Algoritmi za prepoznavanje pokreta i primjena na automatsko praćenje objekata

Kunštek, Alan

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Alan Kunštek

ALGORITMI ZA PREPOZNAVANJE POKRETA I PRIMJENA NA AUTOMATSKO PRAĆENJE OBJEKATA

Diplomski rad

Voditelj rada:
Dr. sc. Goran Igaly

Zagreb, veljača, 2016.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

Proučavanje algoritama za detekciju pokreta je tema koja u zadnje vrijeme dobiva sve više pozornosti u znanstvenim krugovima. Spada pod granu računarstva koja se naziva računalni vid. Rana istraživanja sežu čak do sedamdesetih godina prošlog stoljeća. Računalni vid ima mnoge primjene kao što su prepoznavanje (pokreta, lica, znakovlja), analiza pokreta, rekonstrukcija (stvaranje 3D modela na temelju snimaka iz više kutova), interakcija između čovjeka i stroja, te robotička navigacija. U ovom diplomskom radu proučit ćemo kako možemo na video snimkama ili živim video podacima prepoznati zanimljiva područja, odnosno područja na kojima se nešto kreće.

Većina algoritama za prepoznavanje pokreta temelji se na principu oduzimanja pozadine (eng. background subtraction). Algoritam stvara model pozadine (eng. background) u toku svog rada, te usporedbom s trenutnom sličicom (eng. frame) zaključuje što je u prvom planu (eng. foreground). Postoje mnoge prepreke i okolnosti koje otežavaju rad takvih algoritama. Prva i osnovna među njima je kretanje kamere koja snima. Naime, ako pretpostavimo da je kamera koja snima fiksna, prilično je očito koliko je jednostavnije implementirati prepoznavanje pokreta, jer će stvari koje su stacionarne u stvarnosti biti stacionarne i na videu od sličice do sličice. Međutim, ako dozvolimo da se kamera kreće tada moramo pribjeći raznim kompenzacijskim tehnikama da bismo uspjeli razlikovati pozadinu od prvog plana. Iduća potencijalna okolnost koja može otežati rad algoritma su promjene u iluminaciji scene. Ako trebamo pratiti pokret u zatvorenom i u relativno kratkom vremenskom razdoblju navedena situacija nam ne smeta, no što ako se radi o praćenju pokreta na otvorenom na neodređeno vrijeme? Očito algoritam treba kompenzirati za promjene radi ciklusa dana i noći, te promjene osvjetljenja uslijed na primjer iznenadnog naoblačenja. Još jedna okolnost koja može otežati rad algoritma je pozadina sa ”šumovima”. Na primjer, ako se snima scena s vodenim površinama očito je da valovi uzrokuju razliku u pozadini od sličice do sličice. Slična situacija je i sa nebom te oblacima koji prolaze. Još jedan potencijalni ”šum” u pozadini može biti vegetacija. Za razliku od vode uglavnom je stacionarna, no što kada krene puhati vjetar? Algoritam koji se izabire za implementaciju praćenja pokreta mora uzeti u obzir sve moguće okolnosti te biti sposoban za razlikovanje prvog plana i pozadine unatoč svim smetnjama.

U prvom poglavlju ovog diplomskog rada navodim nekoliko tehnika za praćenje pokreta uz pretpostavku stacionarne kamere, dok se tehnike za pokretnu kameru objašnjavaju u drugom poglavlju. U trećem poglavlju daje se prijedlog za implementaciju automatskog praćenja objekata kamerom.

Poglavlje 1

Oduzimanje pozadine na snimkama sa statičnom kamerom

1.1 Osnovne tehnike

Metoda razlike sličica

Ova metoda je jedna od osnovnih i najrasprostranjenijih metoda kada je u pitanju detekcija pokreta na snimci statičnom kamerom. Metoda se temelji na pronalaženju razlike među pikselima u uzastopnim sličicama, te se provodi u nekoliko koraka (vidi [8]). Prvi od tih koraka je definiranje slike apsolutne razlike:

Definicija 1.1.1. *Neka je I_k vrijednost k-te sličice a I_{k+1} vrijednost k+1-ve sličice u slijedu sličica. Tada se slika apsolutne razlike definira kao:*

$$I_{d(k,k+1)} = |I_{k+1} - I_k|$$

Istaknimo samo da se razlika provodi na svakom pikselu u sličicama. Idući korak koji radimo da bi se olakšala daljnja obrada je pretvaranje dobivene sličice u sličicu u nijansama sive (eng. grayscale). Pretvorba se obavlja po formuli

$$0.299 * R + 0.587 * G + 0.114 * B \rightarrow y \tag{1.1}$$

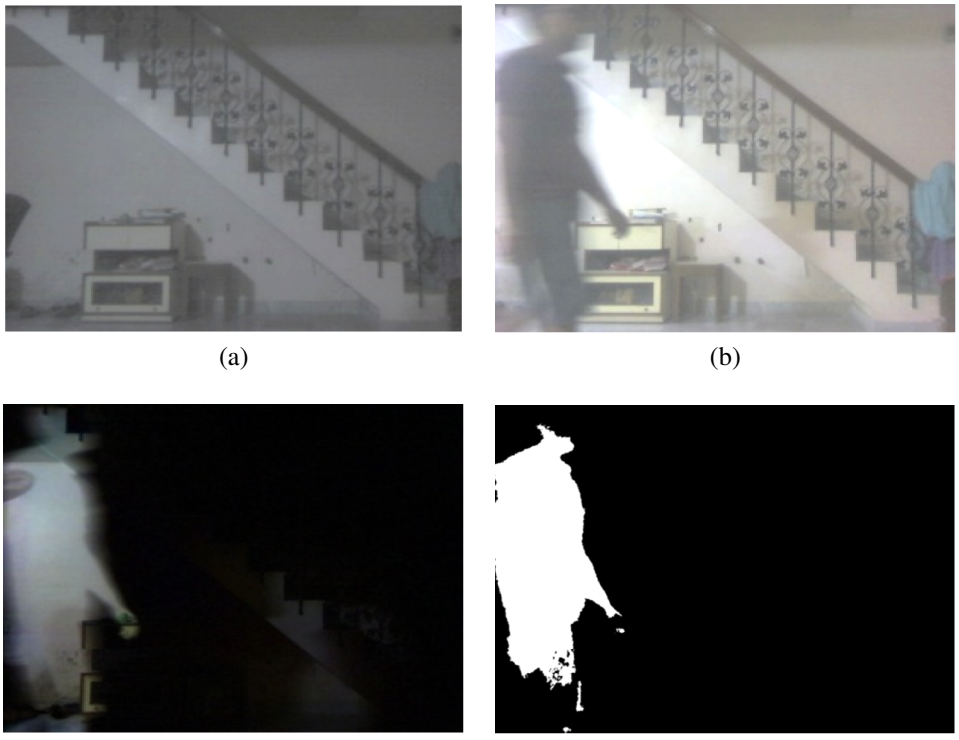
gdje su R, G i B odgovarajuće vrijednosti RGB spektra.

Opcionalni predzadnji korak je da se dobivena sličica provlači kroz neki filtar koji bi otklonio rupe u objektu koji se kreće. Spomenute rupe mogu nastati ako je objekt koji se kreće uniformne boje a između dvije sličice koje se uspoređuju postoji presjek pozicije objekta. Tada će algoritam taj dio objekta koji je u presjeku shvatiti kao pozadinu u sličici.

U posljednjem koraku za svaki piksel utvrđujemo pripada li pozadini ili prvom planu po principu:

$$I_{df(k,k+1)}(x,y) = \begin{cases} 1\ , I_{d(k,k+1)}(x,y) > Th \\ 0\ , \text{inače} \end{cases}$$

gdje je Th proizvoljna vrijednost praga (eng. threshold), te $I_{df(k,k+1)}(x,y)$ poprima vrijednost 1 ako odgovarajući piksel pripada prvom planu, a 0 ako pripada pozadini. Prag određujemo prije početka rada. Ako odaberemo premalu vrijednost za prag može se desiti da algoritam uhvati i dijelove pozadine te ih prepozna pogrešno kao dio prvog plana. Ako pak odaberemo preveliku vrijednost desit će se da dijelovi prvog plana ne budu prepoznati.



