Univerza v Ljubljani
Fakulteta za računalništvo in informatiko ter
Fakulteta za matematiko in fiziko

Blaž Peterlin

Izboljšava metode za sledenje objektov z dinamičnimi grafi

diplomsko delo
na interdisciplinarnem univerzitetnem študiju
računalništva in matematike

Mentor: doc. dr. Matej Kristan

Ljubljana, 2016

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Vizualno sledenje objektov je trenutno izredno aktivno področje računalniškega vida. Zgolj v zadnjih nekaj letih smo priča izjemnemu porastu števila novih sledilnih algoritmov, o čemer pričajo številni pregledni članki in velika udeležba na mednarodnih izzivih. Rezultati izzivov VOT pričajo o potencialu sledilnikov, ki temeljijo na delih. V nalogi izberite sledilnik iz te družine sledilnikov ter predlagajte izboljšave. Izboljšave kvantitativno podprite z analizo na standardni zbirki VOT.

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Blaž Peterlin,

z vpisno številko 63080385,

sem avtor diplomskega dela z naslovom:

Izboljšava metode za sledenje objektov z dinamičnimi grafi

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom
  doc. dr. Mateja Kristana
- 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 26. avgusta 2016 Podpis avtorja:

Zahvaljujem se svojemu mentorju doc. dr. Mateju Kristanu za potrpežljivost, nasvete in entuziazem, ki so mi dali dodatno motivacijo za dokončanje diplomskega dela. Globoko sem hvaležen tudi svoji družini za nenehno podporo skozi nastajanje diplomskega dela.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
DGT	Dynamic Graph Tracker	Sledilnik na podlagi dinamičnih grafov
SVM	Support Vector Machine	Metoda podpornih vektorjev
MRF	Markov Random Field	Markovsko slučajno polje

Povzetek

Naslov: Izboljšava metode za sledenje objektov z dinamičnimi grafi

Diplomska naloga obravnava izboljšavo sledilnika za kratkotrajno vizualno sledenje objektom. Sledilnik modelira sledilni objekt kot dinamični graf slikovnih regij, kjer so bližnje regije med seboj povezane. Struktura in prilagodljivost grafa sta uporabna za kvalitetno sledenje navkljub obsežnim vizualnim spremembam sledilnega objekta skozi zaporedje slik. Sledilnik se je v preteklih primerjavah s konkurenco izkazal kot kvaliteten, a z nekaj očitnimi pomanjkljivostmi, predvsem pri primerih s pogosto spreminjajočo se osvetlitvijo objekta. Na podlagi poglobljene analize sledilnika predlagamo nekaj izboljšav. Odkrili smo pomanjkljivo implementacijo v sklopu razdelitve slike na ospredje in ozadje. Predlagamo ustrezno izboljšavo, ki sledilniku poveča natančnost, predvsem v primerih hitrih sprememb osvetlitve. Poleg tega predlagamo dinamično prilagajanje omenjene razdelitve glede na velikost nastalega ospredja, ki sledilniku poveča zanesljivost trajnega sledenja tarči. Analizo in primerjavo razvitih izboljšav smo izvedli s pomočjo ocenjevalnih algoritmov iz tekmovanja VOT2015. Popravek algoritma v sklopu razdelitve slike na ospredje in ozadje izboljša natančnost sledilnika predvsem v primerih hitrih sprememb osvetlitve sledilnega objekta, hkrati pa mu izboljša zanesljivost. Obenem naredi dinamično prilagajanje razdelitve sledilnik natančnejši in precej zanesljivejši v veliki večini situacij izven hitrih sprememb osvetlitve.

Ključne besede: računalniški vid, vizualno sledenje objektom, model na podlagi delcev, dinamični graf, analiza strukture grafa.

Abstract

Title: An improved dynamic graph tracking algoritm

We propose several improvements of an existing baseline short-term visual tracking algorithm. The baseline tracker applies a dynamic graph representation to track the target. The target local parts are used as nodes in the graph, while the connections between neighboring parts represent the graph edges. This flexible model proves useful in the presence of extensive target visual changes throughout the sequence. A recent benchmark has shown that the tracker compares favorably in performance with other state-of-the-art trackers, with a notable weakness in cases of input sequences with high variance in scene and object lighting. We have performed an in-depth analysis of the tracker and propose a list of improvements. With respect to an unstable component in the tracker implementation of the foreground/background image segmentation, we propose an improvement which boosts the accuracy in cases of rapid illumination change of the target. We also propose a dynamic adjustment of the aforementioned segmentation with respect to the size of the resulting foreground, which improves tracking reliability and reduces the number of tracking failures. The implemented improvements are analyzed on the VOT2015 benchmark. Fixing the unstable component yields improvements in cases of rapid illumination change and reduces failure rate, while the dynamic segmentation adjustment improves tracking accuracy and robustness in the vast majority of cases, barring rapid illumination change.

Keywords: computer vision, visual object tracking, part-based model, dynamic graph, graph structure analysis.

Poglavje 1

Uvod

1.1 Motivacija

Zajem objektov iz videoposnetkov je temeljna zahteva z razmeroma dolgo zgodovino širom področja računalniškega vida [1]. Osnovna naloga je natančno sledenje objekta v videoposnetku - iz vsake slike videa je potrebno izvleči položaj želenega sledilnega objekta (glej Sliko 1.1). Problem lahko poglobimo z zahtevo po simultanemu sledenju več objektom [2] (angl. "multi-target") namesto enemu [26, 47, 48] (angl. "single-target"). Po drugi strani imamo lahko na voljo več hkratnih videoposnetkov, ki prikazujejo isto sceno [49] (angl. "multi-camera"), namesto enega [50] (angl. "single-camera").

Težnja po kakovostnem zajemu objektov iz videoposnetkov se poraja pri

Slika 1.1: Primer sledenja objektu iz videoposnetka. Rešitev sledenja je prikazana z očrtanim pravokotnikom.

mnogih realnih programskih zahtevah, naštejmo jih le nekaj:

- napredni varnostni sistemi s kamerami,
- medicinska slikanja,
- kompresija in urejanje videa,
- zajem gibanja v okviru interakcije človek – računalnik.

V praksi se pri reševanju tega problema pogosto srečujemo z naslednjimi težavami, ki kvarijo kakovost digitalnega zajema objekta:

1. Sprememba osvetlitve (angl. "illumination change"). Zaradi premikanja objektov po prostoru in perturbacije virov osvetlitve se lahko barvna sestava objekta skozi čas drastično spremeni.
2. Sprememba velikosti objekta (angl. "object size change"). Objekt se na videoposnetku lahko veča ali manjša, glede na bližino kameri.
3. Delna ali polna zakritost dela objekta zaradi okolice (angl. "object occlusion").
4. Sprememba izgleda objekta (angl. "object deformation"). Izraz vključuje spremembo vidne površine objekta zaradi rotacije in/ali preoblikovanja strukture objekta.

Zaradi čedalje pogostejše težnje po kakovostni rešitvi problema in večje dostopnosti do dovoljšnje računske moči se zadnja leta po hitrem postopku razvijajo čedalje boljše rešitve. Stanje in primerjavo raznolikih rešitev je moč spremljati preko raznih tekmovanj [3, 4, 5, 6, 7, 8], med drugim tudi na tekmovanju Visual Object Tracking Challenge (s kratico *VOT*) [26].

VOT se osredotoča na problem sledenja enojnemu objektu z eno kamero. Natančneje od prisostvujočih sledilnikov zahteva:

- naj bodo splošni, brez predhodnih oz. parametričnih informacij o sledilnem objektu (nekatere delavnice se osredotočijo npr. specifično na sledenje človeškemu obrazu),

- sledilnik dobi kot začetno informacijo le očrtani pravokotnik (poravnan po oseh) v prvi sliki, znotraj katerega se nahaja objekt, zatem pa mora sproti za naslednje slike izračunati nadaljnje očrtane pravokotnike objekta,
- sledilniku ni treba posebej preverjati, če znotraj posnetka izgubi sled za iskanim objektom, saj bo avtomatsko postavljen nazaj na pravilne tirnice, če za objekt predolgo določa napačno lokacijo na slikah videa.

Podrobnejši opis ocenjevalca sledilnikov v okviru VOT sledi v Poglavju 4.

1.2 Sorodna dela

Najuspešnejši algoritmi za sledenje objektom spadajo v dve kategoriji glede na osnovni pristop k problemu: holistični sledilniki, ki tarčo modelirajo kot eno skupno celoto, in pa sledilniki z regijami, ki tarčo modelirajo kot strukturo več med seboj povezanih regij. Sledilniki pod drobnogledom nadalje uporabljajo raznolik razpon orodij.

Generativni holistični sledilniki sestavijo model o izgledu tarče, nato pa poiščejo za ta model lokacijo na sliki, ki maksimira podobnost med modelom in okolišem lokacije na sliki. Med generativne modele uvrščamo npr. bližinske histograme [9], glasovanja glede na podobnost razpršenih slik blizu tarče [10], glasovanja na podlagi metode glavnih osi (angl. "Principal Component Analysis") [11], in kombiniranje bližnjih značilk [10, 12].

Diskriminativni holistični sledilniki se po drugi strani lotevajo problema tako, da ga prevedejo na problem razločevanja slike na ospredje in ozadje - ospredje slike je tisti del slike, kjer se nahaja sledilni objekt, ozadje slike pa obsega vse ostalo. Nekateri sledilniki uporabljajo binarne klasifikatorje [13, 14] (npr. metodo podpornih vektorjev in Adaboost). Uporablja se tudi prirejena strukturirana metoda podpornih vektorjev, ki namesto binarne klasifikacije le glasuje za ospredje ali ozadje [17]. Uspešni so algoritmi, ki uporabljajo naučene korelacijske filtre, ki sliko razdelijo na ospredje in ozadje [57, 15, 16].

Sledilniki z regijami so odpornejši na spremembo izgleda objekta in delno zakritost objekta. Za določanje regij je na voljo več različnih strategij: enakomerna porazdelitev regij po sliki [18, 19, 20], združevanje razbitij [21, 22, 23], kombiniranje bližnjih točkovnih značilk [24], itd. Tudi struktura regij je lahko definirana na več načinov: model zvezdice temelji na relativnih lokacijah vsake ospredne regije glede na centralno lokacijo objekta [18, 20], model neusmerjenega grafa interpretira regije kot vozlišča grafa [21], itd.

Izsledki iz VOT2013 [42], VOT2014 [43], in VOT2015 [44] kažejo, da so na prvih mestih povečini sledilniki brez znanja o strukturi sledilnega objekta. Ena od izjem je sledilnik Dynamic Graph Tracker (v nadaljevanju DGT) [35]. DGT deluje na podlagi regij - značilk z barvnimi karakteristikami in se močno naslanja na ohranjanje medsebojne strukture značilk v ospredju. V VOT2014 se je DGT izkazal kot kakovosten sledilnik, ki pa je izpadel iz samega vrha zaradi slabših rezultatov v posnetkih, kjer so bili sledilni objekti prepogosto zakriti ali pa se je osvetlitev preveč spreminjala [36]. Sledilnik se je izkazal kot najzanesljivejši v primerih visoke dinamike velikosti sledilnega objekta.

1.3 Prispevki

V diplomski nalogi se osredotočamo na sledilnik DGT [35]. Z eksperimentalno analizo originalnega sledilnika smo identificirali njegove glavne pomanjkljivosti in predlagamo potencialne izboljšave. Izvorni sledilnik je pogosto izgubil sled za sledilnim objektom, kadar je le-ta v videoposnetku prehitro spreminjal barvno sestavo. V Poglavju 3.1 je opisan pogost vzrok ranljivosti ter predlagan popravek implementacije sledilnika. Popravek hkrati izboljša zgornjo mejo časovne zahtevnosti algoritma.

