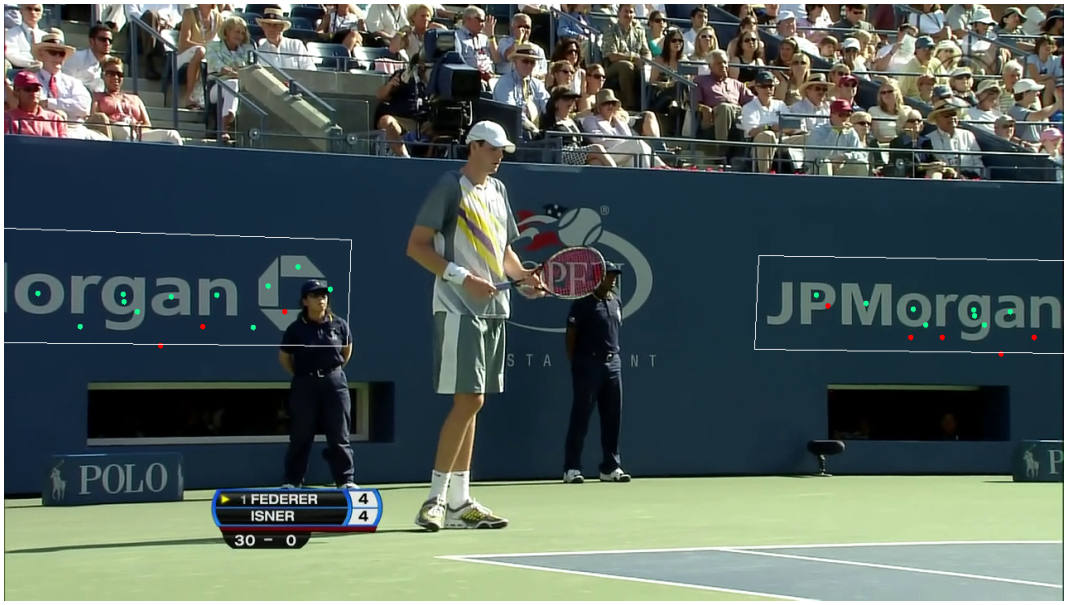
(a) (b) (c)

Slika 4.7: MSER sledenje

Poglavje 5

Prototip

Program za iskanje in sledenje panojev smo razvili v jeziku C++. Za delo s slikami smo potrebovali knjižnico CImg [27], za delo z matrikami pa knjižnico LAPACK++ [28]. Program je razvit za operacijski sistem Linux.



Slika 5.1: Primer dveh najdenih panojev

5.1 Opis delovanja

Najprej uporabnik določi, katere panoje želi menjati. Izvede se metoda za dodajanje primerov panojev objekta Tracker. Ustvari se objekt Subtracker, ki predstavlja dani primer panoja. V objektu Subtracker se ustvarita dva Tree objekta. Prvi je zgrajen na podani črno-beli sliki, drugi pa na njeni inverzni sliki. Tako dobimo v prvem primeru drevo ekstremnih regij, omejenih s slikovnimi elementi svetlejših barv, v drugem pa s temnejšimi. Koristno je imeti dve ločeni množici regij, ker nam to zmanjša zahtevnost algoritma v fazi iskanja ujemajočih regij, saj regiji dveh različnih tipov nista ujemajoči. Na zgrajenih drevesih sistem določi ekstremne regije in za njih izračuna opisnike.

Ta del se lahko izvede kadarkoli. Nov primer panoja se lahko doda tudi med delovanjem glavnega algoritma.

Glavni del algoritma je branje slik videa in iskanje ter sledenje panojev na njem. Ker je bolj kompleksen, smo ga predstavili na diagramu na sliki 5.2. Ko pride do nove slike videa, se izvede metoda razreda Tracker, ki ustvari nov objekt Frame v medpomnilniku teh objektov. V prototipu se uporablja medpomnilnik za 25 objektov. Če imamo prostor za večji medpomnilnik, nam to omogoča večjo zanesljivost algoritma.

Z vsako petindvajseto sliko videa (z začetkom pri prvi sliki) se iščejo panoji. V objektu Frame trenutne slike videa se, enako kot pri Subtrackerju, tudi tu ustvarita dva objekta Tree, ki predstavljata drevesi ekstremnih regij celotne slike videa. Da bi se izognili primeru, da algoritem ponovno najde isti pano, naredimo za vsako od dreves masko MSER. V njej se vse MSER, ki imajo center v okviru katerega izmed sledenih panojev, označijo za neveljavne.

Za vsak primer panoja (torej za vsak objekt Subtracker v Trackerju) opravimo več aktivnosti. Med veljavnimi regijami dreves objekta Frame in dreves Subtrackerja se po metodi, opisani v poglavju o teoretičnih osnovah, določi ujemajoče regije. Na težiščih ujemajočih regij obeh tipov se z RANSAC in metodo DLT išče homografijo. Če najdemo homografijo, smo našli pano.

Tu velja omeniti še problem z iskanjem panojev. Pogosto se dogaja, da najdena homografija ne predstavlja panoja. Uporabili smo preprosto rešitev. Za ujemajoče regije na sliki v videu in tiste na sliki primera panoja, smo izračunali, kakšen delež površine celotnega panoja pokrivajo. Če sta deleža preveč odstopala, smo najdeno homografijo zavrgli. Rešitev se je izkazala za dobro.