(a) (b)

(c) (d)

Slika 1.1: Koraci u metodi razlike sličica

Na slici 1.1 pod (a) i (b) vidimo 2 sličice koje se uspoređuju. Pod (c) vidimo rezultat nakon apsolutne razlike piksela a pod (d) rezultat nakon zadnjeg koraka algoritma kada je sličica binarizirana. Preuzeto iz [8]

Pozadina kao prosjek n prethodnih sličica

Prethodna metoda je prilično jednostavna, no puno bolji rezultati se postižu metodama koje koriste statistiku za procjenu pozadine. Najjednostavnija među njima računa pozadinu kao aritmetičku sredinu n prethodnih sličica [5], gdje je n proizvoljan prirodan broj. Za identifikaciju prvog plana u ovoj metodi koristimo standardnu tehniku razlike piksela u trenutnoj sličici i piksela u procijenjenoj pozadini te usporedbu dobivene razlike s nekom vrijednosti praga, slično kao u prethodnoj metodi.

Ovaj pristup je procesorski jednostavan, te se računanja obavljaju prilično brzo, no problem predstavljaju memorijski zahtjevi. Naime, s obzirom da za svaku sličicu pozadinu procjenjujemo na temelju prethodnih n sličica, potrebno je u svakom trenutku u memoriji držati $n * size(frame)$ byte-ova.

Varijacija ove metode je metoda koja koristi medijan umjesto aritmetičke sredine. Ova varijacija rezultira sličnim, iako malo preciznijim rezultatima.

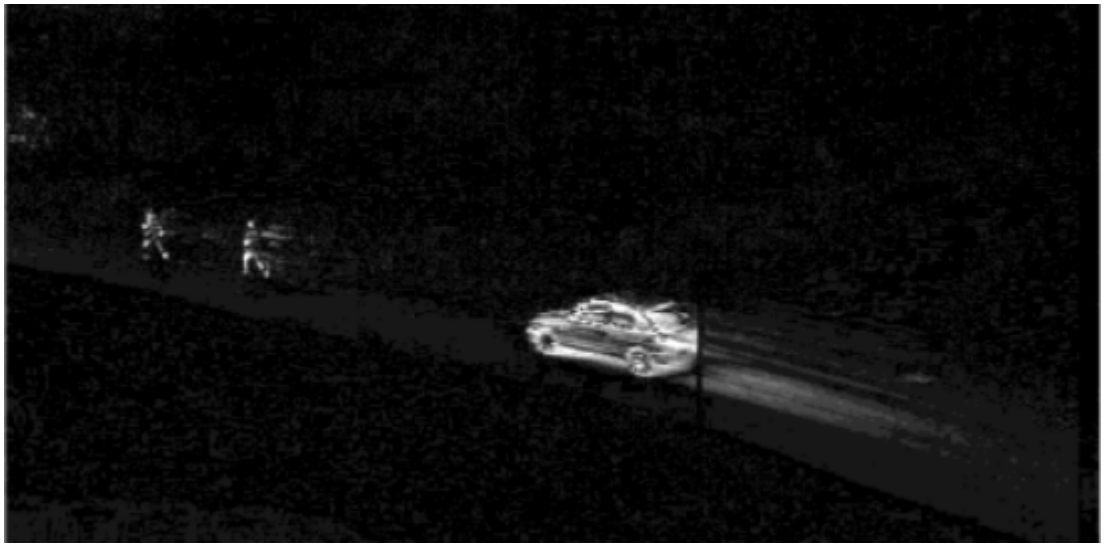
Pozadina kao tekući prosjek

Unapređenje prethodne metode možemo postići uvođenjem formule koja će dobro aproksimirati prosjek iz prethodne metode. Predložena formula [5] je dana sa:

$$B_n(x, y) = \alpha \cdot F_n(x, y) + (1 - \alpha) \cdot B_{n-1}(x, y) \tag{1.2}$$

gdje su:
$B_n(x, y)$ - nova procjena prosjeka
$B_{n-1}(x, y)$ - dosadašnja procjena prosjeka
$F_n(x, y)$ - piksel iz n-te odnosno trenutne sličice
α - faktor procjene, obično neki broj iz intervala $[0.01, 0.1]$

Ovom metodom postižemo minimalna poboljšanja u procesorskim zahtjevima, no memorijski zahtjevi postaju zanemarivi u odnosu na prosjek n prethodnih sličica. Prosjek sličica kroz vrijeme rezultira time da se smanjuju šumovi, te dobivamo pouzdanu metodu u kojoj se pozadina postepeno mijenja. Neki od nedostataka ove metode su što ima problema s pozadinama koje se brzo mijenjaju, pogotovo ako te promjene imaju veliku varijancu, te što pogrešan odabir faktora procjene α može uzrokovati pojavljivanje tragova iza objekata u pokretu (vidi sliku 1.2).



Slika 1.2: Pojava tragova iza objekta u pokretu

Tekući prosjek sa selektivnim ažuriranjem

Uobičajeni način da se otklone neki od nedostataka prethodne metode je da se uvede postupak u kojem selektivno ažuriramo pozadinu, pa tada pikseli koji su pripadali prvom planu ne ”zagađuju” pozadinu. To možemo postići sljedećom formulom[5]:

$$B_n(x,y) = \begin{cases} B_{n-1}(x,y) & , \text{ ako } |F_n(x,y) - B_{n-1}(x,y)| > T \\ \alpha \cdot F_n(x,y) + (1-\alpha) \cdot B_{n-1}(x,y) & , \text{ inače} \end{cases}$$

gdje su oznake iste kao u jednadžbi (1.2), a T je proizvoljni prag za razlikovanje pozadine od prvog plana.

Ovaj pristup omogućuje otkrivanje sporo krećućih ili čak nepomičnih objekata koji pripadaju prvom planu, no nažalost može također uzrokovati dodatne šumove ili čak fantomske objekte.

Metoda aproksimacije medijana

Sada kada smo pokazali kako se mogu otkloniti memorijski zahtjevi iz tehnike s aritmetičkom sredinom, pokazati ćemo još i način za aproksimaciju medijana, koji se onda može koristiti umjesto metode prosjeka n prethodnih sličica s medijanom. Formula [5] je

rekurzivna, te je dana s:

$$B_{n+1}(x,y) = \begin{cases} B_n(x,y) + 1 & , \text{ za } F_n(x,y) > B_n(x,y) \\ B_n(x,y) - 1 & , \text{ za } F_n(x,y) < B_n(x,y) \\ B_n(x,y) & , \text{ za } F_n(x,y) = B_n(x,y) \end{cases}$$

gdje su:
$B_{n+1}(x,y)$ - nova procjena prosjeka
$B_n(x,y)$ - dosadašnja procjena prosjeka
$F_n(x,y)$ - piksel iz n-te odnosno trenutne sličice

Isto kao kod aproksimacije aritmetičke sredine ovaj način omogućuje otklanjanje većine memorijskih zahtjeva, te praktičnu primjenu uslijed niskih zahtjeva na računalne resurse.

1.2 Gaussova krivulja tekućeg prosjeka

Temelj metode

Ova tehnika se zasniva na modeliranju pozadine pomoću Gaussove krivulje. Osnovna ideja je da se vrijednosti piksela u posljednjih n sličica opišu pomoću Gaussove krivulje, no vidjeli smo u nekim od prethodnih metoda da pristupi koji se temelje na n prethodnih sličica nisu baš iskoristivi u praktične svrhe radi ogromnih memorijskih zahtjeva.