Občasno se zgodi, da izvorni sledilnik kot del objekta nehote zajame tudi kos ozadja za objektom, ker je podobnih barv ali pa se zdi, da spada v strukturo objekta. V Poglavju 3.2 opisujemo predlog izboljšave, ki pogosto prepreči takšno divergenco sledilnika. Uspešnost izboljšav smo analizirali s

pomočjo ocenjevalnega protokola iz VOT2015 [44].

1.4 Struktura naloge

Diplomsko delo je v nadaljevanju razdeljeno na štiri poglavja. Poglavje 2 vsebuje teoretično podlago ter podroben opis izvornega sledilnika Dynamic Graph Tracker [35]. Sledi Poglavje 3 s predlaganimi izboljšavami sledilnika. Implementacija izboljšav in analiza uspešnosti sta opisani v Poglavju 4, nakar v Poglavju 5 sledi še sklep in predlog dodatnih možnih nadgradenj algoritma.

Poglavje 2

Izvorni sledilnik DGT

V tem poglavju podrobno predstavimo sledilnik DGT. V Poglavju 2.1 so predstavljeni teoretični koncepti, potrebni za razumevanje opisa algoritma. Zavoljo razumevanja je nato začrtana osnovna ideja algoritma DGT v Poglavju 2.2, zatem pa je v Poglavju 2.3 algoritem še podrobno opisan. Poglavje predpostavlja poznavanje osnov teorije grafov, linearne algebre, in strojnega učenja.

2.1 Teoretične osnove

2.1.1 Barvni prostori

Na področju računalniškega vida se za različne potrebe uporablja nekaj različnih barvnih prostorov. V nadaljevanju opisujemo le prostore RGB, Luv ter HSV, ki so nujni za razumevanje delovanja sledilnika DGT. Širša obravnava barvnih prostorov se nahaja v [59]. Za potrebe razlage postavimo vse dimenzije prostorov na interval $[0, 1]$.

- RGB je najpogostejši 3D barvni prostor, kjer vsaka dimenzija predstavlja vsebnost ene od treh barv (rdeča, zelena in modra). Vsaka od treh vrednosti opiše, koliko svetlobe v ustrezni barvi je potrebno oddati.
- HSV je eden od alternativnih barvnih prostorov v treh dimenzijah. Naj

za barvo c poznamo vrednosti v prostoru RGB. V pomoč si definirajmo vmesne spremenljivke:

$$\begin{gathered} M_{HSV} \coloneqq \max(R,G,B) \\ m_{HSV} \coloneqq \min(R,G,B) \\ C_{HSV} \coloneqq M_{HSV} - m_{HSV} \\ r \coloneqq \frac{M_{HSV}-R}{C_{HSV}},\ g \coloneqq \frac{M_{HSV}-G}{C_{HSV}},\ b \coloneqq \frac{M_{HSV}-B}{C_{HSV}}. \end{gathered} \tag{2.1}$$

Vrednosti v prostoru HSV izračunamo na naslednji način [45]:

- Za barvni odtenek (angl. "hue") H najprej izračunamo še vmesno vrednost H_1 kot

$$H_1(c) \coloneqq \begin{cases} 5+b; & R \equiv M_{HSV}, G \equiv m_{HSV} \\ 1-g; & R \equiv M_{HSV}, G \neq m_{HSV} \\ 1+r; & R \neq M_{HSV}, G \equiv M_{HSV}, B \equiv m_{HSV} \\ 3-b; & R \neq M_{HSV}, G \equiv M_{HSV}, B \neq m_{HSV} \\ 3+g; & R \neq M_{HSV}, G \neq M_{HSV}, B \equiv M_{HSV}, B \equiv m_{HSV} \\ 5-r; & \text{sicer} \end{cases},$$

  kjer velja

$$H(c) \coloneqq \frac{H_1(c)}{6}.$$

- Nasičenost (angl. "saturation") je definirana kot

$$S(c) \coloneqq \frac{M_{HSV}-m_{HSV}}{M_{HSV}}.$$

- Informacija o svetlosti barve je definirana kot

$$V(c) \coloneqq M_{HSV}.$$

- Barvni prostor "Luv" (natančneje CIELUV) je 3D barvni prostor, ki podpira opis vseh barv, ki jih lahko vidimo s človeškim očesom, kar

ne velja za ostale zgoraj opisane prostore. Dimenziji u in v opisujeta odtenek barve (u med zeleno in magento, v med modro in rumeno). Dimenzija L opisuje svetlost barve.

Podrobnejša definicija prostora "Luv" je kompleksna in je zato tu izpuščena [25].

2.1.2 Superpiksli

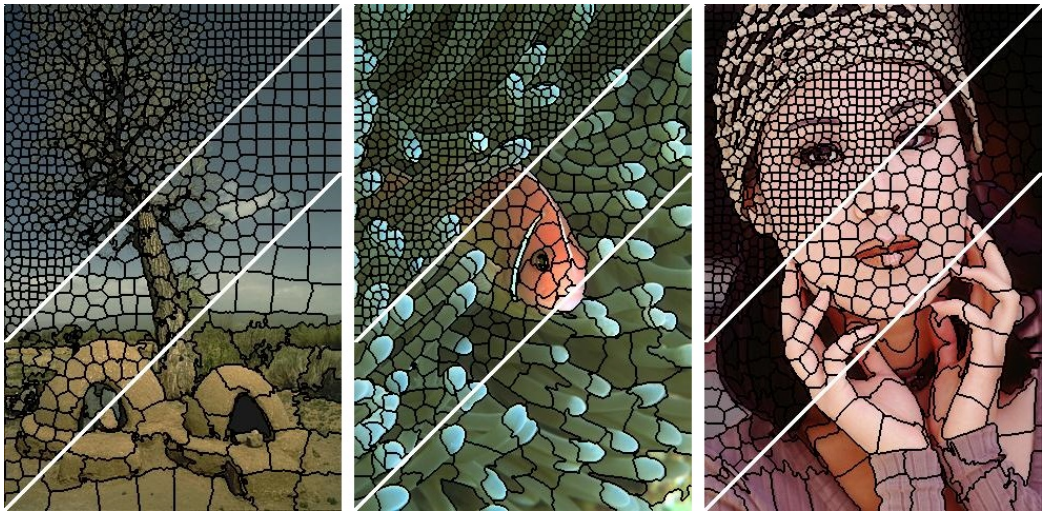
Naloga superpikselskih algoritmov je razbitje slike v kose pikslov - superpiksle [46] (glej Sliko 2.1), kjer se za vsak kos pričakuje, da se vizualno razume kot enotna regija (tipično se išče čimvišjo homogenost po barvni sestavi). Število končnih superpikslov je precej manjše od števila izvornih pikslov slike - glede na parametre najpogosteje za več redov velikosti, kar botruje občutnemu zmanjšanju zahtevnosti pri algoritmičnem obdelovanju slik. Razbitje na superpiksle deluje učinkovito za potrebe segmentacije slike (kvalificiranje slike na več delov). Nabor končnih superpikslov je možno interpretirati tudi kot povezan graf značilk slike.

Algoritem Simple Linear Iterative Clustering

Obstaja več pristopov za algoritme, ki izračunavajo superpiksle slike [30, 31, 32, 33]. Osredotočimo se na algoritem Simple Linear Iterative Clustering (SLIC) [28], ki ga odlikuje kakovostno in hitro delovanje. Algoritem prejme kot parameter poleg slike še želeno število superpikslov K.

Algoritem vsak piksel slike obravnava kot točko v petih dimenzijah, ki vsebuje lokacijo piksla na sliki (x, y) in tri komponente barve v prostoru "Luv" (L, u, v). Nato definira razdaljo med piksli d_{SLIC}, ki nagrajuje lokacijsko bližino, hkrati pa tudi podobnost barve.

V grobem algoritem SLIC najprej po vhodni sliki postavi K približno enakomerno razporejenih točk, ki služijo kot središča začetnih superpikslov. Nato algoritem ponavlja naslednje korake iteracije do konvergence (ponavadi od 4 do 10 ponavljanj):



Slika 2.1: Primeri razbitja slik na superpiksle. Za vsako od treh slik so podani trije primeri razbitij, glede na želeno velikost superpikslov, podano s parametrom. Vir: [27]

1. Vsak piksel slike se razporedi v superpiksel, ki pripada najbližjemu središču (po Evklidski razdalji).

2. Ponovno se izračunajo središča za vsak superpiksel. Središča so izračunana kot povprečja vsebovanih pikslov v prostoru $Luvxy$.

Po končanih iteracijah algoritem priredi končne superpiksle, da dobijo še lastnost povezanosti (vsak superpiksel je ena povezana gruča pikslov, brez ločenih regij).

Ob pogledu na končne rezultate delovanja algoritma SLIC (glej Sliko 2.1) lahko opazimo naslednje značilnosti, ki se izkažejo za uporabne pri praktični uporabi superpikslov:

- V veliki večini so superpiksli medsebojno podobni po velikosti (tj. širini regije, dolžini regije in številu vsebovanih pikslov).

- Superpiksli izvirajo iz približnih centrišč, ki so enakomerno mrežno porazdeljena po sliki.

- Superpiksli dosledno zaznamujejo meje med različnimi objekti na sliki, dokler se objekti dovolj razlikujejo po barvi in teksturi.

Algoritem SLIC ima časovno zahtevnost $O(n)$, kjer n označuje število pikslov. Obstaja tudi podrazličica algoritma, SLICO [29], ki ne potrebuje parametra za želeno število končnih superpikslov.

2.1.3 Mere razdalj med barvnimi histogrami

V sklopu naloge se uporablja razdalja $hi - kvadrat$. Denimo, da imamo podana barvna histograma velikosti N, $\mathbf{h_1}$ in $\mathbf{h_2}$, z vrednostmi med 0 in N:

$$\mathbf{h_1}, \mathbf{h_2} \in [0 \,..\, U]^N .$$

Razdalja $hi - kvadrat$ je definirana kot

$$\chi^2(\mathbf{h_1}, \mathbf{h_2}) \coloneqq \frac{1}{2} \sum_i \frac{(\mathbf{h_1}[i] - \mathbf{h_2}[i])^2}{(\mathbf{h_1}[i] + \mathbf{h_2}[i])},$$

kjer je $\mathbf{h}[i]$ i-ta celica histograma.

Metrika deluje na temeljih norme L^2, vendar poleg tega še pomanjša občutljivost med vrednostima, ki sta si relativno podobni (za primer - metrika je petkrat občutljivejša na razliko med vrednostima 0 in 0.2, kot pa med vrednostima 0.4 in 0.6).

2.1.4 Razbitje grafa

Razbitje slike na dve ali več regij je pogost problem na področju računalniškega vida. Podana je množica vseh značilk V na sliki, hkrati pa ocenjevalna funkcija

$$f : V \times \{0, 1\} \mapsto \mathbb{R}. \tag{2.2}$$

S pomočjo f bi lahko po značilkah neposredno izpeljali verjetnost, ali je značilka del ospredja

$$P : V \mapsto [0, 1]. \tag{2.3}$$

Vendar pa od osprednega objekta ponavadi pričakujemo, da je geometrijsko čimbolj homogen in brez samotnih značilk, nasutih po sliki, pa tudi brez posamičnih lukenj. Preprosta rešitev, s katero bi značilke razločili glede na neko mejo po P, se tako izkaže za nepopolno. Za boljšo rešitev se zatečemo k algoritmu za razbitje grafa GraphCut [39].

Za uporabo algoritma GraphCut moramo problem prirediti na utežen graf. Definirajmo nov utežen graf

$$G = (V \cup \{u, b\}, E)\,, \tag{2.4}$$

kjer dodatni točki konceptualno predstavljata izvir (u) in ponor (v). Težo povezav med u ter ostalimi točkami nastavimo na

$$w_{ux} = f(x, 1); x \in V,$$

do v pa

$$w_{vx} = f(x, 0); x \in V.$$

Točki u in v med sabo nimata povezave. Dodajmo še povezave med prostorsko sosednjimi točkami $a, b \in V$:

$$w_{ab} = g(a, b),$$

kjer $g(a, b)$ določi barvno podobnost med značilkama a in b.