Ko imamo pravo homografijo, ustvarimo objekt Tobject v objektu Frame. Nastavimo mu najdeno homografijo, okvir okoli panoja in seznam regij MSER z vsakega izmed dreves slike videa, ki so v skladu s homografijo. Te regije

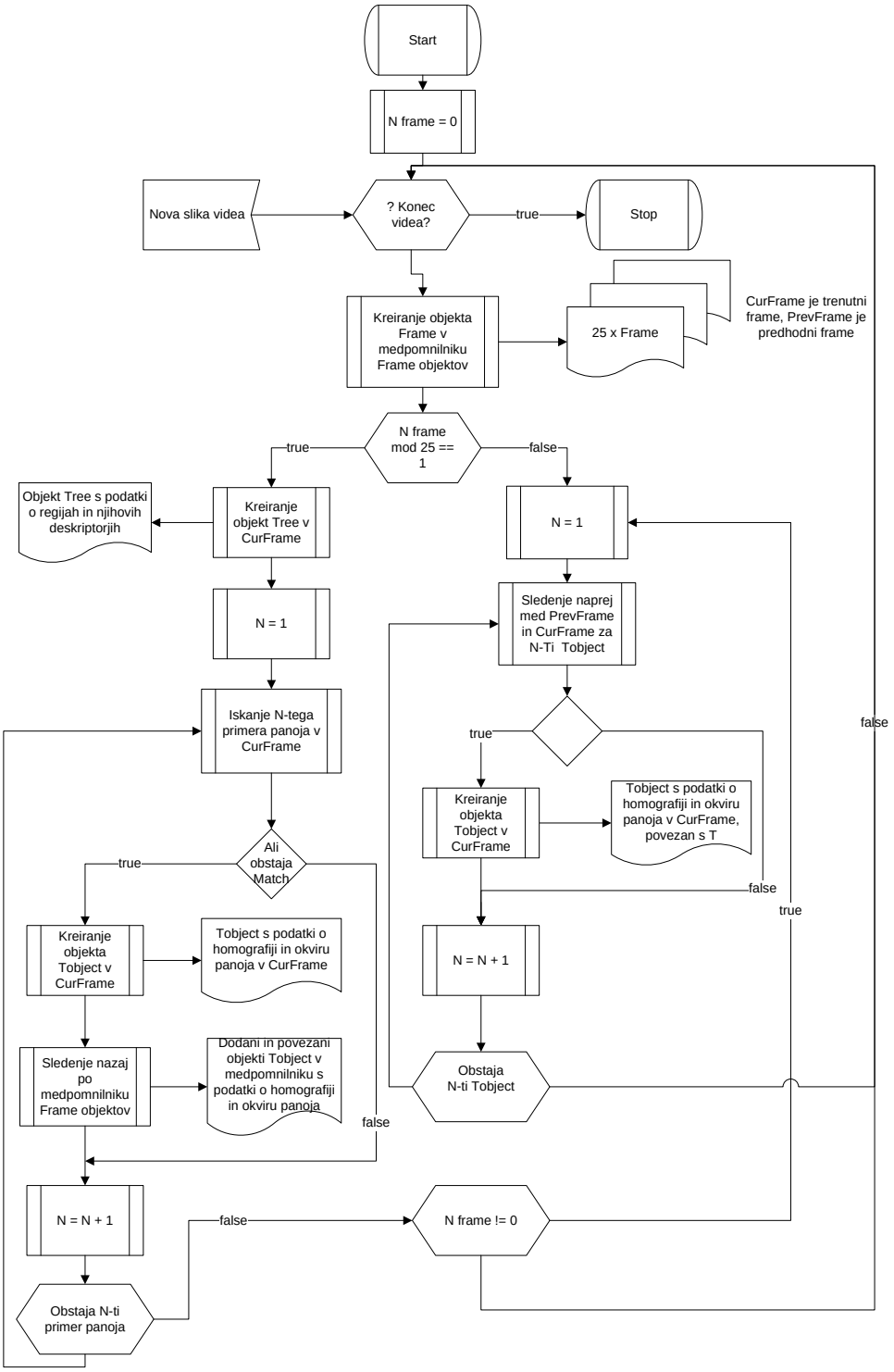
bodo kasneje uporabljene pri sledenju. Regije MSER, ki imajo center v okvirju najdenega panoja, pa označimo za neveljavne.

V primeru, da trenutna slika ni prva v videu, je potrebno sledenje nazaj po slikah videa v medpomnilniku, dokler panoja ne najdemo več oziroma pridemo do konca medpomnilnika. Sledenje posameznih regij poteka tako, kot je opisano v poglavju o teoretičnih osnovah. Ko imamo ujemajoče regije dveh sosednjih slik videa, se po istem postopku kot pri iskanju panoja, izračuna homografija. Poleg tega se še med sledenjem dodaja in odvzema regije. Če regije, ki smo jim do zdaj sledili, niso več v skladu z izračunano homografijo, jih odvzamemo. Če obstajajo regije v skladu s homografijo, ki jim do zdaj nismo sledili, jih dodamo. Ta pristop je nujen, če sledimo panojem dalj časa, sicer bi lahko zaradi prekrivanja panojev z igralci, izgubili sledeni pano.

Omenjeni postopek ponavljamo, dokler še najdemo kakšen pano, saj je možno, da se isti tip panoja na sliki pojavi večkrat.

Če je trenutna slika prva v videu, je iteracija končana, sicer pa moramo opraviti še sledenje vseh panojev, najdenih v prejšnji sliki.

Pri vsaki sliki videa, razen prvi, se opravi sledenje. Za vsak objekt Tobject v objektu Frame, ki predstavlja prejšnjo sliko videa, se opravi sledenje na trenutno sliko. Postopek sledenja je enak, kot v primeru sledenja nazaj.



Slika 5.2: Shema delovanja algoritma

Poglavje 6

Testi prototipa

Izvedli smo tri serije testov: v prvi smo testirali natančnost prototipa pri določanju lege panoja, v drugi uspešnost sledenja regij in v tretji časovno zahtevnost posameznih delov prototipa.

6.1 Natančnost določanja lege panoja

Na testnih slikah(prikazane na sliki 6.1) smo določili meje panojem in jih podali prototipu kot primere panojev. Nato smo slike transformirali in na njih s prototipom določil lokacijo panojev.