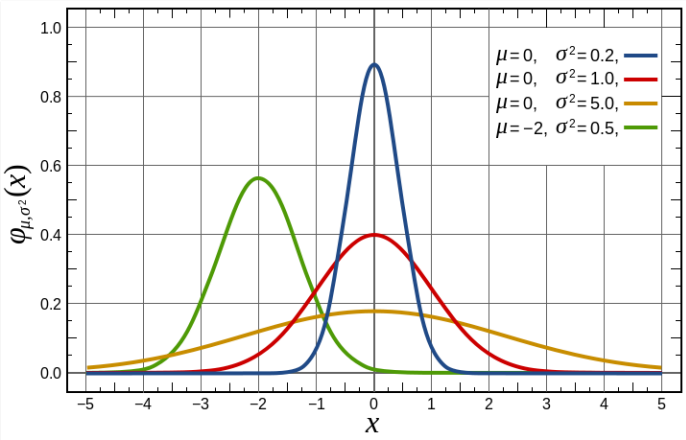
Gaussova krivulja je definirana s:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

gdje je μ očekivanje a σ^2 varijanca.

Konstrukcija pozadinskog modela

Kako bismo omogućili korištenje u praktične svrhe, konstrukcija pozadinskog modela na temelju n prethodnih sličica se obavlja samo u fazi inicijalizacije, a kasnije u toku rada algoritma model se ažurira samo na temelju trenutne sličice.



Slika 1.3: Primjeri Gaussovih krivulja

U inicijalizaciji [6] uzimamo n sličica te na temelju njih računamo srednju vrijednost za svaki piksel posebno po formuli:

$$\mu = \frac{1}{n}\sum_{t=1}^{n} x_t$$

gdje je n broj sličica za inicijalizaciju a x_t vrijednost piksela za sličicu t.

Nakon što smo izračunali srednju vrijednost za svaki piksel, standardnu devijaciju možemo procijeniti na sljedeći način:

$$\sigma^2 = \frac{1}{n-1}\sum_{t=1}^{n}(x_t - \mu)^2$$

Oduzimanje pozadine

Trenutnu sličicu oduzimamo od prethodno procijenjene slike srednjih vrijednosti, te dobivamo sliku razlike. Možemo pretpostaviti da kamera ne daje savršenu sliku, te da se šumovi na slici mogu uhvatiti Gaussovom krivuljom. Stoga obavljamo korak normalizacije pouzdanosti, koristeći dva praga. Odabrani pragovi su zapravo standardna devijacija pozadinskog modela pomnožena nekim skalarom, te dijele prostor mogućih vrijednosti piksela na dva intervala gdje je moguća jasna klasifikacija, te treći interval gdje je potrebna dodatna analiza.

Rezultat ove normalizacije predstavlja pouzdanost svakog piksela da bude klasificiran kao pripadnik prvog plana. Ako je vrijednost razlike ispod praga $m\sigma$, možemo pretpostaviti da se pripadnom pikselu vrijednost promijenila zbog šumova u slici te da on ipak pripada

pozadini. U ovom slučaju pouzdanost se postavlja na 0%. S druge strane ako je vrijednost razlike iznad praga $M\sigma$, to znači da je vrijednost jako različita od srednje vrijednosti te može biti uzrokovana jedino objektom u pokretu; pouzdanost se postavlja na 100%.

Za vrijednosti razlike koje se nalaze između dva praga, pouzdanost skaliramo linearno po formuli:

$$C = \frac{D - m\sigma}{M\sigma - m\sigma}$$

gdje je D vrijednost razlike. Naposljetku, segmentaciju slike dobivamo primjenom neke funkcije praga na sliku pouzdanosti.

Ažuriranje pozadinskog modela

Nagle promjene u sceni, poput početka ili zaustavljanja kretanja objekta, trebale bi se trenutačno odražavati na pozadinski model, kako bi se omogućilo prepoznavanje samo objekata koji se uistinu kreću. Iz tog razloga uvodimo novu varijablu α, koja će nam predstavljati stopu učenja modela. Za piksele koje smo prepoznali kao pripadnike prvog plana, trebamo uzeti nisku vrijednost α kako bi se spriječila integracija pokretnih objekata u pozadinski model. Za pozadinske piksele vrijednost α uzima se po potrebi. Ako želimo veću stabilnost modela uzimamo nešto nižu vrijednost, a ako želimo brzo ažuriranje promjena uzimamo nešto višu vrijednost.

Sam pozadinski model ažuriramo za svaku novu sličicu u svakom pikselu. Srednju vrijednost ažuriramo po formuli:

$$\mu_t = \alpha x_t + (1 - \alpha)\mu_{t-1}$$

gdje su:
μ_t - srednja vrijednost procijenjena do sličice t
α - stopa učenja modela
x_t - vrijednost piksela u sličici t
Kao što vidimo srednja vrijednost se računa težinskom funkcijom koja uzima u obzir sve prijašnje vrijednosti, te trenutnu vrijednost piksela.

Kada smo ažurirali srednju vrijednost, možemo ažurirati i standardnu devijaciju, i to na idući način:

$$\sigma_t^2 = \alpha(x_t - \mu_t)^2 + (1 - \alpha)\sigma_{t-1}^2$$

Kao što vidimo, korištenje ove metode u svakom trenutku zahtjeva spremanje uređenog para (μ_t, σ_t) za svaki piksel, što je puno bolje od toga da spremamo vrijednosti n prethodnih piksela, jer memorijski zahtjevi opadaju drastično a ni pad procesorskih zahtjeva nije zanemariv. Ovdje je navedena generalizacija metode za jedan kanal, no u [6] autor navodi kako se metoda može primjenjivati i direktno na videu u više kanala boje, poput RGB ili YIQ. Ako koristimo više kanala metoda se provodi na svaki kanal zasebno te se navode

tehnike za spajanje zasebnih rezultata u konačan rezultat. Također se detaljno objašnjava konverzija RGB slike u nijanse sive (eng. grayscale), koju smo koristili u formuli (1.1).

Varijacija kombiniranjem s metodom razlike sličica

U [9] autor predlaže kombiniranje dviju metoda, kako bi se dobio što točniji rezultat. Rezultati dviju metoda se spajaju te se dobiva nova točnija binarizirana slika. Nakon toga se provode koraci kako bi se otklonile "rupe" u objektima koji se kreću. Koristeći 3×3 ili 5×5 prozore testiramo svaki piksel iz prvog plana. Postavljamo ga u sredinu prozora, te provjeravamo koliko okolnih piksela također pripada prvom planu. Ako je odgovor više od pola tada svi pikseli u prozoru postaju prvi plan. Ovaj korak je učinkovit za male rupe ali beskoristan za velike. Zato nam je potreban idući korak u kojem koristimo prozore veličine 25×25 da bismo testirali piksele koji pripadaju pozadini. Ako ima "dovoljno" piksela prvog plana u svakom od osam smjerova (vertikalno, horizontalno te dijagonalno) tada zaključujemo da bi piksel trebao pripadati prvom planu, a ne pozadini.

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

Slika 1.4: Kombinirana metoda.

Na slici 1.4 možemo vidjeti metodu u akciji. (a) prikazuje rezultat metode Gaussove krivulje tekućeg prosjeka. (b) prikazuje rezultat metode razlike sličica. (c) prikazuje spoj dviju metoda. (d) prikazuje konačan rezultat nakon popunjavanja rupa.