Sedaj lahko problem razbitja množice značilk V prevedemo v minimalni u-v razrez grafa G. Rešitev razreza je hkrati tudi maksimalno *a posteriori* razbitje značilk na ospredje in ozadje, glede na informacije funkcij f in g.

2.1.5 Ujemanje med grafi barvnih značilk prek podobnostne matrike

Podana imamo grafa $G = (V, E)$ in $G' = (V', E')$, kjer sta V in V' množici barvnih značilk. Želimo najti ujemajoče se pare značilk $(v, v'); v \in V, v' \in V'$. Ujemanje med podanima grafoma prevedimo na ujemanje med vozlišči grafov

(v našem primeru značilkami). Ponavadi bi to predstavili kot dodelitveno matriko

$$\mathbf{Y} \in \{0,1\}^{|V|\times|V'|},$$

kjer

$$\mathbf{Y}_{ii'} = 1 \iff T_i \text{ se ujema s } T_{i'}.$$

Dodelitvena matrika naj ima vsoto po vsakem posameznem stolpcu in po vsaki vrstici kvečjemu 1, tako, da se vsako vozlišče ujema ali z enim ali z nobenim sovozliščem iz nasprotnega grafa.

Za potrebe algoritma raje definirajmo rešitev v eni dimenziji - dodelitveni vektor

$$\mathbf{z} \in \{0,1\}^{|V|\times|V'|},$$

kjer

$$\mathbf{z}_{i+i'|V|} = 1 \iff T_i \text{ se ujema s } T_{i'}.$$

Dodelitveni vektor naj vsebuje tudi na enak način preoblikovane dodatne omejitve. Naj obstaja ocenjevalna funkcija $S(z)$, ki ocenjuje primernost dodelitve z. Optimalna rešitev $\tilde{z}$ je potemtakem enaka

$$\tilde{\mathbf{z}} = \arg\max_{\mathbf{z}} S(\mathbf{z}). \tag{2.5}$$

Tako moramo poiskati čimboljšo ocenjevalno funkcijo $S(z)$ ter zatem še ustrezno optimizacijsko metodo za iskanje $\tilde{\mathbf{z}}$. Naivna inačica $S(z)$ bi nagrajevala le podobnost barvnih izgledov značilk. Sestavimo podobnostno matriko

$$\mathbf{A} \in \mathbb{R}^{n_P \times n_Q},$$

kjer elementi $\mathbf{A}_{ii'}$ predstavljajo podobnost barvnega izgleda med vozliščema T_i in $T_{i'}$, glede na neko razdaljo med barvnimi histogrami ρ_{col}.

$$\mathbf{A}_{ii'} = \rho_{col}(\mathbf{h_i}, \mathbf{h_{i'}})$$

Zopet prevedimo rešitev v enodimenzionalni prostor z vektorjem

$$\mathbf{b} \in \mathbb{R}^{|P|\times|Q|}.$$

Sedaj lahko izračunamo kandidatno rešitev

$$S_0(\mathbf{z}) = \mathbf{z}^T\mathbf{b}.$$

Barvni histogrami po značilkah so si med seboj lahko zelo podobni po obširnih območjih na sliki. Poleg tega je območij z enakovredno barvno predstavo pogosto tudi več. Ker z omenjeno rešitvijo upoštevamo le podobnost barv, bi optimalni rezultat lahko vseboval tudi nepravilna ujemanja med značilkami. Za vsako značilko bi si intuitivno želeli ne le, da ohrani približno barvno sestavo in pozicijo, temveč, da hkrati obdrži tudi nabor čimveč sosednih značilk. Prevedeno v teorijo grafov to pomeni, da želimo ohraniti tudi povezave vsake točke. Za nove zahteve bo potrebno sestaviti bolj zapleteno ocenjevalno funkcijo. Dosedaj smo maksimirali $\mathbf{z}$ glede na dodatni vektor $\mathbf{b}$, ki je za vsak element $\mathbf{z}_{i+i'|V|}$ našel podobnost glede na faktor, podan v elementu $\mathbf{b}_{i+i'|\mathrm{V}|}$, ki predstavlja barvno podobnost. Po novem bi si za vsak element iz $\mathbf{z}$ izračunali tako barvno podobnost, kot tudi barvno podobnost sosedov, za podani značilki T_i in $T_{i'}$.

Definirajmo novo matriko

$$\mathbf{C} \in \mathbb{R}^{(|P||Q|)\times(|P||Q|)},$$

ki je sestavljena iz

$$\mathbf{C} = \mathbf{C^{part}} + \mathbf{C^{con}}.$$

$\mathbf{C^{part}}$ naj bo diagonalna matrika, kjer po diagonali potekajo po vrsti vrednosti, identične zgoraj definiranemu vektorju $\mathbf{b}$. Če primerjamo značilki i in i', dobimo indeks

$$i'' \coloneqq (i-1)n_V + i'.$$

Za tak indeks i'' torej velja:

$$\mathbf{C^{part}}_{i'',i''} \coloneqq \mathbf{b}_{i''}$$

$\mathbf{C^{con}}$ naj bo matrika, ki zapolni vse vrednosti, razen diagonale. Naj obstajata točki $i, j \in G$ in točki $i', j' \in G'$. Želimo preveriti podobnost med relacijo $\vec{ij}$

in relacijo $i'\vec{j}'$. Pozicija te vrednosti na matriki izhaja iz indeksov

$$i'' \coloneqq (i-1)n_V + i'$$

ter

$$j'' \coloneqq (j-1)n_V + j'.$$

Potem velja:

$$\mathbf{C^{con}}_{i'',j''} \coloneqq \rho_{\text{con}}(e_{ij}, e_{i'j'}),$$

kjer je ρ_{con} neka razdalja med povezavama e_{ij} in $e_{i'j'}$. Iz tako dobljene matrike C izgotovimo ocenjevalno funkcijo:

$$S_1(\mathbf{z}) = \mathbf{z}^T\mathbf{C}\mathbf{z} \tag{2.6}$$

Glede na izbiro funkcij ρ_{col} in ρ_{con} bo $S_1(\mathbf{z})$ ustrezno ocenila primernost ujemanja $\mathbf{z}$.

Spektralno ujemanje

Želimo rešiti optimizacijski problem iz enačbe (2.5), kjer je $S(\mathbf{z})$ predstavljen v enačbi (2.6). Tu uporabimo spektralno metodo [34], ki ponuja hiter izračun rešitve, kar ustreza potrebam algoritma. Problem ujemanja grafov glede na podobnostno matriko prevedemo na iskanje najvišjih lastnih vektorjev matrike. Soočamo se z optimizacijskim problemom:

$$\tilde{\mathbf{z}} = \arg\max_{\mathbf{z}} \left(\mathbf{z}^T\mathbf{C}\mathbf{z}\right) \tag{2.7}$$

Formulo pretvorimo na tak način, da hkrati opisuje formulo za Rayleighov kvocient:

$$\tilde{\mathbf{z}} = \arg\max_{\mathbf{z}} \left(\frac{\mathbf{z}^T\mathbf{C}\mathbf{z}}{\mathbf{z}^T\mathbf{z}}\right);\ \mathbf{z}^T\mathbf{z} \equiv 1 \tag{2.8}$$

Pod predpostavko, da je matrika $\mathbf{C}$ hermitska, je dominantna lastna vrednost matrike enaka optimalni rešitvi enačbe (2.8). Iskanje $\tilde{\mathbf{z}}$ je torej ekvivalentno iskanju dominantnega lastnega vektorja $\mathbf{z_1}$ matrike $\mathbf{C}$. Pod predpostavko, da je matrika $\mathbf{C}$ nenegativna in simetrična, je dominantni lastni vektor $\mathbf{z_1}$ izračunljiv ter nenegativen. $\mathbf{z_1}$ torej vsebuje vrednosti v intervalu

$[0, 1]$. Manjka nam še omejitev optimalne rešitve $\tilde{\mathbf{z}}$ na diskretni vrednosti 0 in 1:

$$\tilde{\mathbf{z}} \in \{0, 1\}^{|V||V'|}.$$

Vsako vrednost znotraj $\mathbf{z_1}$ lahko interpretiramo kot zaupanje v ujemanje $U_{ii'}$ [34]. Zato se odločimo le za ujemanja znotraj $\mathbf{z_1}$, ki presegajo nek prag. Tiste vrednosti znotraj $\mathbf{z_1}$, ki presegajo določen prag, določimo v $\tilde{\mathbf{z}}$ kot 1, ostale pa kot 0. Podrobnejši opis algoritma za rešitev enačbe (2.8) je obrazložen v viru [34].

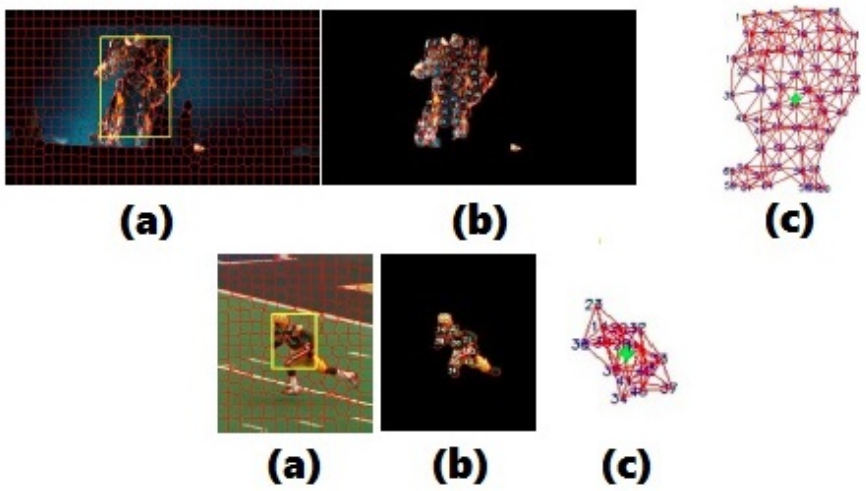
2.2 Kratek opis sledilnika DGT

Cilj vizualnega sledilnika je zaporedno določanje lokacije sledilnega objekta iz podanih zaporednih slik videa. Sledi grob oris glavnih korakov sledilnika, zatem pa še poglavje s podrobno razlago korakov. V pomoč določitvi sledilnega objekta algoritem sproti vsako sliko najprej razbije na številne drobne kose oz. regije, nakar poskuša čimbolje razdeliti nabor regij na tiste iz ospredja in tiste iz ozadja. Regije ospredja nato združi v skupno strukturo – vizualne analize vsake ospredne regije ter njihove medsebojne povezave se združijo v neusmerjeni dinamični graf G. Sledenje objektu se tako prevede na sledenje grafu slikovnih regij, katerih namen je sestaviti mozaik sledilnega objekta (Slika 2.2).

Sproti pri vsaki naslednji sliki posnetka se tako najprej soočimo z nalogo razvrstitve regij slike na ospredje in ozadje. Ozirajoč se na končni dinamični (neusmerjeni) graf prejšnje slike G_0 je potrebno iz sprotne slike sestaviti naslednji dinamični graf G_1. V iteraciji po slikah videoposnetka torej za vsako sliko v grobem izvedemo naslednje korake:

1. Sliko najprej razbijemo na mozaik slikovnih regij s pomočjo superpikselskega algoritma (glej Poglavje 2.1.2). Slikovne regije lahko v nadaljevanju (za lažje razumevanje) okličemo kot superpiksle.