Natančnost prototipa pri določanju mej panoja smo izračunali s površino preseka in unije med najdenim območjem panoja in dejanskim območjem panoja na sliki. Natančnost predstavlja razmerje med obema in je definirana v enačbi 6.1. $R_{najdeni}$ je najdeno območje panoja, $R_{dejanski}$ je dejansko območje panoja. Funkcija $\mathcal{A}(x)$ je površina območja x.

$$\frac{\mathcal{A}(R_{dejanski} \cap R_{najdeni})}{\mathcal{A}(R_{dejanski} \cup R_{najdeni})} \tag{6.1}$$

Uporabili smo naslednje transformacije izbranih slik:

- zameglitev zaradi gibanja v horizontalni smeri za 1%, 2%, 3%, 4% in 5% celotne širine slike,
- zmanjšanje intenzitete na 0.85, 0.7, 0.55, 0.4 in 0.25 njihove prejšnje vrednosti,
- zmanjšanje slik na 2/3, 1/2, 2/5, 1/3 in 2/7 prejšnje velikosti,

zahtevnost transformacije	**1**	**2**	**3**	**4**	**5**
merilo	99	93	90	84	56
zameglitev	99	97	98	63	30
osvetlitev	96	96	96	73	65
perspektiva	97	94	91	83	74

Tabela 6.1: Natančnost iskanja panoja v prototipu

Slika 6.1: Slike uporabljene za testiranje

- 5 projekcijskih transformacij - od najmanjše spremembe do največje.

V izračunu so upoštevani primeri, ko prototip detektira pano. V primerih prizorov od daleč je bil pogosto delež najdenih ujemajočih regij premajhen in ni bilo mogoče določiti homografije.

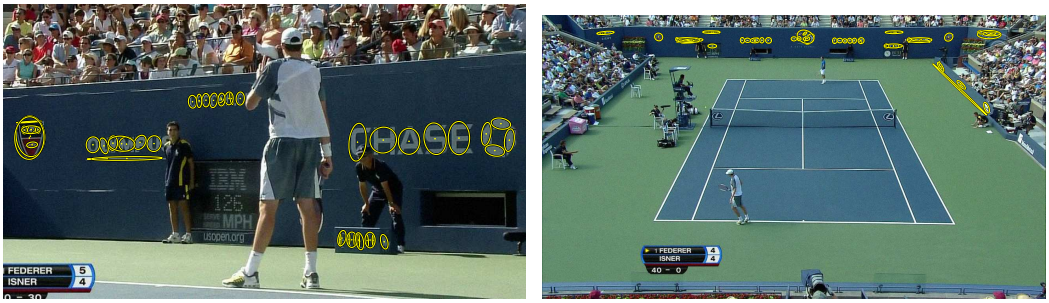
6.2 Uspešnost sledenja regij MSER

Pri testiranju natančnosti sledenja smo hoteli izvedeti, kolikšen delež regij algoritem uspešno sledi ob določeni velikosti spremembe med slikama. Test smo opravili na izbranih slikah prikazanih na sliki 6.1. Slike smo transformirali z izbranimi trasformacijami in preizkusili sledenje iz originalne na spremenjeno sliko. Za tak način smo se odločili, ker z njim lahko ocenimo odvisnost uspešnosti sledenja glede na spremembo slike.

Izbrane slike so skupno vsebovale 150 panojev. Slike smo razdelili v dve skupini: **posnetki od daleč** in **posnetki od blizu**. Razloga za ločevanje sta dva:

- zelo različna velikost regij,
- razlike pri hitrosti gibanja kamere.

Na sliki 6.2 so prikazane regije obeh tipov posnetkov.



Slika 6.2: Regije na posnetkih od blizu in daleč

Slike smo transformirali z naslednjimi transformacijami:

- translacija v horizontalni smeri,
- sprememba merila,
- sprememba intenzitete.

S transformacijo bomo testirali, kako se sledenje obnese pri gibanju kamere okoli navpične osi (v testih ponazorjeno s translacijo), s spremembo merila(zoom) in spremembo pogojev osvetlitve s spremembo intenzitete.

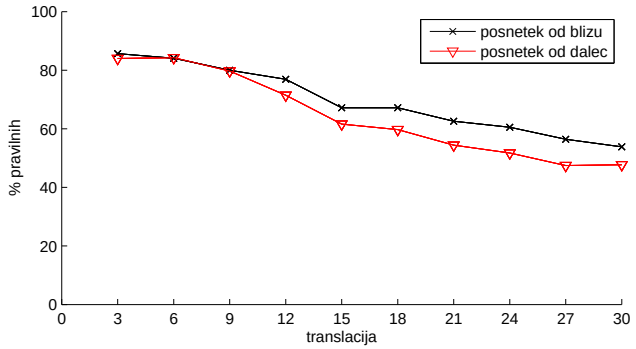
Pri testiranju smo določili regije MSER na originalni sliki in jim sledili na transformirano. Sledenje regije smo določili za pravilno, če se je težišče regije, iz originalne na transformirano sliko, spremenilo v skladu s transformacijo z napako manj kot θ. Pogoj je zapisan v enačbi 6.2. V enačbi x težišče regije na originalni sliki in x' težišče regije, ki je bila določena za ujemajočo na transformirani sliki. Transformacija med slikama je označena s $\mathcal{T}$.

$$|\mathcal{T}(x) - x'|_2 < \theta \tag{6.2}$$

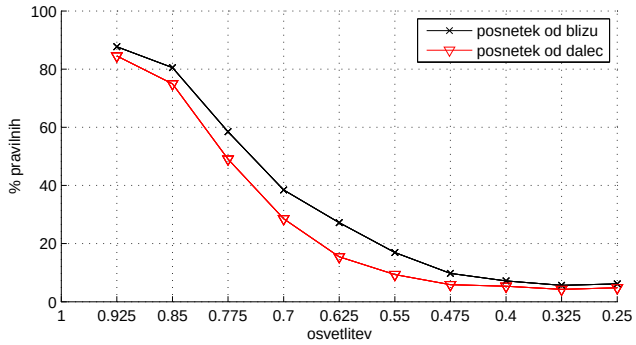
Za vsako izmed transformacij smo določili 10 stopenj spremembe. Pri translaciji smo zamaknili sliko v x smeri za $3, 6, 9, .., 3n, ..30$ slikovnih elementov (en slikovni element predstavlja razdaljo $1/1280$ širine slike), pri spremembi merila smo zmanjšali sliko na $95\%, 90\%, .., (100-5n)\%, .., 50\%$ in pri spremembi intenzitete smo vrednosti slikovnih elementov zmanjšali na $95\%, .., (100-7.5n)\%, .., 25\%$ prejšnje vrednosti. Vrednost θ smo določili 1.