1.3 Metoda mješavine Gaussovih krivulja

Ideja

Sljedeća metoda koju pokazujemo prvi put je predložena 1999.j a njeni su autori Stauffer i Grimson. Njihova ideja je generalizacija osnovne ideje iz konteksta nadzora prometa, koja predlaže da se svaki pozadinski piksel modelira koristeći mješavinu triju Gaussovih krivulja. Jedna bi odgovarala cesti, druga vozilu, a treća sjenama. Metoda se inicijalizira pomoću EM (expectation-maximization) algoritma, koji je statistička iterativna metoda za procjenu najvjerojatnijih vrijednosti parametara u statističkim modelima, gdje model ovisi

o neopaženim latentnim varijablama. Nakon inicijalizacije Gaussove krivulje se ručno označuju: najtamnija komponenta kao sjena, od preostale dvije ona s najvećom varijancom kao vozilo, te zadnja kao cesta. Ovi parametri ostaju fiksirani u toku čitavog rada algoritma što uzrokuje manjak sposobnosti prilagodbe kroz vrijeme. Za detekciju prvog plana svaki piksel se uspoređuje sa svim krivuljama redom, te se na temelju toga donosi zaključak.

Inicijalizacija

Metoda se zasniva na ideji da se računa vjerojatnost pojavljivanja trenutne vrijednosti piksela u pozadini, na temelju svih distribucija kojima smo okarakterizirali pozadinu. To radimo na idući način:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(X_t, \mu_{i,t}, \Sigma_{i,t})$$

gdje su:
K - broj distribucija
$\omega_{i,t}$ - težina pripisana i-toj Gaussovoj krivulji u vremenu t
$\mu_{i,t}$ - srednja vrijednost
$\Sigma_{i,t}$ - standardna devijacija
η - funkcija Gaussove krivulje

K, odnosno broj Gaussovih krivulja kojima modeliramo pozadinu ovisi o raznolikosti pozadine, te o dostupnosti memorije i procesorske snage. Autori izvorno predlažu da K bude neki broj od 3 do 5.
Inicijalizacija ostalih parametara se obavlja ranije spomenutim EM algoritmom.

Detekcija prvog plana

Nakon inicijalizacije parametara, spremni smo za prvu detekciju prvog plana. Gaussove krivulje se poredaju po omjeru težina i standardnih devijacija. Taj poredak se nameće iz pretpostavke da pozadinski piksel odgovara visokoj težini te niskoj varijanci zbog činjenice da je pozadina prisutnija od prvog plana te da joj je vrijednost više manje konstantna. Sada prvih nekoliko Gaussovih krivulja odgovara pozadini, dok ostatak odgovara prvom planu. Kada nova sličica uđe u algoritam u vremenu $t + 1$, svaki piksel se pokušava spariti s nekom od krivulja. Piksel odgovara Gaussovoj distribuciji ako mu je vrijednost unutar 2.5 standardnih devijacija distribucije. Ovaj prag od 2.5 se može podešavati, i ne mora biti isti za sve distribucije, što pomaže kad različiti dijelovi slike imaju različito osvjetljenje, jer načelno slika ima više šumova na dijelovima s jačim osvjetljenjem.

Kada se piksel upari s nekom od distribucija, za njega možemo tada reći pripada li pozadini ili prvom planu. Ako se pak desilo da nije pronađena distribucija kojoj piksel odgovara, tada proglašavamo da piksel pripada prvom planu.

Ažuriranje parametara

Ažuriranje parametara obavlja se slično kao kod detekcije prvog plana. Pikseli se redom pokušavaju upariti s nekom distribucijom.
Ako niti jedna distribucija ne odgovara trenutnoj vrijednosti piksela, zamijenimo najmanje vjerojatnu distribuciju distribucijom kojoj je srednja vrijednost trenutna vrijednost, te koja ima inicijalno visoku varijancu, te nisku težinu.
Ako pak nađemo distribuciju kojoj piksel pripada tada ažuriramo parametre na sljedeći način:

$$\omega_{i,t+1} = (1 - \alpha)\omega_{i,t} + \alpha$$

gdje je α konstantna stopa učenja.

$$\mu_{i,t+1} = (1 - \rho)\mu_{i,t} + \rho X_{t+1}$$

$$\sigma^2_{i,t+1} = (1 - \rho)\sigma^2_{i,t} + \rho(X_{t+1} - \mu_{i,t+1})(X_{t+1} - \mu_{i,t+1})^T$$

gdje $\rho = \alpha\eta(X_{t+1}, \mu_i, \Sigma_i)$
Još treba ažurirati težine svih ostalih distribucija po formuli:

$$\omega_{j,t+1} = (1 - \alpha)\omega_{j,t}$$

Moguća poboljšanja metode

U [3] autori govore o mnogim mogućim načinima poboljšanja ove metode kao što su:
- varijabilan broj distribucija
- korištenje drugih algoritama za inicijalizaciju, ili dozvoljavanje da objekti prvog plana budu prisutni u sekvenci inicijalizacije.
- modifikacije u pravilima i mehanizmima ažuriranja.
- uvođenje promjenjivih stopa učenja.
- brojne modifikacije u detekciji prvog plana.

1.4 Pomućen pristup za oduzimanje pozadine

Ovo je jedna od metoda koje koriste boju za detekciju prvog plana. Postoje mnoge slične metode, no u njima se detekcija prvog plana obavlja na svakoj dimenziji neovisno, te se na koncu spajaju, na primjer binarnim operatorom *ili*. Problem u tome je što lažni pozitivni rezultat u nekoj dimenziji uzrokuje lažni pozitivan rezultat u konačnici. Ova metoda pokušava zaobići taj nedostatak koristeći neki pomućeni operator, npr. Choquet-ov integral.

Detekcija prvog plana korištenjem pomućenog integrala

Uvedimo nekoliko pojmova preuzetih iz [2].

Definicija 1.4.1. *Mjera sličnosti boje je skalar $S_k^C(x,y)$ sa vrijednošću između 0 i 1 u pikselu (x,y) dan s:*

$$S_k^C(x,y) = \begin{cases} \frac{I_k^C(x,y)}{I_k^B(x,y)} & , za\ I_k^C(x,y) < I_k^B(x,y) \\ 1 & , za\ I_k^C(x,y) = I_k^B(x,y) \\ \frac{I_k^B(x,y)}{I_k^C(x,y)} & , za\ I_k^C(x,y) > I_k^B(x,y) \end{cases}$$

gdje je $k \in \{1,2,3\}$ oznaka kanala boje, B i C predstavljaju pozadinu i trenutnu sličicu u vremenu t respektivno.

Napomenimo da do B dolazimo bilo kojom metodom za modeliranje pozadine. Primijetimo također da je vrijednost mjere sličnosti boje blizu jedinice ako su $I_k^C(x,y)$ i $I_k^B(x,y)$ vrlo slični.

Neka je μ pomućena mjera na konačnom skupu kriterija X te $h : X \to [0,1]$ skupovna funkcija.

Definicija 1.4.2. *Choquet-ov integral od h s obzirom na varijablu μ definiran je s:*

$$C_\mu = \sum_{i=0}^{n} h(x_{\sigma(i)})(\mu(A_{\sigma(i)}) - \mu(A_{\sigma(i+1)}))$$

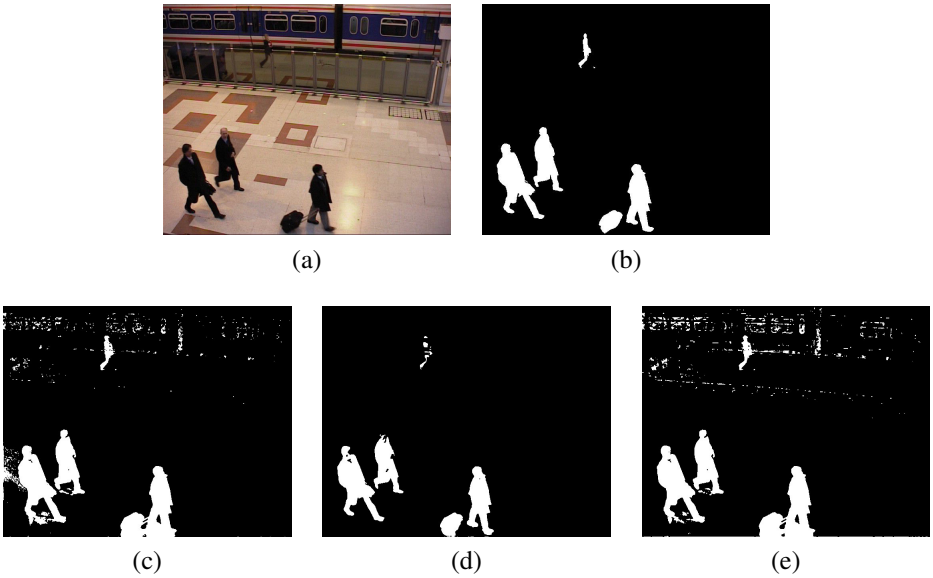
gdje je σ permutacija indeksa takva da $h_{\sigma(1)} \leq ... \leq h_{\sigma(n)}$ i $A_{\sigma(i)} = \{\sigma(1), ..., \sigma(n)\}$