2. Mozaik želimo razdeliti na dve gruči - ospredje in ozadje. Pri tem se

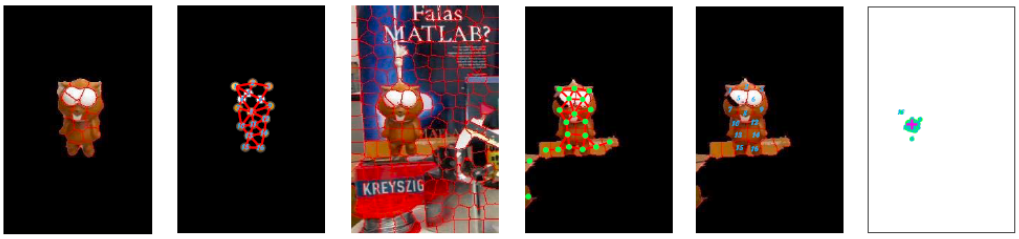


Slika 2.2: Dva primera poteka sledilnika DGT. Na sliki (a) je prikazano razbitje slike na razmeroma homogene slikovne regije. Na sliki (b) je prikazan nabor slikovnih regij, ki jih algoritem označi kot del ospredja. Na sliki (c) je prikazan končni graf, ki predstavlja sledilni objekt. Vir slik: [35]

ozremo na barvno sestavo ospredij in ozadij preteklih slik. Razdelitev je opravljena s pomočjo metode podpornih vektorjev (angl. "Support Vector Machine", SVM) ter še izboljšana z uporabo Markovskega slučajnega polja [60] (angl. "Markov Random Field", MRF).

3. Ustvarimo kandidatni neusmerjeni graf G' iz osprednih superpikslov (kot vozlišč) in njihovih internih medsebojnih relacij (kot povezav). Medsebojne relacije dodamo le med bližnjimi, povečini dotikajočimi se superpiksli.

4. Med zadnjim končnim dinamičnim grafom G_0 in kandidatnim grafom G' izračunamo optimalno ujemanje, ki iz grafa G' izlušči najbolj primerne superpiksle za ospredje. Ujemanje izračunamo na podlagi podobnostne matrike med grafoma - s pomočjo spektralne analize algoritem izlušči poglavitno skupino ujemanj, tj. ujemanj, ki so statistično najbolj verjetna.



Slika 2.3: Osnovni prikaz delovanja sledilnika. V (a) je razvidna uvodna izbira sledilnega objekta glede na začetni okvirni pravokotnik. V (b) je pokazan skonstruiran graf objekta. (c) predstavlja razbitje naslednje slike na superpiksle, (d) izbrane kandidatne superpiksle ter kandidatni graf, (e) končni izračunan graf objekta po obdelavi nove slike, (f) pa glasovalne točke za izbiro končne izračunane lokacije objekta. Vir: [41]

5. Iz grafov G_0 in G' na podlagi ujemanj sestavimo končni graf G_1 na novi sliki, ki vsebuje vse superpiksle, ki predstavljajo sledilni objekt.

6. Ker se za ocenjevanje sledilnih algoritmov navadno zahtevajo enostavnejši opisi sledilnega objekta (kot npr. očrtan pravokotnik v VOT [26]), na koncu izračunamo še čimbolj reprezentativen očrtane pravokotnik.

2.3 Podroben opis sledilnika DGT

Kot je opisano v Poglavju 2.2, sledilnik DGT sestoji iz več ločenih zaporednih podnalog. V naslednjih odsekih je podrobno opisano delovanje vsake podnaloge.

2.3.1 Konstrukcija kandidatnega grafa

Ob vsaki sprotni sliki je najprej potrebno sestaviti kandidatni graf, ki poskusi opisati strukturo sledilnega objekta v sprotni sliki brez dodatne informacije o strukturi objekta iz prejšnjih slik. Sestaviti želimo graf, ki vsebuje čim manj

artefaktov, ki bi spadali v ozadje slike. Iz sprotne slike upoštevamo le okvir, postavljen v bližnji okolici lokacije objekta na prejšnji sliki.

Izbira slikovnih regij iz ospredja

Najprej se izvorno sliko razbije na množico slikovnih regij T_p s pomočjo superpikselskega algoritma SLIC [27], kot je razvidno na Sliki 2.3(c). Superpiksle želimo razdeliti na ospredje in ozadje. Najprej definirajmo energijsko funkcijo Markovskega slučajnega polja:

$$E(B) = \sum_{p \in S} D_p(b_p) + \sum_{p,q \in U} V_{p,q}(b_p, b_q). \tag{2.9}$$

Tu je $B = \{b_p \mid b_p \in \{0,1\}, p \in S\}$ ena od vseh kombinatorično možnih razdelitev superpikslov na ospredje in ozadje:

$$b_p = 1 \iff \text{superpiksel } t_p \text{ je postavljen v ospredje.}$$

S je množica vseh superpikslov v ciljnem okviru dobljene slike, U pa množica vseh parov sosednjih (t.j. dotikajočih se) superpikslov. $D_p(b_p)$ predstavlja unarni potencial superpiksla b_p, $V_{p,q}(b_p, b_q)$ pa doda še medsebojni potencial sosednjih superpikslov t_p in t_q.

Potrebno je najti razdelitev superpikslov B, ki minimira rezultat energijske funkcije (2.9). Za rešitev optimizacijskega problema je uporabljen algoritem *GraphCut* [38]. Za unarni potencial $D_p(b_p)$ se uporablja utežen seštevek

$$D_p(b_p) = \lambda_{\text{kum}} D_p^{KUM}(b_p) + D_p^{SVM}(b_p). \tag{2.10}$$

$D_p^{KUM}(b_p)$ je izračunana verjetnost, ali superpiksel t_p spada v ospredje oziroma v ozadje, glede na celotno kumulativno zgodovino barvne sestave objekta v videoposnetku:

$$D_p^{KUM}(b_p) = \begin{cases} -\frac{1}{N_p} \sum_{i=1}^{N_p} logP\left(c_i \mid \mathbf{h_1}\right); & b_p \equiv 1 \\ -\frac{1}{N_p} \sum_{i=1}^{N_p} logP\left(c_i \mid \mathbf{h_0}\right); & b_p \equiv 0 \end{cases}.$$

Tu $\mathbf{h_0}$ in $\mathbf{h_1}$ označujeta histograme ozadja in ospredja, izračunane iz vseh dosedajšnjih slik pred trenutno, c_i je barvna vrednost piksla i, N_p pa število

pikslov v superpikslu t_p. $P(C_i \mid H)$ je verjetnost, da se barva C_i nahaja znotraj histograma H.

Drugi del seštevka znotraj enačbe (2.10) se nanaša na precej bolj dinamično komponento za klasificiranje glede na barvno sestavo - uporablja se SVM, naučen iz preteklih ospredij in ozadij,

$$D_p^{SVM}(b_p) = \begin{cases} \lambda_{\text{svm}}\widehat{SVM}(\mathbf{h_p}); & \widehat{SVM}(\mathbf{h_p}) \geq 0, b_p \equiv 1 \\ 1 - \lambda_{\text{svm}}\widehat{SVM}(\mathbf{h_p}); & \widehat{SVM}(\mathbf{h_p}) \geq 0, b_p \equiv 0 \\ \widehat{SVM}(\mathbf{h_p}); & \widehat{SVM}(\mathbf{h_p}) < 0, b_p \equiv 1 \\ 1 - \widehat{SVM}(\mathbf{h_p}); & \widehat{SVM}(\mathbf{h_p}) < 0, b_p \equiv 0 \end{cases}$$

kjer $\mathbf{h_p}$ označuje histogram superpiksla t_p v barvnem prostoru HSV. λ_{svm} je parameter, s pomočjo katerega lahko povečamo moč osprednih barv - zaradi potreb algoritma si namreč prej želimo izvleci preveč "osprednih" superpikslov, kot pa premalo. V nadaljevanju algoritma postojijo še drugi mehanizmi, ki kasneje zmanjšajo končno število osprednih superpikslov. Klasifikator SVM se uči sproti preko učnega algoritma LASVM (glej Poglavje 2.1.2).

Medsebojni potencial superpikslov $V_{p,q}(b_p, b_q)$ nastavimo tako, da je enaka barvni podobnosti med superpiksloma

$$V_{p,q}(b_p, b_q) = e^{-\rho_{\text{col}}(\mathbf{h_p}, \mathbf{h_q})},$$

kjer je $\rho_{\text{col}}(\cdot, \cdot)$ definirana kot standardna razdalja hi-kvadrat med histogramoma.

S tako definiranim medsebojnim potencialom dobi potencialna funkcija težnjo, da sosednje superpiksle s podobno barvo poskusi obdržati skupaj v ospredju ali ozadju.

Postopek izbire osprednih slikovnih regij sklenemo z uporabo algoritma GraphCut [38] nad potencialno funkcijo iz enačbe (2.9), s čimer dobimo množico kandidatnih osprednih superpikslov B'. Ospredne superpiksle definirajmo kot množico T'_p, kjer velja:

$$t'_p \in T'_p \iff b'_p = 1.$$

Slika 2.4: Primer povezanosti osprednih superpikslov na sledilnem objektu. V (a) je viden sledilni objekt iz originalne slike, v (b) pa so izrisani ospredni, medsebojno povezani superpiksli.

Konstrukcija grafa

Iz množice kandidatnih osprednih superpikslov T'_p definirajmo graf $G' = (V', E')$, kjer dodamo le povezave med geometrijsko bližnjimi superpiksli.

$$V' = T'_p E' = \{e_{i'j'};\ ||\ t_{p'} - t_{r'}\ ||_2 \leq \epsilon\}$$

Spomnimo se, da imajo superpiksli, pripravljeni s pomočjo algoritma SLIC, standardno količino vsebovanih pikslov N_t. Ker so superpiksli razporejeni po kvadratni mreži, je njihova povprečna dolžina (in širina) enaka $d = \sqrt{N_t}$. Potem nastavimo $\epsilon = d$, s čimer poskrbimo, da je vsak obravnavan superpiksel potencialno povezan z osmimi geometrijskimi sosedi (vodoravno, navpično in po obeh diagonalah). Primer povezav lahko vidimo na Sliki 2.4.

2.3.2 Ujemanje dinamičnega in kandidatnega grafa

Za dobro delovanje algoritma je potrebno kvalitetno ujemanje med dosedajšnjim grafom $G_0 = (V, E)$ in novim kandidatnim grafom $G' = (V', E')$, saj prek ujemanja dobimo informacijo o tem, kateri deli objekta iz prejšnjih slik so obstali tudi na novi sliki.

Konstrukcija podobnostne matrike

Skonstruirati želimo podobnostno matriko $\mathbf{C}$, in sicer na način, opisan v Poglavju 2.1.5. Matrika $\mathbf{C}$ je sestavljena iz dveh matrik: $\mathbf{C} = \mathbf{C^{part}} + \mathbf{C^{con}}$, kjer je element diagonalne matrike $\mathbf{C^{part}}$ definiran kot podobnost značilk T_i in $T_{i'}$: $\mathbf{C^{part}}_{\mathbf{i''},\mathbf{i''}} := \omega^{part}(T_i, T_{i'})$. Podobnost značilk $\omega^{part}(\cdot,\cdot)$ izračunamo glede na razdaljo med barvnimi histogrami:

$$\omega^{part}(T_i, T_{i'}) = e^{-\frac{1}{2}\rho_{\mathrm{col}}(\mathbf{h_i},\mathbf{h_{i'}})^2},$$

kjer je ρ_{col} zopet postavljena kot razdalja hi-kvadrat. Matrika $\mathbf{C^{con}}$ obravnava ujemanje med dvema povezavama značilk, $e_{ij} \in E$ in $e'_{i'j'} \in E'$. Hkrati torej ujemamo para značilk $(T_i, T_{i'})$ in $(T_j, T_{j'})$, glede na podobnost njunih povezav: $\mathbf{C^{con}}_{\mathbf{i''},\mathbf{j''}} := \omega^{con}(T_i, T_j, T_{i'}, T_{j'})$. Podobnost povezav izračunamo kot

$$\omega^{con}(T_i, T_j, T_{i'}, T_{j'}) = e^{-\frac{d^2}{2}\rho_{\mathrm{con}}\left(e_{ij},e_{i'j'}\right)^2},$$

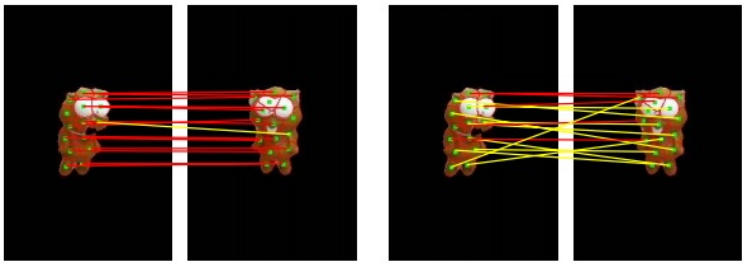
kjer je ρ_{con} definirana kot norma

$$\rho_{\mathrm{con}}(e_{ij}, e_{i'j'}) := ||\ (l_i - l_j) - (l_{i'} - l_{j'})\ ||_2\ ,$$

kjer l_i označuje lokacijo superpiksla t_i.