Slika 6.3 prikazuje delež pravilno sledenih regij MSER glede na stopnjo izbrane transformacije. Parameter θ je nastavljen na 1 slikovno enoto, oziroma $1/1280$ širine slike.

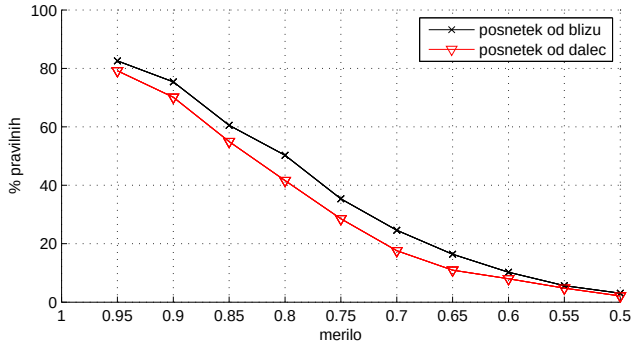
Poleg tega smo testirali še povprečno velikost translacije, ki je prisotna med sosednjima slikama v videu. Izbrali smo 11 posnetkov od blizu in 17 posnetkov



(a)

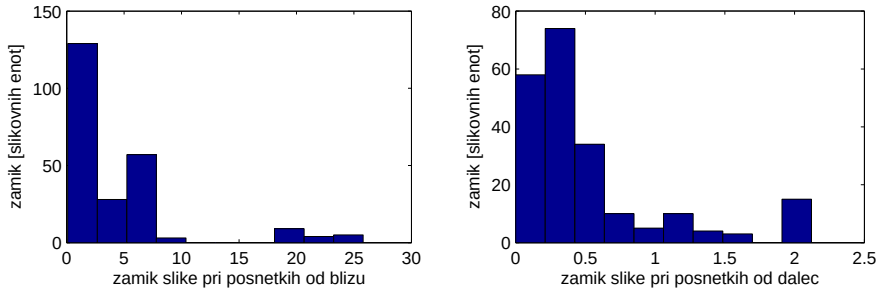


(b)



(c)

Slika 6.3: Delež pravilno sledenih regij glede na stopnjo transformacije slike



(a) (b)

Slika 6.4: Histograma velikosti zamika med sosednjima slikama

od daleč s skupno 800 slikami. S poravnavo sosednjih slik smo ugotovili translacijo med njima. Uporabljena metoda je opisana v članku "Efficient subpixel image registration algorithms"[9]. Na sliki 6.4 sta prikazana histograma velikosti zamika med sosednjima slikama v videu. Histogram (a) prikazuje zamik slike v posnetkih od daleč in histogram (b) od blizu.

V videu glavni problem predstavlja translacija, spremembe merila in osvetlitve minimalne. Iz testa natančnosti pri translaciji in testa velikosti translacije lahko sklepamo, da je delež pravilno sledenih regij v izbranem videu pri posnetkih od daleč okoli 80% pri posnetkih od blizu pa okoli 60%.

6.3 Časovna zahtevnost posameznih komponent prototipa

Namen testiranja časovne zahtevnosti je prikazati, kako zahtevni so posamezni deli algoritma. Test časovne zahtevnosti je potekal na enakih primerih kot testi natančnosti določanja mej panoja in uspešnosti sledenja.

Testirali smo s funkcijo *getrusage()* [22], ki vrne podatke o resursih, ki jih proces ali njegovi potomci rabijo. Poizvedovali smo po podatkih o porabi procesorskega časa v uporabniškem načinu.

Merili smo procesorski čas, porabljen za naslednje aktivnosti:

- **določanje regij** obsega aktivnosti grajenja obeh MSER dreves na sliki iz videa,
- **izračun opisnikov** je izračun opisnikov na najdenih regijah MSER iz videa,

	procesorski čas v ms
določanje regij	493
izračun opisnikov	143
določanje homografije	109
en korak sledenja - posnetki od daleč	5
en korak sledenja - posnetki od blizu	19

Tabela 6.2: Časovna zahtevnost prototipa

- **določanje homografije** je iskanje ujemajočih regij med sliko primera panoja in sliko iz videa ter ocenjevanje homografije,

- **en korak sledenja** je računanje drevesa ekstremnih regij na zanimivem območju, iskanje ujemajočih regij med prejšnjo in trenutno sliko ter računanje transformacije med slikama. Računanje drevesa predstavlja 85% vsega časa, porabljenega za sledenje.

Povprečno časovno zahtevnost sledenja smo testirali na istih primerih kot uspešnost sledenja. Merili smo povprečni čas porabljen za en korak sledenja enega panoja. Rezultate smo zaradi velikih razlik v velikosti panojev razdelili v dve kategoriji - posnetki od blizu in posnetki od daleč.

Čas določanja regij in računanja opisnikov smo beležili na slikah videa in njihovih transformiranih slikah. Skupno je bilo testiranih 570 slik velikosti 1280x720.

Določanje homografije smo beležili pri vsakem iskanju panoja v testu natančnosti določanja mej panoja. Skupno je bilo opravljenih 475 iskanj homografije.

Rezultati testiranja so v tabeli 6.2. Na podlagi teh podatkov lahko predvidevamo, kakšen procesorski čas je potreben na eno sliko videa. V ciklu določanja mej panoja se izvede določanje regij na sliki iz videa, izračuna opisnika regij in določa homografijo za vsak tip iskanega panoja. Če je pano določenega tipa najden, se iskanje ponovi, če obstaja še kakšen pano takega tipa. Iz tega sledi, da je porabljen čas enak seštevku časa določanja regij, izračuna opisnikov, določanja homografije za vsak iskani pano in še enkrat za vsak najdeni. Za vsak najdeni je potrebno izvesti tudi sledenje nazaj.