Sada možemo prijeći na detekciju objekata iz prvog plana korištenjem Choquet-ovog integrala. Za svaki piksel, mjere sličnosti boje računamo po 1.4.1. Definiramo skup kriterija $X \in \{x_1, x_2, x_3\}$ s (x_1, x_2, x_3) = tri komponente boje u nekom odabranom spektru (npr. RGB, Ohta, HSV, YCrCb). Za svaki x_i, neka su $\mu(x_i)$ važnosti komponente x_i. pomućena

funkcija $h(x_i)$ definirana je na $[0, 1]$ te $h(x_1) = S_1^C(x, y)$, $h(x_2) = S_2^C(x, y)$ i $h(x_3) = S_3^C(x, y)$. Sada kažemo da piksel na poziciji (x,y) pripada prvom planu ako $C_\mu(x, y) < T$ gdje je T neka vrijednost praga.

Što se tiče ažuriranja pozadine, možemo koristiti slijepo ili selektivno ažuriranje koje se provodi slično kao u metodi tekućeg prosjeka.

Na slici 1.5 možemo vidjeti kako predložena metoda stoji u usporedbi s običnim spajanjem rezultata pomoću logičkog operatora *ili*.



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

Slika 1.5: Učinkovitost pomućene metode: (a) Originalna snimka, (b) Teoretski idealni rezultat, (c) OR-YCrCb, (d) Choquet-YCrCb, (e) Choquet-RGB

Otklanjanje šumova

Promatranja su pokazala da ako je piksel proglašen pripadnikom prvog plana zbog jakog šuma u slici, malo je vjerojatno da će šum također utjecati na njemu susjedne piksele, bilo u prostoru ili vremenu. Da bi djelomično izbjegli utjecaj šumova na metodu možemo računati Choquet-ov integral piksela, te svih njegovih 8 susjeda, zatim uzeti medijan tih 9 vrijednosti te njega testirati na prag.

Poglavlje 2

Oduzimanje pozadine na snimkama s pokretnom kamerom

2.1 Dual-Mode Single Gaussian model sa starošću

Detekcija objekata u pokretu na kamerama koje nisu statične u stvarnom vremenu je zahtjevan problem zbog ograničenosti računalnih resursa, te pokreta kamere. Prilikom dizajniranja metoda za pokretne kamere jako je bitno razmotriti koliko računalnih resursa će metoda zahtijevati. Također je bitno imati na umu ne samo šumove u slici zbog kvalitete kamere, već i greške koje nastaju zbog kompenzacije pokreta kamere. Ova metoda predložena je u [11] te se sastoji od tri glavna dijela, a to su predobrada (eng. preprocessing) u svrhu smanjivanja šumova, modeliranje pozadine, te kompenzacija pokreta kamere. Primjer metode u praksi se može vidjeti na usporednom testu.[1]

Predobrada

Predobrada se obavlja jednostavnim prostornim filtriranjem.

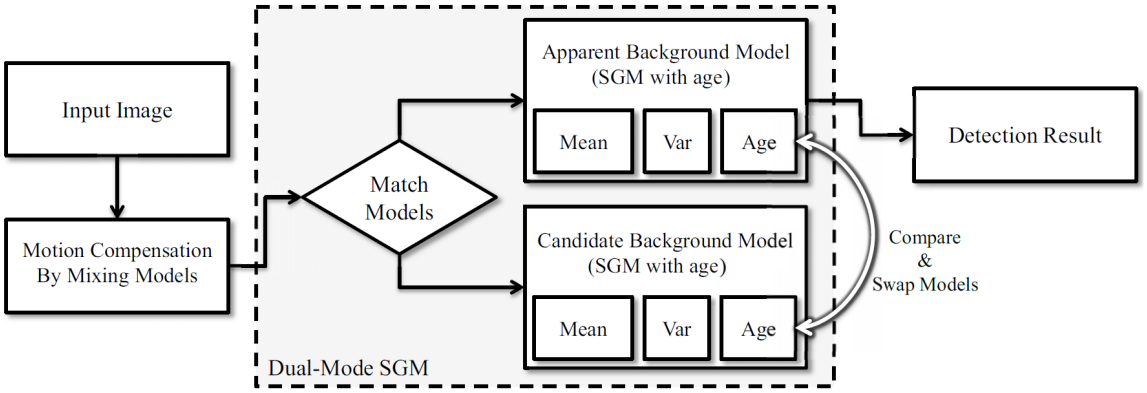
Definicija 2.1.1. *Prostorni filtar je bilo koja operacija na slici u kojoj se vrijednost svakog piksela $I(u, v)$ mijenja po nekoj funkciji njemu okolnih piksela.*

Više o prostornom filtriranju može se vidjeti u [1]

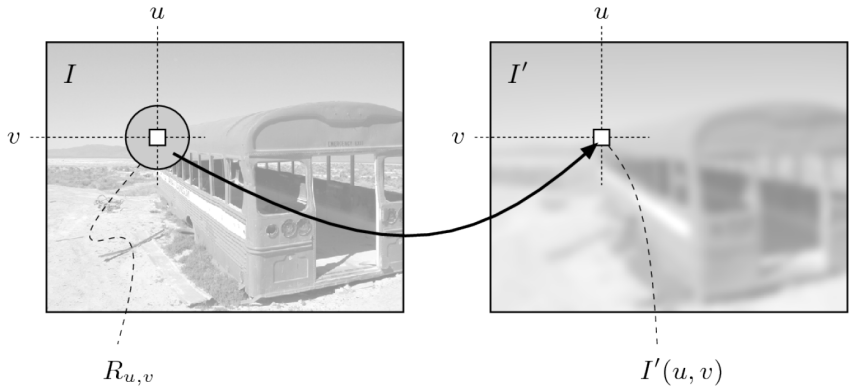
Modeliranje pozadine

Jedan od glavnih razloga koji čine metode sa statističkim modelima nepodobnima za pokretne kamere je što obično imaju fiksnu stopu učenja. Fiksna stopa učenja ima za poslje-

[1] https://www.youtube.com/watch?v=2UOu4OuBYUs



Slika 2.1: Dijagram toka predložene metode



Slika 2.2: Prostorno filtriranje

dicu da se prvo opažanje piksela smatra srednjom vrijednošću za model koji uči u nedogled. To ne uzrokuje kritične probleme za stacionarne kamere jer u mnogim slučajevima vrijednost piksela se gotovo ne mijenja za pozadinske piksele. Međutim, za pokretne kamere, greške u metodama kompenzacije pokreta su neizbježne koliko god te metode bile precizne, te ne možemo pretpostaviti da će prvo opažanje piksela odgovarati srednjoj vrijednosti koju bismo dobili daljnjim opažanjima. Stoga nam je potrebna promjenjiva stopa učenja, koju definiramo uz pomoć starosti.