Funkcija $\omega^{part}(T_i, T_{i'})$ ocenjuje primernost ujemanja $(T_i, T_{i'})$ glede na barvno sestavo ujemajočih se superpikslov, $\omega^{con}(T_i, T_j, T_{i'}, T_{j'})$ pa primernost dveh ujemanj $(T_i, T_{i'})$ in $(T_j, T_{j'})$ glede na podobnost njunih notranjih povezav. $\omega^{con}(\cdot,\cdot,\cdot,\cdot)$ deluje kot močna geometrijska omejitev pri ujemanju grafov - če sta bila dva superpiksla T_i in T_j iz prejšnjega grafa sosednja, dobi ujemalna funkcija močno incentivo, da najde kandidatna superpiksla $T_{i'}$ in $T_{j'}$, ki poleg podobne barvne sestave vsebujeta tudi podoben par povezav ter sta geometrijsko med seboj na podobnih lokacijah.

Končne dimenzije matrike $\mathbf{C}$ so $(|V||V'|) \times (|V||V'|)$. Dimenzije so preobsežne za praktično reševanje problema prek enačb (2.5) in (2.6). Poleg tega je matrika še vedno preveč splošna in dopušča preveč možnih rešitev. Matriko $\mathbf{C}$ zato najprej še razredčimo - poskusimo nastaviti čimveč vrednosti matrike $\mathbf{C}_{\mathbf{i''},\mathbf{j''}}$ na 0, če lahko za takšna ujemanja z gotovostjo zatrdimo, da bi



Slika 2.5: Primerjava dveh načinov ujemanja dinamičnega in kandidatnega grafa glede na matriko **C**. Na sliki (a) je prikazana rešitev, pridobljena s pomočjo spektralnega ujemanja. Na sliki (b) je prikazana rešitev, pridobljena s pomočjo pohlepne strategije, ki po vrsti ujema pare značilk z najvišjimi ocenami ujemanja. Rdeče črte prikazujejo pravilna ujemanja, rumene pa napačna. Vir: [35]

bila tako ali tako nesmiselna. Podpremo le ujemanja med superpiksli, katerih medsebojna razdalja je pod določeno zgornjo vrednostjo: $|| \ l_i - l_{i'} \ ||_2 \leq d\delta$, kjer je $\delta = \frac{3}{2}$, d pa je enaka povprečni širini oz. dolžini superpiksla (glej Poglavje 2.3.1).

Dodamo še ostro omejitev ujemanja med superpiksli glede na barvno podobnost. Dopustimo le ujemanja, kjer velja $\omega^{part}(T_i, T_{i'}) \geq 0.3$. Zatem dopustimo za vsako značilko iz prejšnjega grafa le 5 najpodobnejših kandidatnih značilk (glede na ω^{part}). Omejimo tudi ujemanje med povezavami - morebitno razliko nagiba povezave omejimo na kvečjemu 60°. Odpravimo tudi povezave, kjer $\rho_{\text{con}}(e_{ij}, e_{i'j'}) \geq d$.

Spektralno ujemanje

Glede na podano in ustrezno razredčeno podobnostno matriko **C** sestavimo funkcijo $S(\mathbf{z})$ glede na (2.6). V nadaljevanju se lotimo optimizacijskega problema iz (2.5), za kar uporabimo rešitev, opisano v Poglavju 2.1.5. Prek primerjave s pohlepnim ujemanjem lahko na Sliki 2.5 vidimo, da je uporaba spektralnega ujemanja ključnega pomena za uspeh algoritma. S pomočjo

optimalnega ujemanja smo torej dobili nabor superpikslov T' iz kandidatnega grafa G', ki smo jih zaznali kot "naslednike" superpikslov iz prejšnjega dinamičnega grafa G_0.

2.3.3 Sestava končnega grafa sprotne strukture objekta

Za končni preračunan dinamični graf G_1 v iteraciji želimo, da vsebuje vse superpiksle, ki naj bi predstavljali sledilni objekt na sliki. Logika posodobitve dinamičnega grafa vsakemu vozlišču predpisuje eno od treh možnih stanj:

- **Začetek**: Značilka t'_i (t.j. eden od superpikslov iz kandidatnega grafa G') je v začetnem stanju, če ni v množici "naslednikov" T'. Da dinamični graf ne izpade preveč gost, naj bo značilka tudi dovolj daleč od ostalih potencialnih značilk: $|| \ t'_i - t'_j \ ||_2 > 0.35d \ \forall t'_j \in T' \backslash \{t'_i\}$.

- **Stabilnost**: Značilka t'_i je stabilna, če je v množici T'. Poleg tega kot stabilne upoštevamo tudi značilke, ki niso vsebovane v optimalnem ujemanju, a lahko zanje uspešno najdemo predhodnike iz G_1 s podobnim izgledom in približno isto lokacijo: $\rho_{\text{col}}(\mathbf{h}_p, \mathbf{h}_q) > 0.4$ in $|| \ l_i - l_{i'} \ ||_2 < 0.25d$.

- **Zaključek**: Značilka je v zaključnem stanju, če ni bila stabilna že zadnjih 5 slik.

Ustvarimo nov dinamični graf G_1, ki je kopija prejšnjega grafa G_0. Ko ima vsaka značilka določeno stanje, se iz dinamičnega grafa G_1 izbrišejo značilke z zaključnim stanjem ter dodajo značilke z začetnim stanjem (ki so bile sicer znotraj kandidatnega grafa G'). V G_1 se dodajo tudi ustrezne povezave med novimi in starimi značilkami, glede na njihovo geometrijsko bližino. Tako smo dobili končni graf G_1, ki nam bo v pomoč pri računanju lokacije objekta v naslednji sliki.

Poglavje 3

Predlagane izboljšave sledilnika DGT

Sledilnik DGT [35] se je znotraj VOT2014 [36] izkazal kot eden najnatančnejših, a je imel težave v primerih spremembe osvetlitve sledilnega objekta. Izboljšave delovanja so bile zato usmerjene predvsem napram izboljšavam pri teh problematičnih odstopanjih.

3.1 Izboljšava uporabe klasifikatorja SVM

V Poglavju 2.3.1 je opisana uporaba klasifikatorja SVM, s pomočjo katerega algoritem pridobi sposobnost zanesljivega prilagajanja klasificiranj barv na ospredje in ozadje. Za učenje SVM uporabi algoritem LASVM [40]. LASVM učne podatke pridobi na koncu vsake iteracije za vsak piksel posebej. Obravnavajo se le piksli znotraj ciljnega okvira, ki obsega bližnjo okolico lokacije objekta.

Ob podrobnem pregledu implementacije algoritma se je pokazalo, da je količina učnih podatkov skozi vrsto slik preveč obremenila klasifikator SVM. Zahtevan čas za klasifikacijo superpiksla se je v zaporedju približno petdesetih slik postopoma toliko povečal, da je zahteval že več kot polovico skupnega časa celotnega algoritma DGT. V izogib tej težavi algoritem DGT resetira

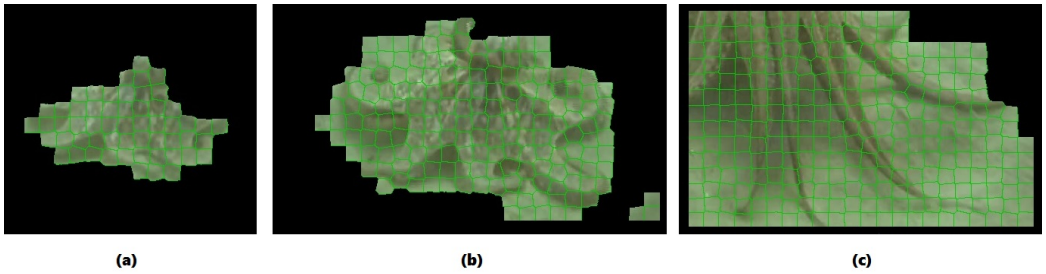
Slika 3.1: Primer treh zaporednih slik, kjer se osvetlitev objekta hitro spreminja.

celoten klasifikator SVM na vsakih 20 slik. Posledično je klasifikator na vsaki dvajseti sliki postal za nekaj sprotnih slik precej nezanesljiv. Algoritem v večini primerov pri mnogokratnikih 20. slike ni imel težav, saj se sledilni objekti le redko močno spremenijo po barvni sestavi. Vseeno je bilo možno najti primere posnetkov, kjer se je osvetljenost objekta močno spreminjala ravno pri občutljivih slikah (glej Sliko 3.1), s čimer so se zamenjale tudi ospredne barve objekta. V takšnih primerih je algoritem popolnoma izgubil sled za objektom.

Sistem klasifikacije superpikslov na ospredje in ozadje je bil v okviru diplomske naloge prirejen. Enojni klasifikator SVM se je zamenjalo z dvema ločenima klasifikatorjema SVM, ki se izmenjujeta. Vsak klasifikator se najprej inicializira iz 15 sprotnih slik, nato pa naslednjih 15 slik (poleg učenja) tudi klasificira superpiksle. Po 30 slikah se klasifikator resetira. Klasifikatorja se izmenjujeta, tako, da je eden od njiju vedno v stanju učenja, drugi pa v stanju klasificiranja. Na ta način se je odpravila občutljivost na hude spremembe osvetljenosti objekta v določenih primerih.

3.2 Robustna detekcija ospredja

V preizkusih algoritma smo opazili, da sledilnik DGT pogosto ne uspe najti sledilnega objekta niti v uvodni sliki, temveč kot ozadje označi kar celotno sliko. Poleg tega ima algoritem tendenco, da skozi čas začenja nekatere dele objekta napačno ocenjevati kot ozadje, posledica česar je, da se skozi dolgo zaporedje slik čedalje večji delež sledilnega objekta označi kot del ozadja.



Slika 3.2: Primer postopnega poslabšanja razločevanja med ospredjem in ozadjem slike zaradi podobnosti barv. Na vseh treh slikah je prikazan le tisti predel slike, ki ga algoritem DGT poroča kot ospredje.

Občasno se zgodi tudi nasprotna situacija, kjer sledilnik začne pomotoma povečevati ospredje (glej primer na Sliki 3.2).

Vzrok problema tiči v delu algoritma, opisanem v Poglavju 2.3.1. Za glasovanje, ali superpiksel spada v ospredje ali v ozadje glede na pretekle slike, uporabljamo utežen seštevek funkcij $D_p^{KUM}(b_p)$ in $D_p^{SVM}(b_p)$. Vendar pa nobena od teh dveh funkcij nima vpeljanih varovalk, ki bi poskrbele, da je vedno vsaj nek delež superpikslov izglasovan kot del ospredja. Obe funkciji priredimo tako, da lahko s parametrom γ povišamo delež superpikslov, ki bi bili izglasovani kot del ospredja:

$$D_p^{KUM'}(b_p, \gamma) = \begin{cases} -\frac{\gamma}{N_p} \sum_{i=1}^{N_p} logP\left(C_i \mid H_1\right); & b_p \equiv 1 \\ -\frac{1}{N_p} \sum_{i=1}^{N_p} logP\left(C_i \mid H_0\right); & b_p \equiv 0 \end{cases}$$

$$D_p^{SVM'}(b_p, \gamma) = \begin{cases} \gamma \lambda_{\text{svm}} \widehat{SVM}(f_p); & \widehat{SVM}(f_p) \geq 0, b_p \equiv 1 \\ \gamma \left(1 - \lambda_{\text{svm}}\right) \widehat{SVM}(f_p); & \widehat{SVM}(f_p) \geq 0, b_p \equiv 0 \\ \widehat{SVM}(f_p); & \widehat{SVM}(f_p) < 0, b_p \equiv 1 \\ 1 - \widehat{SVM}(f_p); & \widehat{SVM}(f_p) < 0, b_p \equiv 0 \end{cases}$$

Sedaj lahko po dokončani razdelitvi nabora superpikslov na ospredje in ozadje preverimo, če je v bilo v ospredje dodeljenih dovolj elementov. V primeru premajhnega števila poskusimo znova s povišanim parametrom γ.