Določanje regij smo razdelili še na 5 podaktivnosti. Rezultati kažejo vpliv računanja lastnosti regij, ki so potrebne za sledenje, na časovno zahtevnost iskanja regij. Aktivnosti, katerim smo računali porabljen procesorski čas, so naslednje:

	procesorski čas v ms
inicializacija drevesa	22
inicializacija lastnosti regij	8
grajenje drevesa - osnova	190
grajenje drevesa - sledenje	24
ostalo	<4

Tabela 6.3: Časovna zahtevnost določanja MSER

inicializacija drevesa je branje slike, pretvorba v črno-belo, rezerviranje prostora za vse potrebne podatkovne strukture,

inicializacija lastnosti regij je določanje lastnosti regijam velikosti 1. Od tega je večina časa (78%) porabljenega za lastnosti potrebnih za sledenje, ostali del pa za lastnosti elipse, ki so potrebne za določanje približkov regij,

grajenje drevesa je grajenje drevesa ekstremnih regij brez operacij računanja lastnosti regij,

grajenje drevesa - izračun lastnosti regij je računanje novih lastnosti regij v drevesu, ko pride do združitve dveh regij pri grajenju drevesa. Večina operacij (78%) je potrebnih za združevanje lastnosti potrebnih za sledenje,

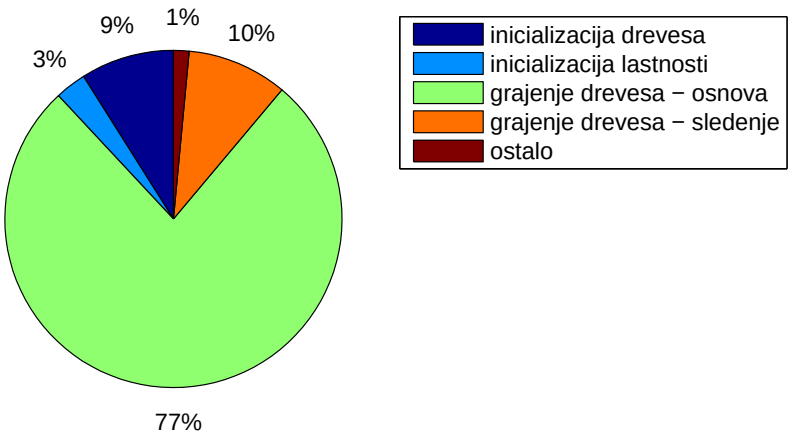
ostalo je določanje maksimalno stabilnih regij in določanje parametrov elipse.

Rezultati testiranja so v tabeli 6.3. Graf 6.5 predstavlja deleže časov posameznih aktivnosti.

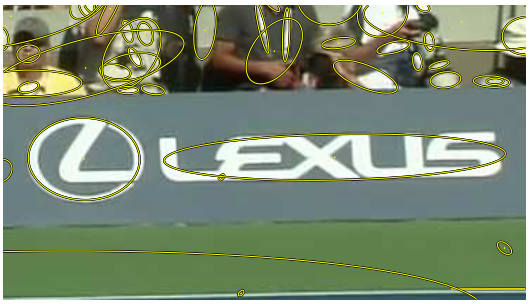
6.4 Problemi prototipa

Pri testiranju prototipa so se pokazale nekatere slabosti. Prva slabost je majhno število regij MSER na panoju. Homografije ni mogoče določiti v primerih, ko je število regij manjše od 4. Na sliki 6.6 je prikazan pano, ki vsebuje le 2 regiji. Takega panoja z našim pristopom ni mogoče najti.

Naslednja slabost je sledenje panojev izven vidnega polja. Homografije ni mogoče določiti v primerih, ko algoritem ne sledi vsaj štirim regijam, ki so popolnoma vidne. Na sliki 6.7(a) najdenemu panoju skušamo slediti na 6.7(b). Nekatere regije s slike (a) niso več enake oblike, ali pa so izven vidnega polja.



Slika 6.5: Delež procesorskega časa za posamezne dele računanja regij



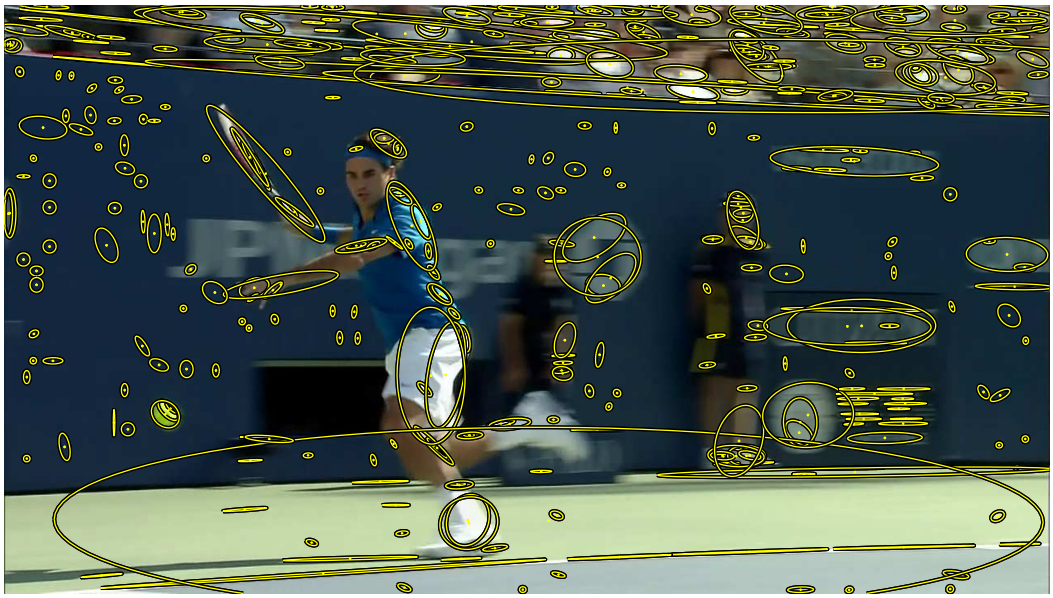
Slika 6.6: Pano s premalo regijami MSER

Le trem parom regij je mogoče pravilno slediti. V takem primeru ne moremo določiti meje panoja.