Za modeliranje scene koristimo Gaussove krivulje. Da bi se dodatno smanjilo opterećenje na računalne resurse, sličice dijelimo na mrežu s kvadratnim poljima jednake veličine, te čuvamo jednu Gaussovu krivulju za svako polje. Ako grupu piksela u polju *i*

u vremenu t označimo s $G_i^{(t)}$, broj piksela u $G_i^{(t)}$ s $\left|G_i^{(t)}\right|$, te opaženi intenzitet piksela j u vremenu t s $I_j^{(t)}$, model ažuriramo na sljedeći način:

$$\mu_i^{(t)} = \frac{\tilde{\alpha}_i^{(t-1)}}{\tilde{\alpha}_i^{(t-1)} + 1}\tilde{\mu}_i^{(t-1)} + \frac{1}{\tilde{\alpha}_i^{(t-1)} + 1}M_i^{(t)} \tag{2.1}$$

$$\sigma_i^{(t)} = \frac{\tilde{\alpha}_i^{(t-1)}}{\tilde{\alpha}_i^{(t-1)} + 1}\tilde{\sigma}_i^{(t-1)} + \frac{1}{\tilde{\alpha}_i^{(t-1)} + 1}V_i^{(t)} \tag{2.2}$$

$$\alpha_i^{(t)} = \tilde{\alpha}_i^{(t-1)} + 1 \tag{2.3}$$

gdje su $\mu_i^{(t)}$ srednja vrijednost, $\sigma_i^{(t)}$ standardna devijacija, te $\alpha_i^{(t)}$ starost modela pripisanog grupi piksela $G_i^{(t)}$, a $M_i^{(t)}$ i $V_i^{(t)}$ su definirani s:

$$M_i^{(t)} = \frac{1}{|G_i|}\sum_{j \in G_i} I_j^{(t)}$$

$$V_i^{(t)} = \max_{j \in G_i}\left(\mu_i^{(t)} - I_j^{(t)}\right)^2$$

Vrijednosti $\tilde{\mu}_i^{(t-1)}$, $\tilde{\sigma}_i^{(t-1)}$, te $\tilde{\alpha}_i^{(t-1)}$ su kompenzirane vrijednosti iz modela za vrijeme $t-1$, u skladu s metodom za kompenzaciju pokreta. Navedena kompenzacija navodi se u kasnijem potpoglavlju.

Prednost predloženog modela sa starošću je u tome što se greške u kompenzaciji pokreta uspješno izbjegavaju na temelju starosti modela umjesto da se nakupljaju u nedogled. Primjećujemo također da su formule za ažuriranje slične onima u ranije opisanim metodama koje koriste Gaussove krivulje.

Korištenje jedne Gaussove krivulje za modeliranje scene obično vrlo dobro radi u jednostavnim slučajevima, međutim, kad se koriste brze stope učenja, pozadinski model se vrlo lako zagadi podacima koji dolaze od piksela u prvom planu što se vidi na slici 2.3(a). Ovaj problem najizraženiji je kod objekata koji se sporo kreću te velikih objekata. S obzirom da u ovoj metodi imamo promjenjivu stopu učenja, često se dešava da imamo brzu stopu učenja. Na primjer, pri inicijalizaciji svi pikseli počinju sa starošću 1, što bi značilo da bi njihova stopa učenja za sljedeću sličicu bila 0.5. Jedno od mogućih rješenja ovog problema već smo komentirali, a sastoji se od selektivnog ažuriranja, odnosno da ažuriramo model samo onda kada je piksel klasificiran kao pozadina. Problem kod te metode je što lažni objekti prvog plana, kao što su vozila koja se dovezu te ostanu parkirana, nikad ne bi bili naučeni.

Da bismo izbjegli taj nedostatak, koristimo dodatni model s Gaussovom krivuljom (u daljem tekstu kandidat) koji djeluje kao kandidat za pozadinu, a originalni model proglašavamo prividnom pozadinom. Kandidat ostaje kandidat sve dok njegova starost ne premaši starost prividne pozadine, a u tom trenu dva modela zamjenjuju mjesta.

Ako označimo srednju vrijednost, standardnu devijaciju, te starost kandidata i prividnog modela u vremenu t za polje i mreže s $\mu_{C,i}^{(t)}$, $\sigma_{C,i}^{(t)}$, $\alpha_{C,i}^{(t)}$ i $\mu_{A,i}^{(t)}$, $\sigma_{A,i}^{(t)}$, $\alpha_{A,i}^{(t)}$, respektivno, tada ako je:

$$\left(M_i^{(t)} - \mu_{A,i}^{(t)}\right)^2 < \theta_s \sigma_{A,i}^{(t)}$$

gdje je θ_s parametar praga, ažuriramo $\mu_{A,i}^{(t)}$, $\sigma_{A,i}^{(t)}$ i $\alpha_{A,i}^{(t)}$ po formulama (2.1), (2.2) i (2.3). Ako pak navedeno ne vrijedi, a istovremeno vrijedi:

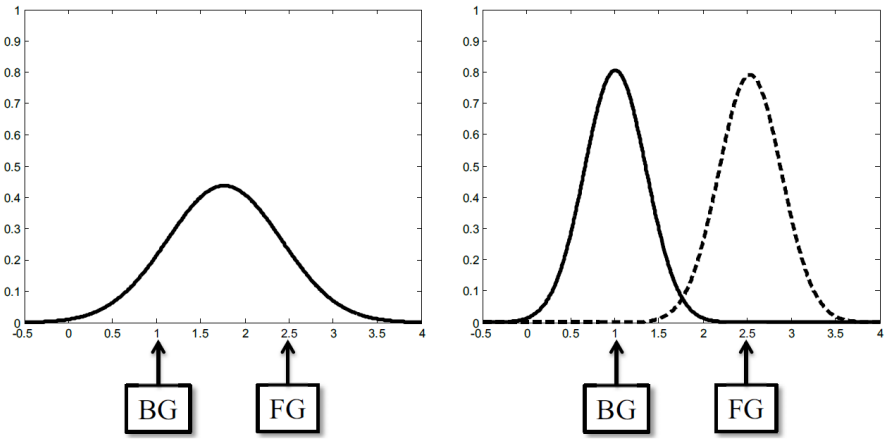
$$\left(M_i^{(t)} - \mu_{C,i}^{(t)}\right)^2 < \theta_s \sigma_{C,i}^{(t)}$$

tada ažuriramo $\mu_{C,i}^{(t)}$, $\sigma_{C,i}^{(t)}$ i $\alpha_{C,i}^{(t)}$ po formulama (2.1), (2.2) i (2.3). Ako pak ne vrijedi niti jedno od toga, tada inicijaliziramo kandidata s trenutnim opažanjem.

Primijetimo da se ažurira samo jedan od dva modela, dok drugi ostaje netaknut. Nakon ažuriranja, dva modela se zamijene ako:

$$\alpha_{C,i}^{(t)} > \alpha_{A,i}^{(t)}$$

Nakon zamjene, kandidat se inicijalizira trenutnim opažanjem.



(a) SGM (b) Dual-Mode SGM

Slika 2.3: Usporedba modela s jednom Gaussovom krivuljom (a) i dvomodela predloženog u ovoj metodi (b)

Konačno za određivanje piksela prvog plana koristimo samo prividni model, koji bi sada trebao biti nezagađen pozadinski model. Na ovaj način sprečavamo zagađivanje pozadinskog modela pikselima prvog plana što se vidi na slici 2.3(b), te na slici 2.4. Podaci

prvog plana naučeni su u kandidatu (na slici označenom crtkano), dok prividna pozadina korektno opisuje stvarnu pozadinu. Također se ne moramo brinuti da se lažni prvi plan nikada neće naučiti u model jer će starost kandidata sigurno prije ili poslije premašiti starost prividne pozadine.