Sledilnik v kasnejšem koraku ujema ustvarjen kandidatni graf ospredja z dolgotrajnim dinamičnim grafom; tam s pomočjo dodatnih omejitev pomaga odstraniti superpiksle, ki so v ospredje prišli ponesreči, dokler se jih ne pojavi preveč v predolgem zaporedju slik. Zato bi bilo idealno v ospredje prinesti nekaj več objektov, kot pa jih sledilni objekt dejansko vsebuje. Za objekt lahko na naslednji sliki predpostavimo, da vsebuje vsaj nek delež α števila superpikslov, ki jih je vseboval v prejšnji sliki.

Pri iterativnem povečevanju parametra γ je potrebno biti pazljiv. Če parameter v enem koraku preveč povečamo, bi se kot ospredje lahko označilo preveč novih superpikslov. Zato postavimo zgornjo mejo - število superpikslov v ospredju lahko doseže največji delež β od števila vseh obravnavanih superpikslov znotraj ciljnega okvirja. Če se izkaže, da je delež osprednih superpikslov presegel vrednost β, začnemo parameter γ polagoma manjšati ($\gamma \leftarrow 0.97\gamma$), dokler se pogoj ne uresniči.

Poglavje 4

Eksperimentalna analiza

4.1 Implementacija in parametri

Izvorni algoritem DGT je napisan v programskem jeziku C++ (izvorna koda je dosegljiva na spletu [41]). Implementacija sprememb je bila narejena v razvijalskem okolju Visual Studio. Analiza učinkovitosti sprememb algoritma se je izvajala s pomočjo v ta namen pripravljene knjižnice za MATLAB, narejene za potrebe VOT2015.

Izbrani so bili naslednji parametri:

- Utež kumulativne klasifikacije ospredja/ozadja v energijski funkciji je $\lambda_{\text{kum}} = 0.1$ - tako je klasifikacija superpikslov na ospredje/ozadje odvisna predvsem od dinamične komponente s klasifikatorjem SVM. Le v primeru neodločne klasifikacije pride v poštev tudi verjetnost glede na celotno kumulativno zgodovino barvnih sestav ospredja/ozadja. Sledilnik je močno občutljiv na ta parameter - ob zvišanju bi postal slabši v primerih spremembe osvetlitve scene, ob znižanju pa bi po daljši zakritosti dela sledilnega objekta le-tega ob ponovni pojavitvi večkrat izgubil. Glede na to, da ima sledilnik pogoste težave v obeh primerih, bi bilo utež λ_{kum} nespametno spreminjati za več kot nekaj decimalk.
- Utež klasificiranega ospredja klasifikatorja SVM je $\lambda_{\text{svm}} = 15$. Želimo namreč obdržati čimveč pravilnih superpikslov iz ospredja, tudi, če zato

pade v ospredje nekaj dodatnih nepravilnih superpikslov. Problemi izvornega algoritma se pogosto izražajo zaradi točne izbire te uteži - ob previsoki vrednosti bi algoritem hitro kot ospredje vzel celotno sliko, ob prenizki pa bi kaj kmalu izgubil celotno ospredje. Optimalna izbira uteži je močno odvisna od primera do primera. Po implementaciji izboljšave za robustnejšo detekcijo ospredja je algoritem malce odpornejši na manjše spremembe uteži.

- Zgornjo mejo β (v sklopu predloga robustne detekcije ospredja) postavimo na $\beta = 0.6$. Če bi število superpikslov preseglo to mejo pri pravilnem delovanju, bi se moral objekt povečati za skoraj sedemkrat, a se algoritmu precej večkrat zgodi, da do te meje pride zaradi napačnega delovanja.

4.2 Protokol evaluacije in mere

Za evaluacijo algoritma je bil uporabljen sistem za analizo sledilnikov iz VOT2015 [44]. Nabor uporabljenih sekvenc v VOT2015 je sestavljen iz zbirk OTB [47], ALOV [48], PTR [51], nekaj pa tudi iz drugih virov. Protokol iz VOT2015 poda algoritmom očrtan pravokotnik sledilnega objekta na uvodni sliki, nato pa zahteva očrtan pravokotnik na vseh preostalih slikah po vrsti. Če sledilni algoritem izgubi sled za objektom, se na naslednji sliki posnetka ponovno inicializira. Uporabljene so bile naslednje mere:

- Natančnost izmeri, kako dobro je sledilni algoritem po slikah poročal o očrtan pravokotniku objekta.

- Robustnost izmeri, kolikokrat je sledilni algoritem skozi posnetek izgubil sled za objektom. Na neki sliki se sled šteje za izgubljeno, ko se poročan očrtan pravokotnik niti ne dotika več pravilne rešitve.

Na voljo je tudi dodatna analiza uspešnosti sledilnikov v primerih, ko se sprotno sledenje objektu oteži zaradi specifičnih vizualnih sprememb (sprememba osvetlitve, sprememba velikosti objekta, delna ali polna zakritost

Tabela 4.1: Različne variante algoritma DGT. DGT_{ORIG} je izvorna verzija algoritma, DGT_{2SVM} in $\text{DGT}_{\text{SESTAVLJEN}}$ pa izpeljanki.

Naziv sledilnika	Izboljšava uporabe klasifikatorja SVM (Poglavje 3.1)	Robustna detekcija ospredja (Poglavje 3.2)
DGT_{ORIG}	✗	✗
DGT_{2SVM}	✓	✗
$\text{DGT}_{\text{SESTAVLJEN}}$	✓	✓

objekta, premikanje objekta, premikanje kamere), in pa analiza uspešnosti v primerih sledenja brez dodatne težavnosti. VOT ima namreč pripravljeno informacijo o intervalih znotraj videoposnetkov, kjer nastopijo vizualne spremembe, po vsaki vrsti spremembe posebej. Posledično je bilo možno tudi analitično preveriti uspešnost predlaganih izboljšav, predvsem na področju spremembe osvetlitve objekta.

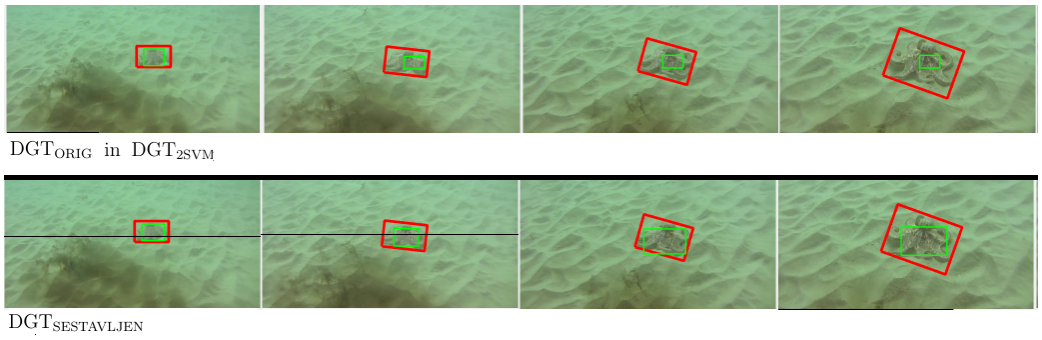
4.3 Analiza uspešnosti predlaganih izboljšav

Analiza je narejena nad tremi različnimi implementacijami algoritma DGT (glej Tabelo 4.1).

4.3.1 Kvalitativna analiza

Problemi sledenja objekta v primerih spremembe osvetlitve se lahko prikažejo na različne načine. Na Sliki 4.1 je prikazan primer, kjer se osvetlitev spreminja zelo hitro. V tem specifičem primeru lahko vidimo, da izboljšava uporabe klasifikatorja SVM uspešno izboljša zanesljivost sledilnika. Po drugi strani na Sliki 4.2 naletimo na problem podobnih barv ospredja in ozadja. Problem sčasoma onesposobi originalni sledilnik, medtem ko mu popravek za robustnejšo detekcijo ospredja prepreči, da bi ospredje prehitro ponesreči označil kot še en del ozadja.

Slika 4.1: Primerjava sledenja slik s hitro spremembo osvetlitve. Vsak rdeči okvir predstavlja pravilno lokacijo tarče, zeleni okvir pa poskus sledenja tarči od sledilnika. V zgornjem zaporedju so vidni poskusi izvornega sledilnika DGT_{ORIG}, v spodnjem pa od obeh izpeljank, DGT_{2SVM} in $\text{DGT}_{\text{SESTAVLJEN}}$. Prikazane slike posnetka so zaporedne.



Slika 4.2: Primerjava sledenja slik s podobnimi barvami v ospredju in ozadju. Vsak rdeči okvir predstavlja pravilno lokacijo tarče, zeleni okvir pa poskus sledenja tarči od sledilnika. V zgornjem zaporedju so vidni poskusi sledilnikov DGT_{ORIG} in DGT_{2SVM}, v spodnjem pa od končne izpeljanke $\text{DGT}_{\text{SESTAVLJEN}}$. Prikazana je vsaka 40. slika posnetka.

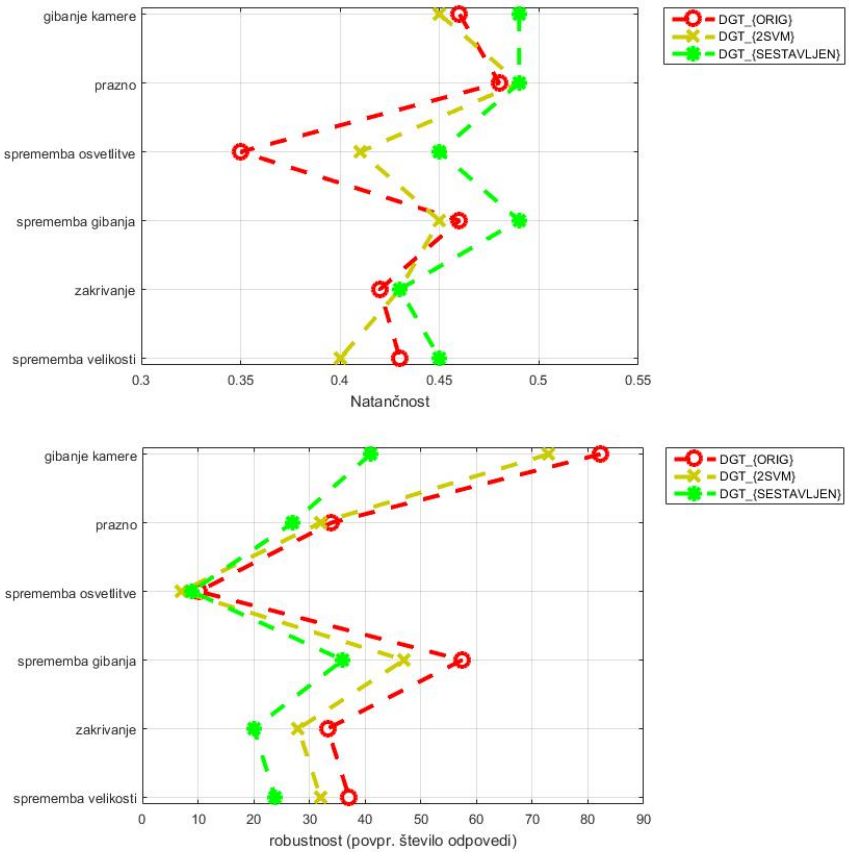
4.3.2 Kvantitativna analiza

Kvantitativna analiza je narejena s pomočjo ocenjevalnih algoritmov iz VOT2015. Medsebojna uspešnost izboljšav algoritma (grupirano po različnih izzivih sledenja) je prikazana na Sliki 4.3. Različica $\mathrm{DGT}_{\mathrm{SESTAVLJEN}}$ je najboljša pri vseh primerih, razen pri robustnosti glede na spremembo osvetlitve (kjer jo prehiti $\mathrm{DGT}_{\mathrm{2SVM}}$). Pri tem velja poudariti, da je sledilnik DGT stohastičen (saj uporablja inačico klasifikatorja SVM z naključnim vzorčenjem učnih primerov), zato je analiza manjših sprememb v ocenah nesmiselna. Iz slike lahko razločimo naslednje opazne spremembe učinkovitosti sledilnika glede na izboljšavi:

- *Izboljšava uporabe klasifikatorja SVM* poviša natančnost pri spremembi osvetlitve in (nepričakovano) poslabša natančnost pri spremembi velikosti - slednje spremembe glede na podrobnosti izboljšave ni moč razložiti. Izboljšava v sklopu robustnosti algoritem naredi zanesljivejši po vseh primerih - povečini zmanjša število odpovedi za ca. 10-20%, razen v primerih brez posebnega izziva, ko robustnost ostaja podobna originalu.