Kot smo pričakovali, iskanje in sledenje ne uspe pri veliki prisotnosti zameglitve zaradi gibanja kamere. Na sliki 6.8 so prikazane najdene regije MSER na takem posnetku. Regije MSER niso bile določene zaradi premalo jasnih mej med ozadjem in črkami. Na nekaterih panojih so se zaradi zameglitve črke povezale v eno regijo.

(a) (b)

Slika 6.7: Problem sledenja izven vidnega polja



Slika 6.8: Regije pri pristotnosti zameglitve

Poglavje 7

Sklepne ugotovitve

Cilj naloge je bil poiskati možno rešitev problema iskanja in sledenja panojev po zgledu sistema opisanega v članku "Real-time billboard substitution in a video stream" [17].

Ohranili smo podobno zgradbo sistema, vse dele sistema iskanja in sledenja pa smo nadgradili z najmodernejšimi pristopi. Originalni sistem je za značilne točke določil kote, v okolici katerih so bili prisotni slikovni elementi barv, ki so se pojavljale na panojih. Tak pristop ni odporen na spremembe merila in občutljiv na spremembe v osvetlitvi. Namesto tega smo uporabili detektor afino kovariantnih regij, ki določi regije, ki so stabilne v različnih pogojih osvetlitve in se spreminjajo skladno z afino transformacijo prostora. Določanje robov panojev je bilo v originalnem sistemu rešeno s poiskušanjem določanja afine transformacije med interesnimi točkami primera panoja in vsemi interesnimi točkami v sliki iz videa. V naši implementaciji določamo ujemajoče točke s primerjanjem opisnikov regij. Izbrane regije ponavadi obsegajo črke ali simbole, ki so zelo kontrastni z okolico. Take oblike so dobra podlaga za izračun zelo značilnih opisnikov Shape Context. S tem postopkom dobimo velik delež pravilnih parov ujemajočih regij, kar olajša ocenjevanje homografije. Za ocenjevanje homografije smo uporabili metodo DLT v kombinaciji z metodo RANSAC za robustnost pri prisotnosti napačni ujemajočih točk. Tudi tu je naš pristop boljši — ocenjujemo projekcijsko transformacijo, v izvirniku pa afino transformacijo. Drugače smo implementirali tudi sledenje. V originalnem sistemu je bilo sledenje realizirano podobno kot iskanje, le da je bilo izvedeno na interesni regiji. To smo nadomestili z metodo sledenja regij MSER. Izbrana metoda se je izkazala za hitro in enostavno, saj je večino funkcionalnosti delila z detektorjem regij MSER.

Rešitev problema smo opisali in ga razčlenil na komponente. Med možnimi

metodami smo na podlagi teoretičnih osnov in izdelanih testov izbrali najbolj ustrezne. Pri izbiri detektorja regij smo se omejili na 5 afinih detektorjev, pri izbiri opisnika pa na 10 tipov opisnikov. Po primerjalnem testu smo izbrali detektor MSER, ker se je izkazal za najhitrejšega in najbolj ponavljivega. Testirali smo uspešnost določanja ujemajočih regij z različnimi opisniki in izbrali opisnik Shape Context.

Razvili smo prototip in predstavil njegovo zgradbo in rezultate testiranj na realnih primerih. Prototip vsebuje vso glavno funkcionalnost iskanja in sledenja. Izdelava sistema za iskanje, sledenje in zamenjavo panojev je zahtevna, saj mora sistem biti hiter, da zagotavlja prenos v živo, izjemno natančen in robusten, da gledalec ne zazna razlik med navadnim prenosom in prenosom z zamenjanimi panoji.

Metode s katerimi smo rešil problem iskanja in sledenja panojev so se izkazale za možen pristop k izdelavi sistema za iskanje, sledenje in zamenjavo reklamnih panojev med prenosi. Izbira detektorja in opisnika regij se je izkazala za dobro, edina slabost je majhno število regij, ki jih dobimo z detektorjem. Primer panoja ima povprečno okoli 10 regij, kar je malo glede na to, da potrebujemo vsaj štiri, da sploh dobimo homografijo in še več za dober približek. Druga kritika detektorja se nanaša na samo razporeditev regij na panoju. Pogosto se zgodi, da so njihovi centri razporejeni skoraj kolinearno, denimo na horizontalni črti na sredini panoja. To nam poveča napako pri izračunu homografije z metodo DLT. Možna rešitev bi bila kombiniranje detektorja MSER s kakšnim drugim detektorjem, kot so to realizirali v članku "Video Google: A Text Retrieval Approach to Object Matching in Videos" [23], kjer predstavijo sistem za pridobivanje ključnih slik videa, ki vsebujejo dani predmet. Ugotavljajo, da se uspešnost izbranega detektorja spreminja s prizori in je zato uporaba dveh detektorjev bolj zanesljiva.

Prvotno smo se izdelave prototipa lotili v MATLAB-u, kar pa se je izkazalo za napako. MATLAB je bilo zelo dobro orodje za testiranje posameznih algoritmov in kot pomoč pri odločitvi, kakšne metode naj uporabimo v prototipu, vendar za obsežne sisteme ni primeren. Prvi problem je počasnost, drugi togost samega jezika, ki nas omejuje, ko je potrebno implementirati nematematične algoritme.

Poleg tega bi bilo bolj smiselno uporabo čim več že implementiranih rešitev, denimo knjižnico OpenCV. Pri izdelavi prototipa smo uporabili le knjižnico LAPACK++ za operiranje z matrikami in CImg za operiranje s slikami ter program za izračun opisnikov.

Izdelan prototip v tej fazi razvoja ni primeren za komercialno uporabo, je pa dobra podlaga za nadaljnje raziskovanje na področju iskanja, sledenja in

zamenjave reklamnih panojev. Glavne smeri raziskovanja bi bile izboljšanje iskanja in sledenja ter zagotavljanje delovanja sistema v vseh pogojih.