Slika 2.4: Primjer metode s jednom krivuljom u usporedbi s dvomodelom. (a) Pozadinski model s jednom krivuljom, (b) prividna pozadina, (c) kandidat, (d) rezultat metode s jednom krivuljom, (e) rezultat metode s dvomodelom

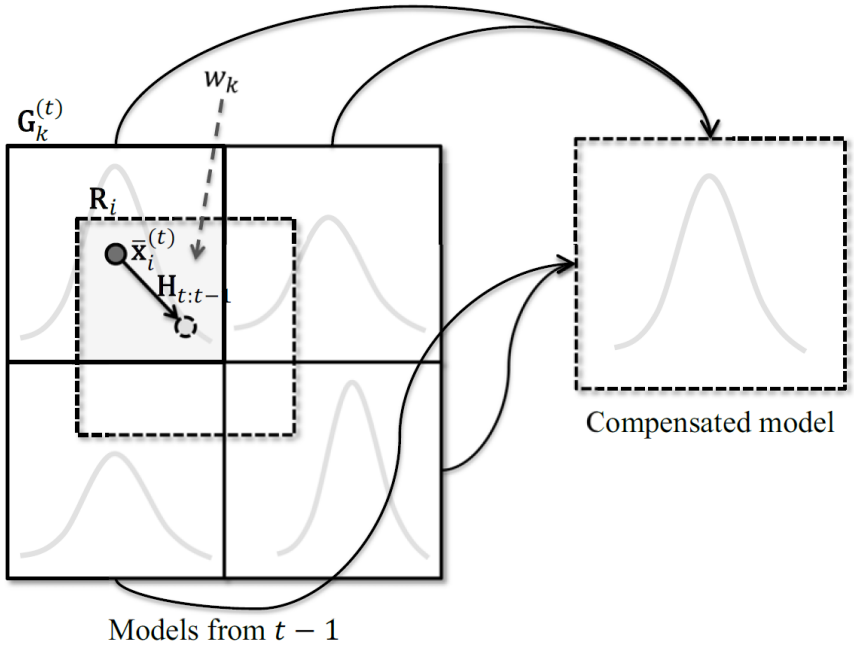
Kompenzacija pokreta miješanjem modela

Za nizove sličica dobivene pokretnom kamerom, model naučen do vremena $t-1$ ne može se koristiti izravno za prepoznavanje prvog plana u vremenu t iz očitih razloga. Da bi model postao iskoristiv, nužna je kompenzacija pokreta. Međutim, s obzirom da koristimo jedan model za sve piksele unutar istog polja u mreži, jednostavno transformiranje pozadinskog modela temeljeno na interpolacijskim strategijama uzrokovalo bi previše grešaka. Stoga, umjesto da jednostavno transformiramo pozadinski model, konstruiramo kompenzirani pozadinski model za vrijeme t stapanjem statistika modela u vremenu $t-1$.

U tom procesu koristimo Kanade-Lucas-Tomasi (KLT) lokator osobina. To je metoda koja koristi informacije o prostornom intenzitetu da bi usmjeravala pretraživanje za mjestom koje daje najbolju podudarnost. Detaljno objašnjenje ove metode izlazi iz opsega ovog diplomskog rada a više o metodi se može pročitati u [10]. Također koristimo Random sample consensus (RANSAC). RANSAC je iterativna metoda za procjenu parametara matematičkog modela iz skupa opažanih podataka koji sadrži opažajne točke koje su udaljene od ostalih. Više o metodi može se pročitati u [4]. Obje metode mogu se implementirati pomoću biblioteke OpenCV, o kojoj se govori u trećem poglavlju.

Provedemo KLT na svakom kutu mreže sličice u vremenu t koristeći sličicu iz vremena $t-1$. S dobivenim rezultatima praćenja točaka provedemo RANSAC da bismo dobili matricu homografije $H_{t:t-1}$ koja opisuje perspektivnu transformaciju svih piksela iz vremena t u piksele iz vremena $t-1$. To smatramo kretanjem pozadine.

Nadalje, označimo poziciju piksela j s x_j, poziciju sredine za $G_i^{(t)}$ s $\bar{x}_i^{(t)}$ te perspektivnu transformaciju točke x u skladu s $H_{t:t-1}$ sa $f_{PT}(x, H_{t:t-1})$. Kao na slici 2.5, za svako polje i



Slika 2.5: Predložena metoda kompenzacije pokreta miješanjem modela

u mreži smatramo da je $\bar{x}_i^{(t)}$ pomaknut, odnosno došao s pozicije $f_{PT}\left(\bar{x}_i^{(t)}, H_{t:t-1}\right)$. Pretpostavimo da nije dolazilo do znatne promjene u skaliranju, to jest zumiranja slike. Neka je $f_{PT}\left(\bar{x}_i^{(t)}, H_{t:t-1}\right)$ sredina kvadrata R_i. Tada je model u vremenu $t-1$ za R_i kompenzirani model za piksele $G_i^{(t)}$. R_i se preklapa s više polja u mreži te miješamo Gaussove krivulje polja s kojima se preklapa u vremenu $t-1$ da bismo dobili $\tilde{\mu}_i^{(t-1)}$, $\tilde{\sigma}_i^{(t-1)}$, te $\tilde{\alpha}_i^{(t-1)}$ koji opisuju kompenzirani pozadinski model. Težine za miješanje se uzimaju tako da budu proporcionalne preklapajučim površinama.

Ako s $O_i^{(t)}$ označimo skup polja koja se preklapaju s R_i, te težine za miješanje s w_k, $k \in O_i^{(t)}$, tada dobivamo kompenzirani pozadinski model na sljedeći način:

$$\begin{aligned}
\tilde{\mu}_i^{(t-1)} &= \sum_{k \in O_i^{(t)}} w_k \mu_k^{(t-1)} \\
\tilde{\sigma}_i^{(t-1)} &= \sum_{k \in O_i^{(t)}} w_k \left[\sigma_k^{(t-1)} + \left(\mu_k^{(t-1)}\right)^2 - \left(\tilde{\mu}_i^{(t-1)}\right)^2\right] \\
\tilde{\alpha}_i^{(t-1)} &= \sum_{k \in O_i^{(t)}} w_k \alpha_k^{(t-1)}
\end{aligned} \tag{2.4}$$

gdje su težine za miješanje dane s:

$$w_k \propto Area\left\{R_i \cap G_k^{(t)}\right\}$$

i

$$\sum_k w_k = 1$$

Za vrijeme procesa miješanja, modeli gdje se obližnja područja jako razlikuju imat će pretjerano visoke standardne devijacije zbog formule (2.4). Zato, nakon kompenzacije, ako je standardna devijacija iznad nekog praga θ_v, smanjimo starost modela po formuli:

$$\tilde{\alpha}_i^{(t-1)} \leftarrow \tilde{\alpha}_i^{(t-1)} e^{-\lambda\left(\tilde{\sigma}_i^{(t-1)} - \theta_v\right)}$$

gdje je λ proizvoljan parametar propadanja. Time sprečavamo da model ima preveliku standardnu devijaciju, što jako pomaže u otklanjanju lažnih prvih planova blizu rubova objekata.