- *Robustna detekcija ospredja* opazno izboljša natančnost pri gibanju kamere, spremembi osvetlitve, spremembi gibanja ter spremembi velikosti. Zmanjša tudi število odpovedi po skoraj vseh primerih za nadaljnjih 20-40%. Opazna izjema pa je poslabšanje robustnosti v primeru spremembe osvetlitve - po pregledu problematičnih sekvenc se je izkazalo, da je sledenje postalo manj zanesljivo v primeri nizkega števila superpikslov v ospredju. Kot se izkaže, je postavljena spodnja meja velikosti klasificiranega ospredja velikokrat povzročila dodajanje kandidatov v ospredje, ki so pravzaprav del ozadja, če so bili kandidati dovolj podobnih barv. Ti nepravilni kandidati so tako v sčasoma "prevzeli" vlogo ospredja sledilnika.

Inačice sledilnika so nadalje primerjane z naslednjimi algoritmi, ki so prisostvovali na tekmovanju VOT2015 [37]:



Slika 4.3: Primerjava uspešnosti treh različic algoritma DGT glede na različne izzive sledenja videoposnetkov. Pri tem oznaka "prazno" označuje primere brez dodatnega izziva.

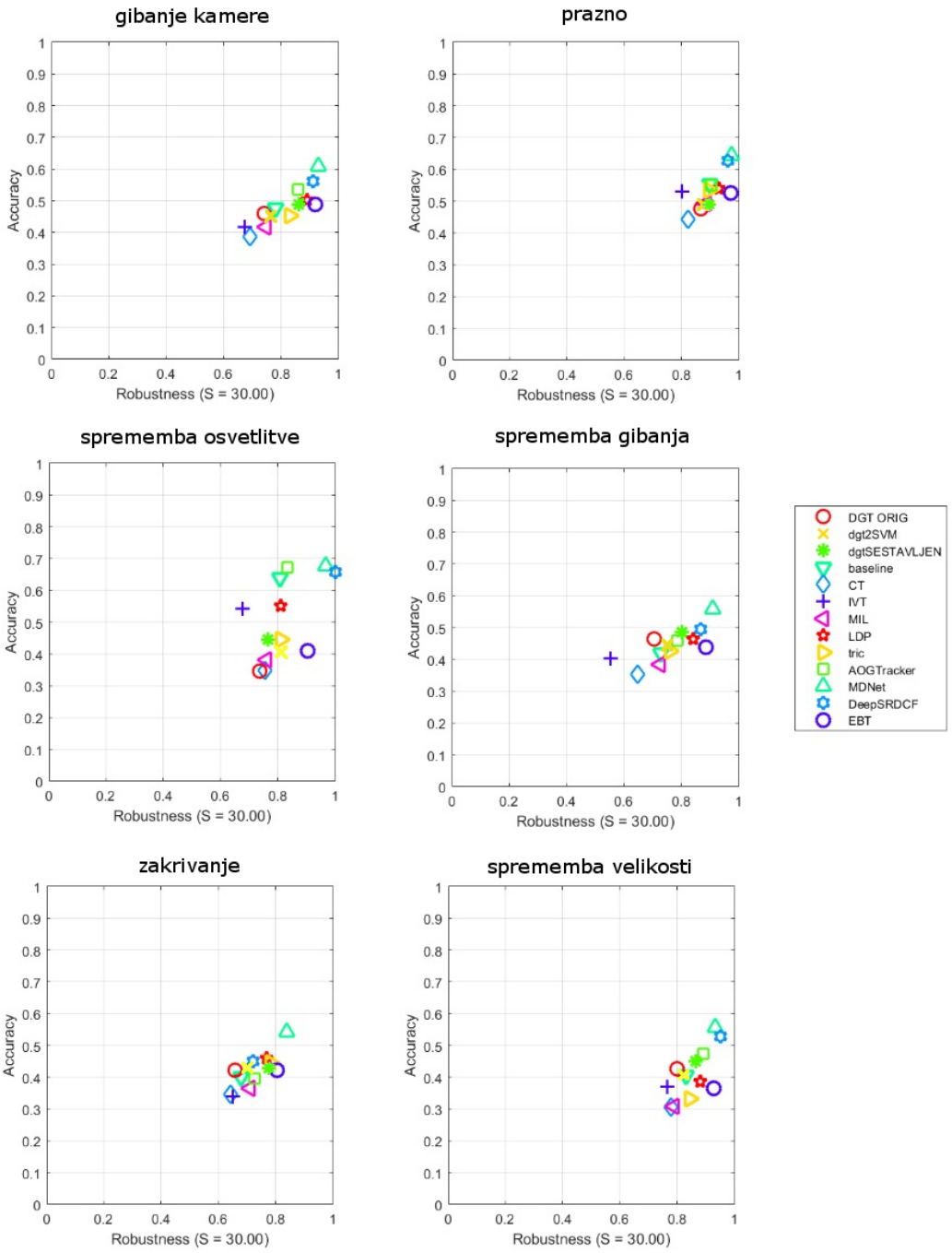
- Standardni osnovni sledilniki: CT [10], IVT [11], MIL [52].
- Najboljši sledilniki z regijami: LDP [53], TRIC-Track [54], AOG [55].
- Najboljši holistični sledilniki: MDNet [56], DeepSRDCF [57], EBT [58].

Obsežna primerjava rezultatov je vidna na Tabeli 4.2, ki vključuje povprečno natančnost, povprečno robustnost, rangiranje natančnosti in robustnosti ter skupno rangiranje. Podrobnejše grafe po različnih kriterijih je moč videti na Sliki 4.4. Ti grafi potrjujejo izsledke iz Slike 4.3 in pokažejo, da sta izboljšavi sledilnik naredila precej kompetentnejši predvsem, kar se tiče robustnosti. Glavna hiba sledilnika ostaja sprememba osvetlitve, kjer v primerjavi z najboljšimi sledilniki iz VOT2015 DGT še vedno močno zaostaja tako po natančnosti kot po robustnosti. Kot najboljša inačica algoritma DGT se je, sodeč po rezultatih, odločno odrezal $\mathrm{DGT}_{\mathrm{SESTAVLJEN}}$. Tudi $\mathrm{DGT}_{\mathrm{2SVM}}$ se je izkazal kot boljši od $\mathrm{DGT}_{\mathrm{ORIG}}$.

Izvedla se je še dodatna primerjava, ki je vključevala prav vse sledilnike iz tekmovanja VOT2015. Sledilnik $\mathrm{DGT}_{\mathrm{SESTAVLJEN}}$ se je v skupnem rangiranju uvrstil na 24. mesto, $\mathrm{DGT}_{\mathrm{2SVM}}$ na 43., $\mathrm{DGT}_{\mathrm{ORIG}}$ pa je zasedel 51. mesto od skupno 65.

Tabela 4.2: Primerjava uspešnejših sledilnikov po surovi natančnosti SN, surovi robustnosti SR, rangu natančnosti RN, rangu robustnosti RR, in skupnem rangu (skupni rang je enak $\frac{SR+RN}{2}$). Najboljši rezultat po kategoriji je označen rdeče, drugi najboljši modro, tretji pa zeleno.

Sledilnik	SN	SR	RN	RR	Skupni rang
MDNet	**2.08**	**1.92**	**1.5**	**2.0**	**0.3783**
DeepSRDCF	**4.19**	**3.17**	**3.5**	**2.0**	**0.3181**
EBT	7.83	**2.13**	7.5	**2.0**	**0.3130**
LDP	6.42	5.52	7.0	**5.0**	0.2785
tric	7.58	6.79	7.5	**7.5**	0.2088
AOGTracker	**5.35**	5.93	**6.0**	**5.0**	0.2080
$DGT_{SESTAVLJEN}$	6.92	6.90	7.0	**5.0**	0.2076
baseline	6.08	8.16	7.0	7.5	0.1935
MIL	9.28	8.92	8.5	10.0	0.1710
DGT_{2SVM}	7.83	9.22	8.0	10.0	0.1610
DGT_{ORIG}	8.21	10.11	8.0	10.0	0.1469
IVT	8.62	11.53	8.5	13.0	0.1220
CT	10.63	11.36	10.6	12.0	0.1135



Slika 4.4: Grafi algoritmov po različnih kriterijih. Vsak graf prikazuje natančnost (angl. "accuracy") in robustnost (angl. "robustness") po algoritmih, kjer višja vrednost pomeni boljši rezultat.

Poglavje 5

Sklep

V okviru diplomske naloge smo analizirali obstoječ algoritem za sledenje vizualnim objektom v videoposnetkih, DGT [35]. Glede na izsledke iz tekmovanja VOT2014 [36] smo ugotovili, da so šibka točka algoritma primeri, ko se osvetlitev sledilnega objekta hitro spreminja. Pregledali smo teoretične osnove ter analizirali osnovno zamisel in podrobno sestavo algoritma.

V algoritmu smo odkrili pomanjkljivo implementacijo uporabe klasifikatorja SVM, za katero smo izpeljali izboljšavo, ki je algoritem pohitrila in mu povišala natančnost, predvsem v primerih hitrih sprememb osvetlitve. Poleg tega je algoritem občasno trpel za izgubo sledi nad tarčo, vzrok katere je bilo preveliko povečanje števila značilk - superpikslov, zaradi česar je algoritem sčasoma ozadje scene privzel kot tarčo. Za bolj robustno detekcijo ospredja smo vpeljali izboljšavo, ki algoritem omeji s spodnjo in zgornjo mejo števila značilk v odvisnosti od začetne velikosti sledilnega objekta. Posledično je sledilnik postal stabilnejši, kar se tiče zanesljivega sledenja tarči.

Iz algoritma smo (glede na predloga) izgotovili dve novi izpeljanki, ki smo ju s pomočjo sistema za analizo sledilnikov iz VOT2015 analizirali ter primerjali s konkurenco. Oba predloga sta se izkazala kot uspešna, saj se je izboljšala natančnost sledilnika, povprečno število odpovedi pa se je zmanjšalo za okoli 30%. Popravek pri uporabi klasifikatorja SVM je močno izboljšal predvsem primere videoposnetkov s hitrimi spremembami osvetlitve. Po

drugi strani je robustnejša detekcija ospredja izboljšala algoritem v veliki večini primerov, vendar se je število odpovedi v primerih hitrih sprememb osvetlitve povišalo. Kljub temu se je sledilnik DGT, opremljen z obema izboljšavama, izkazal podobno ali bolje od originala v vseh preverjenih ocenah. S predlaganimi izboljšavami smo sledilnik približali najuspešnejšim iz tekmovanja VOT2015, saj se je (glede na vse sledilnike v tekmovanju VOT2015) povzpel iz 49. na 24. mesto od skupaj 63. Kljub poudarku na izboljšavah pri spremembah osvetlitve objekta pa je ta problem še vedno najšibkejša točka izboljšanega sledilnika (glede na konkurenco).