Dodatek A

Dodatek

A.1 Primer grajenja drevesa ekstremnih regij

Delovanje algoritma bomo ponazorili na primeru slike prikazane na sliki A.1. Na A.1(a) so prikazani slikovni elementi in njihove vrednosti, na A.1(b) pa zaporedne številke slikovnih elementov. Slikovni element ima največ 4 sosede - zgornjega, spodnjega, levega in desnega. Sosede nekega elementa gledamo v vrstnem redu po njihovem zaporednem številu.

V opisu so slikovni elementi označeni z zaporedno številko, vozlišča drevesa pa z $\{x_1, .., x_n\}$, kjer so x_i slikovni elementi, ki pripadajo povezani komponenti, ki jo vozlišče predstavlja.

Najprej vsem slikovnim elementom določimo, da so vozlišče in s tem drevo velikosti 1. Nato obiskujemo slikovne elemente po naraščujoči vrednosti. Začetnih korakov ne bomo opisovali. Denimo, da smo že obiskali slikovne elemente pod vrednostjo 80. Rezultat sta drevesi na sliki A.2.

Naslednji po vrsti je slikovni element 3. Pogledamo njegove že obiskane sosede. Edini tak je element 0. Ker 0 in 3 ne pripadata istem drevesu in

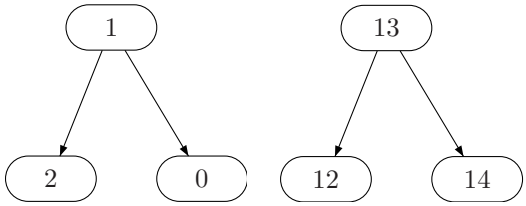
20	40	30
80	80	80
90	110	80
80	80	80
10	60	50

(a) Vrednosti slikovnih elementov

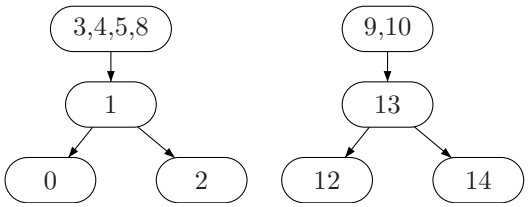
0	1	2
3	4	5
6	7	8
9	10	11
12	13	14

(b) Zaporedne številke slikovnih elementov

Slika A.1: Slika uporabljena v primeru grajenja drevesa ekstremnih regij



Slika A.2: Drevesi po obisku slikovnega elementa 13

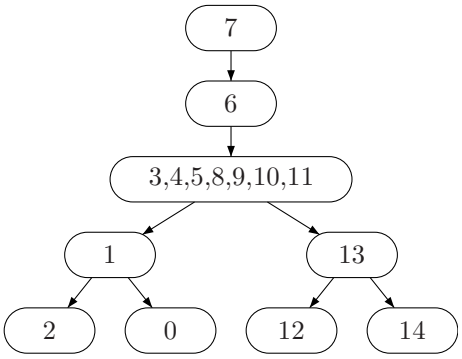


Slika A.3: Drevesi po obisku slikovnega elementa 1

$root(0) = \{1\}$ in $root(3) = \{3\}$ ne predstavljata povezane komponente iste barve, $\{1\}$ postane potomec $\{3\}$. Nato obiščemo element 4. Njegova obiskana soseda sta 1 in 3. Pogledamo element 1. Ker $root(1) = \{3\}$ in $root(4) = \{4\}$ predstavljata povezani komponenti iste vrednosti, ju združimo v isto vozlišče $\{3, 4\}$. Pogledamo element 3. Ker pripada istemu drevesu kot 4, gremo dalje. Podobne še za elementa 5 in 8. Rezultat je drevo prikazano na sliki A.3(a).

Nato obiščemo element 9. Edini obiskani sosed je element 12. Ker predstavlja $root(12) = \{13\}$ povezano komponento nižje vrednosti kot tista v 9, postane $\{13\}$ potomec $\{9\}$. Obiščemo element 10. Obiskana soseda sta 9 in 13. Pogledamo element 9 in ga združimo $\{9\}$ z $\{10\}$ v $\{9, 10\}$. Pogledamo element 13. Ker sta 13 in 10 v istem drevesu, gremo dalje. Na tej stopnji imamo zgrajeni drevesi na sliki A.3.

Obiščemo element 11. Pogledamo soseda, element 8. Vozlišči $root(8) = \{3, 4, 5, 8\}$ in $root(11) = 11$ združimo v $\{3, 4, 5, 8, 11\}$. Pogledamo soseda, element 10, ter združimo korena v $\{3, 4, 5, 8, 10, 11\}$. Obiščemo slikovni element 6, pogledamo soseda 3 in nastavimo vozlišče $root(3) = \{3, 4, 5, 8, 10, 11\}$ za potomca vozlišča $\{3\}$. Ostali obiskani sosedi so v istem drevesu kot 3. Obiščemo vozlišče 7, pogledamo soseda 4 in njegov koren vstavimo za potomca $\{7\}$. Obiskali smo vse slikovne elemente in algoritem se je zaključil. Končno drevo je prikazano na sliki A.4.



Slika A.4: Zgrajeno drevo ekstremnih povezanih komponent

A.2 Opis razredov

Glavni razredi v prototipu so Tracker, Subtracker, Frame, Tree in Tobject.

Tracker je glavni razred, ki vsebuje:

- medpomnilnik 25 objektov tipa Frame,
- seznam objektov Subtracker, vsak predstavlja en tip iskanega panoja, ki ga je uporabnik definiral s sliko,
- metode za dodajanje slik primerov panojev,
- metodo za dodajanje naslednje slike videa, branje spremenjene slike videa in branje podatkov o najdenih panojih,
- metodo Match za iskanje panojev na trenutni sliki videa,
- metodo Track, ki sledi najdenim panojem iz prejšnje slike videa v trenutno sliko.

Subtracker predstavlja primer panoja in vsebuje:

- sliko panoja, ki jo je podal uporabnik,
- mejo panoja na sliki,
- drevo ekstremnih regij, opisniki regij MSER (objekt Tree).