5.1 Možne nadgradnje

Implementacija algoritma DGT je ponekod še vedno preveč toga. Čeprav je algoritem že učinkovit pri občutnih spremembah velikosti objekta, se podre, kadar očrtani pravokotnik okoli tarče preseže dimenzije na začetku izbranega okvirja okoli objekta. Nezanesljiv postane tudi, ko objekt postane tako manj, da ga pokriva le manj kot 5 superpikslov. S pazljivo izbrano dinamično velikostjo pravokotnika in superpikslov bi izboljšali robustnost in tudi hitrost algoritma.

Za izboljšano delovanje v primeru močnih sprememb osvetlitve objekta bi bilo potrebno prevetriti korak začetne razdelitve superpikslov na ospredje in ozadje. Dodali bi lahko detekcijo sprememb svetlosti scene v zadnjih nekaj slikah. V primerih večjih sprememb osvetlitve zadnjih nekaj slik bi lahko vpeljali modeliranje osvetlitve in na novi sliki predvideli na novo osvetljene barve ospredja ter ozadja scene.

Literatura

[1] P. F. Gabriel, J. G. Verly, J. H. Piater, A. Genon. "The State of the Art in Multiple Object Tracking Under Occlusion in Video Sequences".

[2] "Multiple Object Tracking Benchmark". [Online]. Dosegljivo: https://motchallenge.net/ [Dostopano 24. 8. 2016].

[3] D. P. Young, J. M. Ferryman. "PETS Metrics: On-line performance evaluation service", v zborniku: ICCCN '05 Proceedings of the 14th International Conference on Computer Communications and Networks, 2005, str. 317-324

[4] "CAVIAR: Context Aware Vision using Image-based Active Recognition". [Online]. Dosegljivo: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/ [Dostopano 31. 7. 2016].

[5] "Video Understanding Evaluation" [Online]. Dosegljivo: http://www-sop.inria.fr/orion/ETISEO/ [Dostopano 31. 7. 2016].

[6] "CVBASE '06 - Workshop on Computer Vision Based" [Online]. Dosegljivo: http://vision.fe.uni-lj.si/cvbase06/ [Dostopano 31. 7. 2016].

[7] P. J. Phillips, H. Moon, S. A. Rizvi, P. J. Rauss. "The feret evaluation methodology for face-recognition algorithms", v zborniku: IEEE Trans. Pattern Anal. Mach. Intell. 22(10), 2000, str. 1090-1104

[8] R. Kasturi, D. B. Goldgof, P. Soundararajan, V. Manohar, J. S. Garofolo, R. Bowers, M. Boonstra, V. N. Korzhova, J. Zhang. "Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol", v zborniku: . IEEE Trans. Pattern Anal. Mach. Intell. 31(2), 2009, str. 319-336

[9] H. Grabner, H. Bischof. "On-Line Selection of Discriminative Tracking Features", Proc. IEEE CS Conf. Computer Vision and Pattern Recognition

[10] K. Zhang, L. Zhang, M.-H. Yang. "Real-Time Compressive Tracking", v zborniku: Proc. Eur. Conf. Comput. Vis., 2012, str. 864-877

[11] D. Ross, J. Lim, R. Lin, M. Yang. "Incremental Learning For Robust Visual Tracking", v zborniku: Int'l J. Computer Vision, vol. 77, no. 1, 2007, str. 125-141

[12] Z. Hong, X. Mei, D. Prokhorov, D. Tao. "Tracking via robust multi-task multi-view joint sparse representation", v zborniku: Proc. IEEE Int. Conf. Comput. Vis., str. 649-656

[13] S. Avidan. "Support Vector Tracking", v zborniku: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, Vol. 1, 2001, str. 184-191

[14] H. Grabner, M. Grabner, H. Bischof. "Real-time tracking via online boosting", v zborniku: BMVC, 2006, str. 47-56

[15] M. Zhang, J. Xing, J. Gao, X. Shi, Q. Wang, W. Hu. "Joint Scale-Spatial Correlation Tracking with Adaptive Rotation Estimation", v zborniku: 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), 2015, str. 595-603

[16] Y. Li, J. Zhu. "A scale adaptive kernel correlation filter tracker with feature integration", v zborniku: Proceedings of the ECCV Workshop, 2014, str. 254-265

[17] S. Hare, A. Saffari, P. Torr. "Struck: Structured output tracking with kernels", ICCV (2011).

[18] A. Adam, E. Rivlin, I. Shimshoni. "Robust Fragments-Based Tracking Using the Integral Histogram", v zborniku: Proc. IEEE CS Conf. Computer Vision and Pattern Recognition, str. 798-805

[19] L. Cehovin, M. Kristan, A. Leonardis. "An adaptive coupled-layer visual model for robust visual tracking", v zborniku: Proc. IEEE ICCV, str. 1363-1370

[20] W. Wang, R. Nevatia. "Robust object tracking using constellation model with superpixel", v zborniku: Proc. 11th ACCV, vol. 3, str. 191-204

[21] A. B. V. Graciano, R. M. Cesar, I. Bloch. "Graph-based object tracking using structural pattern recognition", v zborniku: Proc. SIBGRAPI, str. 179-186

[22] X. Ren, J. Malik. "Tracking as repeated figure/ground segmentation", v zborniku: Proc. IEEE CVPR, str. 1-8

[23] S. Wang, H. Lu, F. Yang, M.-H. Yang. "Superpixel tracking", v zborniku: Proc. IEEE ICCV, str. 1323-1330

[24] M. Yang, J. Yuan, Y. Wu. "Spatial selection for attentional visual tracking", v zborniku: Proc. IEEE CVPR, str. 1-8

[25] D. B. Judd. "Hue saturation and lightness of surface colors with chromatic illumination", v zborniku: JOSA 30(1), str. 2–32

[26] "Visual Object Tracking Challenge" [Online]. Dosegljivo: http://www.votchallenge.net/ [Dostopano 31.7.2016]

[27] "SLIC Superpixels" [Online]. Dosegljivo: http://ivrl.epfl.ch/research/superpixels [Dostopano 31.7.2016]

[28] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk. "SLIC Superpixels*", v poročilu: EPFL Technical Report no. 149300, 2010.

[29] "SLIC Superpixels - SLICO" [Online]. Dosegljivo: http://ivrl.epfl.ch/research/superpixels#SLICO [Dostopano 31.7.2016]

[30] P. Felzenszwalb, D. Huttenlocher. "Efficient graph-based image segmentation", v zborniku: International Journal of Computer Vision, 59(2), 2004, str. 167-181

[31] G. Mori. "Guiding Model Search Using Segmentation", v zborniku: IEEE international conference on computer vision (ICCV), 2007, str. 1-8

[32] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, K. Siddiqi. "Turbopixels: Fast superpixels using geometric flows.", v zborniku: IEEE Transactions on Pattern Analysis and Machine Intelligence 31(12), 2009, str. 2290-2297

[33] A. Vedaldi, S. Soatto. "Quick shift and kernel methods for mode seeking", v zborniku: European Conference on Computer Vision, 2008, str. 705-718

[34] M. Leordeanu, M. Hebert. "A Spectral Technique for Correspondence Problems", v zborniku: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 (Volume: 2), 2005, str. 1482-1489 (Vol. 2)

[35] Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, S. Z. Li. "Robust Deformable and Occluded Object Tracking With Dynamic Graph", v zborniku: IEEE Transactions on Image Processing. 23(12), 2014, str. 5497-5509.

[36] "The Visual Object Tracking VOT2014 challenge results" [Online]. Dosegljivo: http://www.votchallenge.net/vot2014/download/vot_2014_paper.pdf [Dostopano 31.7.2016]

[37] "The Visual Object Tracking VOT2015 challenge results" [Online]. Dosegljivo: http://www.votchallenge.net/vot2015/download/vot_2015_paper.pdf [Dostopano 25.8.2016]

[38] Y. Boykov, V. Kolmogorov. "An experimental comparison of mincut/max-flow algorithms for energy minimization in vision", v zborniku: IEEE Trans. Pattern Anal. Mach. Intell., Volume 26, št. 9, 2004, str. 1124-1137

[39] Y. Boykov, O. Veksler, R. Zabih. "Fast Approximate Energy Minimization via Graph Cuts", v zborniku: Proc. Medical Image Computing and Computer-Assisted Intervention, 2000, str. 276-286.

[40] A. Bordes, S. Ertekin, J. Weston, L. Bottou. "Fast kernel classifiers with online and active learning", v zborniku: J. Mach. Learn. Res., vol. 6, 2005, str. 1579-1619

[41] "Robust Deformable and Occluded Object Tracking with Dynamic Graph" [Online]. Dosegljivo: https://sites.google.com/site/zhaoweicai1989/dgt/ [Dostopano 14.8.2016]

[42] "VOT2013 Challenge" [Online]. Dosegljivo: http://www.votchallenge.net/vot2013/ [Dostopano 14.8.2016]

[43] "VOT2014 Challenge" [Online]. Dosegljivo: http://www.votchallenge.net/vot2014/ [Dostopano 14.8.2016]

[44] "VOT2015 Challenge" [Online]. Dosegljivo: http://www.votchallenge.net/vot2015/ [Dostopano 14.8.2016]

[45] A. R. Smith. "Color Gamut Transform Pairs", v zborniku: SIGGRAPH 78 Conference Proceedings, 1978, str. 12-19

[46] X. Ren, J. Malik. "Learning a Classification Model for Segmentation", v zborniku: Computer Vision, 2003. Proceedings. Ninth IEEE Internation conference on, 2003, str. 10-17

[47] Y. Wu, J. Lim, M. H. Yang. "Online Object Tracking: A benchmark", Computer Vision and Pattern Recognition, 2013

[48] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah. "Visual Tracking: an Experimental Survey", TPAMI, 2013

[49] F. Fleuret, J. Berclaz, R. Lengagne, P. Fua. "Multicamera people tracking with a probabilistic occupancy map", v zborniku: IEEE Trans Pattern Anal Mach Intell 30(2), 2008, str. 267-282

[50] "Change Detection Workshop". [Online]. Dosegljivo: "http://www.changedetection.net/" [Dostopano 24.8.2016]

[51] T. Vojir, J. Noskova, J. Matas. "Robust scale-adaptive mean-shift for tracking", v zborniku: Image Analysis, 2013, str. 652-663

[52] B. Babenko, M.-H. Yang, S. Belongie. "Robust object tracking with online multiple instance learning", v zborniku: IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011), str. 1619-1632

[53] A. Lukežič. "Improved robust part-based model for visual object tracking". Ljubljana : [A. Lukežič], 2015

[54] X. Wang, M.F. Valstar, B. Martinez, M.H. Khan, T.P. Pridmore. "TRIC-track: Tracking by Regression with Incrementally Learned Cascades", IEEE Int'l Conf. on Computer Vision (ICCV), 2015

[55] T. Wu, Y. Lu, S.-C. Zhu. "Online Object Tracking, Learning and Parsing with And-Or Graphs", v zborniku: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2014, str. 3462-3469

[56] H. Nam, B. Han. "Learning Multi-Domain Convolutional Neural Networks for Visual Tracking", CoRR, 2015

[57] M. Danelljan, G. Hager, F. S. Khan, M. Felsberg. "Learning spatially regularized correlation filters for visual tracking", International Conference on Computer Vision, 2015

[58] G. Zhu, F. Porikli, H. Li. "Tracking randomly moving objects on edge box proposals", CoRR, 2015

[59] M. D. Fairchild. "Color Appearance Models". Wiley, 3rd edition, 2013

[60] R. Kindermann, J. L. Snell. "Markov Random Fields and Their Applications". American Mathematical Society, 1980