Frame predstavlja eno sliko videa in vsebuje:

- sliko iz videa,

- seznam objektov Tobject, ki predstavljajo najdene ali sledene panoje na sliki,
- v fazi iskanja vsebuje tudi drevo ekstremnih regij in opisnike regij MSER (objekt Tree).

Tree vsebuje:

- drevo ekstremnih regij,
- lahko vsebuje tudi opisnike regij MSER,
- metodo za sledenje, torej iskanje ujemanj regij nekega drugega drevesa s svojimi regijami.

Tobject predstavlja sledeni pano v sliki videa. Vsebuje:

- kazalec na Subtracker, ki predstavlja tip najdenega panoja,
- homografijo med primerom slike panoja v Subtracker in sliko videa v Frame,
- okvir panoja na sliki,
- morebitne povezave na Tobject objekta, ki predstavljata isti pano v predhodni in naslednji sliki videa,
- drevo ekstremnih regij v objektu Tree, ki je bilo izračunano na zanimivem območju,
- metodo za izračun zanimivega območja na naslednji sliki videa.

Slike

Tabele

Literatura

[1] S. Belongie, J. Malik, J. Puzicha, "Shape matching and object recognition using shape contexts" v reviji *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, apr. 2002, str. 509-522.

[2] S. Belongie, J. Malik, J. Puzicha, "Matching Shapes", v zborniku *Proceedings of the Eighth IEEE International Conference on Computer Vision*, vol. 1, 2001, str. 454-461.

[3] G. Cai, L. Chen, Li Junchang, "Billboard advertising detection in sport TV," v zborniku *Seventh International Symposium on Signal Processing and Its Applications, 2003. Proceedings.* , vol. 1, 1-4 julij 2003, str. 537-540.

[4] J. Canny, "A Computational Approach To Edge Detection", v reviji *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, nov. 1986, št. 6, str. 679-698.

[5] M. Donoser, H. Bischof, "Efficient Maximally Stable Extremal Region (MSER) Tracking", v zborniku *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 17-22 junij 2006, str. 553 - 560.

[6] R. O. Duda, P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York, 1973, 482 strani.

[7] R. O. Duda, P. E. Hart,"Use of the Hough Transformation to Detect Lines and Curves in Pictures", *Communications of the ACM*, vol. 15, 1973, str. 11 - 15.

[8] M. A. Fischler, R. C. Bolle, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," v knjigi *Readings in computer vision: issues, problems, principles, and paradigms*, Morgan Kaufmann Publishers Inc., 1987.

[9] M. Guizar-Sicairos, S. T. Thurman, J. R. Fienup, "Efficient subpixel image registration algorithms", v zborniku *Optics Letters*, vol. 33, št. 2, 15. jan. 2008, str 156-158.

[10] R. I. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision* , 2. izdaja, Cambridge University Press, 2002.

[11] Y. Ke, R. Sukthankar, *PCA-SIFT: A More Distinctive Representation for Local Image Descriptors*, v zborniku *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004, str. 506-513.

[12] I. T. Jolliffe, *Principal Component Analysis*, 2. izdaja, Springer, 487 strani.

[13] S. Lazebnik, C. Schmid, J. Ponce, "Sparse Texture Representa- tion Using Affine-Invariant Neighborhood", v *Proceedings of the 2003 Conference on Computer Vision and Pattern Recognition*, 2003, str. 319-324.

[14] T. Lindeberg, "Scale-space theory: A basic tool for analysing structures at different scales", v reviji *Journal of Applied Statistics*, vol. 21, 1994 , str. 224-270.

[15] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", v reviji *International Journal of Computer Vision*, vol. 60, 2004, str. 91-110.

[16] J. Matas, O. Chum, M. Urban, T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions" v zborniku *British Machine Vision Conference*, 2002, str. 384-393.

[17] G. Medioni, G. Guy, H. Rom, A. Franois, "Real-time billboard substitution in a video stream" v zborniku *Proceedings of the 10th Tyrrhenian International Workshop on Digital Communications*, 1998, str. 71-84.

[18] K. Mikolajczyk, C. Schmid, "Scale & Affine Invariant Interest Point Detectors" v reviji *International Journal of Computer Vision*, vol. 60, okt. 2004 ,str. 63-86.

[19] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. Van Gool, "A Comparison of Affine Region Detectors", v zborniku *International Journal of Computer Vision*, vol. 65, 2005, št. 1-2, str. 43-72.

[20] K. Mikolajczyk, C. Schmid, "A Performance Evaluation of Local Descriptors", v *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, okt. 2005, št. 1, str. 1615-1630.

[21] L. Najman, M. Couprie, "Quasi-linear algorithm for the component tree", v knjigi *SPIE Vision Geometry XII*, San Jose, Kalifornija, ZDA, SPIE-International Society for Optical Engine, 2004.

[22] S. Sirca, "Meritev časovne zahtevnosti programov na sistemih linux", 2006, dostopno na http://predmeti.fmf.uni-lj.si/mafiprak/naloga1/timers.pdf.

[23] J. Sivic, A. Zisserman, "Video Google: a text retrieval approach to object matching in videos", v *Proceedings of the Ninth IEEE International Conference on Computer Vision*, vol. 2, 13-16 okt. 2003, str. 1470-1477.

[24] T.Tuytelaars, L. Van Gool "Matching Widely Separated Views Based on Affine Invariant Regions", v reviji *International Journal of Computer Vision*, vol. 59, 2004, št. 1, str. 61-85.

[25] A. K. Watve, S. Sural, "Detection of on-field advertisement billboards from soccer telecasts", v zborniku *IET International Conference on Visual Information Engineering, 2006*, 26-28 sept. 2006, str. 12-17.

[26] Spletna stran Affine Covariant features, http://www.robots.ox.ac.uk/ vgg/research/affine/

[27] Spletna stran The CImg Library, http://cimg.sourceforge.net/

[28] Spletna stran Lapack++: v2.5.2 API Documentation, http://lapackpp.sourceforge.net/