Detekcija prvog plana

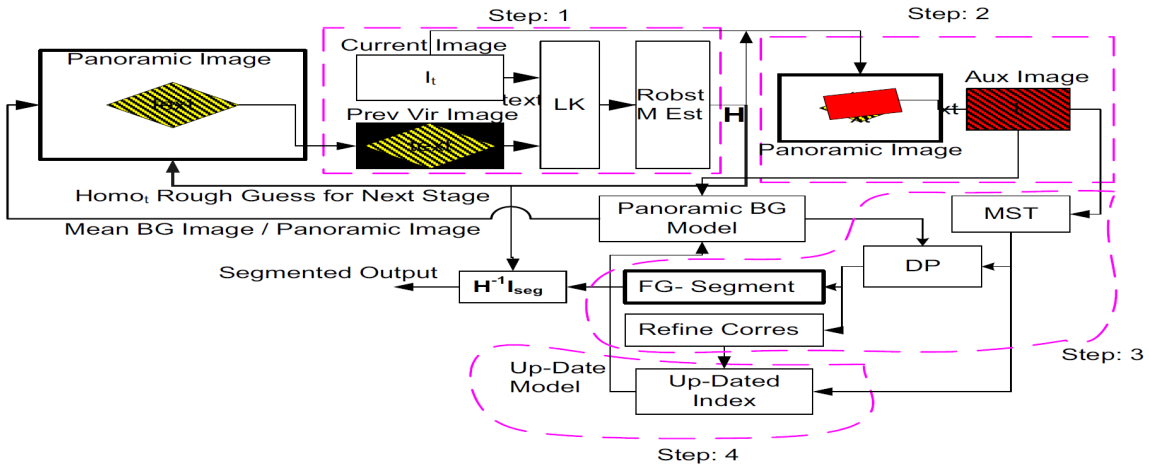
Za detekciju prvog plana koristimo jednostavno uspoređivanje s pragom u odnosu na devijaciju. Pokazalo se da je to dovoljno precizno, a usput se štedi velika količina računalnih resursa. Za svaki piksel j u polju mreže i kažemo da pripada prvom planu ako:

$$\left(I_j^{(t)} - \mu_{A,i}^{(t)}\right)^2 > \theta_d \sigma_{A,i}^{(t)}$$

2.2 Panoramski pozadinski model

Iduća metoda zasniva se na ideji izgradnje modela koji u sebi sadržava informacije ne samo o trenutnom kadru već i o širem području. S obzirom da se kamera smije slobodno kretati, konstruira se panoramski pozadinski model, te je potrebno precizno poravnati uzastopne sličice. Međutim efekt paralakse, izobličenje zbog objektiva, šumovi, neadekvatan model kretanja, efekt podpiksela, te greške u računanju ometaju ostvarivanje stabilnog pozadinskog modela. To je dodatno kompliciranо u vanjskom okruženju zbog dinamične prirode. Bez istodobnog otklanjanja svih tih odstupanja, pozadinski model podbacuje zbog nakupljanja grešaka.

Zbog efekta paralakse te izobličenja objektiva, globalna transformacija između trenutne sličice i pozadinskog modela ne postoji. Na slici 2.6 vidi se tok predložene metode. U prvom koraku približno poravnanje između trenutne sličice i modela se postiže kroz projektivnu transformaciju. Panoramska slika trenutno promatrane scene se generira iz pozadinskog modela tako da se uzme srednja vrijednost najvjerojatnijeg modela, kako bi omogućili



Slika 2.6: Dijagram toka za panoramski pozadinski model

sparivanje. LKT se koristi kako bi se pronašla korespondencija između trenutne sličice i panoramske slike. Da bi se poboljšale performanse LKT-a koristi se prethodno izračunata transformacija za sintezu virtualne slike, te se tada primjenjuje LKT na virtualnu sliku i trenutnu sličicu. Rezultati LKT metode se koriste za M-procjenu[2], te se dobiva projektivna transformacija između trenutne sličice i modela.

U drugom koraku generira se pomoćna slika za kompenzaciju pokreta koja se koristi kao ulaz u idućim koracima. Na temelju procijenjene projektivne transformacije iz prvog koraka, trenutna sličica se transformira u koordinate panorame, te se pomoćna slika dobiva izrezivanjem iz panorame.

U trećem koraku segmentira se prvi plan, te se procjenjuje gusto uparivanje (eng. dense matching) između trenutne sličice i modela.

U četvrtom koraku se ažurira pozadinski model. Cijeli proces je tehnički prilično zahtjevan te se detaljno može proučiti u [7].

[2]https://en.wikipedia.org/wiki/M-estimator

Poglavlje 3

Prijedlog za konkretnu implementaciju automatskog praćenja

3.1 Prijedlog metode

U ovom poglavlju predlaže se metoda za implementaciju automatskog praćenja predavača u učionici. Prvo pitanje koje se nameće je o kakvim uvjetima okoline se radi. Drugo pitanje se odnosi na tehničke zahtjeve.

Uvjeti okoline i tehnički zahtjevi

Ako razmotrimo uvjete prilikom snimanja jednog predavanja na našem fakultetu, zaključujemo da se zapravo radi o vrlo kontroliranim uvjetima. Trajanje snimke je kratko, a neke od učionica su dobro izolirane od vanjskih uvjeta. Stoga možemo pretpostaviti da na snimci neće biti značajnih promjena u osvjetljenju. Također s obzirom da se radi o snimci na zatvorenom prostoru, pozadina će biti strogo statična. Ova dva čimbenika uvelike će nam olakšati pri odabiru metode.

Pretpostavljamo da na raspolaganju imamo kameru koja je sposobna za dvosmjernu komunikaciju s našim programom, dakle da može primati naredbe za pomicanje, te da može obavijestiti program u trenutku kada je pomicanje završeno.

Razrada metode

S obzirom na prilično povoljne uvjete okoline, možemo zaključiti da će za konkretnu implementaciju zapravo biti dovoljna dosta jednostavna metoda. Ideja je da se koristi hibridna metoda, to jest metoda za statičnu kameru, prilagođena za rad s pokretnom kamerom.

Pozadinski model inicijaliziramo vrijednostima piksela na trenutnoj sličici. S obzirom da nam nije cilj opisati točnu siluetu objekata u prvom planu jer ne moramo donositi nikakve dodatne zaključke na temelju oblika, nego nam je dovoljno približno znati u kojem dijelu kadra se dešava kretanje, možemo koristiti najobičniju metodu razlike sličica. Usporedbom dviju sličica dobivamo područja interesa. S obzirom da ne koristimo nikakav pametan način za modeliranje pozadine, područja interesa će biti pikseli koje je predavač zauzimao u prvoj i u drugoj sličici. Ako pretpostavimo da je kamera centrirana na predavača u prvoj sličici nije nam ni bitno što imamo obje informacije. Definiramo srednji dio kadra kao skup piksela koji je udaljen od ruba slike za više od θ_s. Sa G označimo skup svih piksela koji ne pripadaju srednjem dijelu kadra. Sada možemo izračunati prosječnu lokaciju područja interesa po formuli:

$$(x, y)_P = \left(\frac{\sum_{p \in G} p_x}{|G|}, \frac{\sum_{p \in G} p_y}{|G|} \right)$$

gdje su p_x i p_y x i y koordinate piksela iz G.

Kada imamo prosječnu lokaciju područja interesa možemo dobiti željeni vektor za pomicanje kamere kao vektor od sredine sličice do prosječne lokacije područja interesa. Primijetimo da će u područje interesa ući u eventualni šumovi u slici, no to nam nije bitno jer možemo pretpostaviti da će šumovi biti ravnomjerno raspoređeni po kadru.

Na kraju valja još razmisliti o kompenzaciji pokreta kamere. Pretpostavimo da empirijski možemo utvrditi vezu između brzine kretanja kamere i translacije piksela na sličicama. Tada jednostavnim translacijama možemo korigirati usporedbu piksela na dvije uzastopne sličice kada se kamera kreće, a kada kamera pošalje informaciju o prestanku kretanja, kompenzaciju možemo preskakati.

3.2 Biblioteka OpenCV

OpenCV[1] je vrlo popularna biblioteka pod BSD licencom, te je kao takva besplatna za akademsku i komercijalnu upotrebu. Ima sučelja za programske jezike C, C++, Python i Java te podržava operativne sustave Windows, Linux, Mac OS, iOS i Android. Jedna je od najpopularnijih i najsuvremenijih biblioteka na području računalnog vida, te sadrži gotova rješenja za mnoge metode opisane u ovom diplomskom radu, stoga se preporuča korištenje ove biblioteke za bilo kakvu konkretnu implementaciju.

[1] http://opencv.org/

Glavni i dodatni moduli

OpenCV ima oko 30 glavnih te još toliko dodatnih modula. Spomenut ćemo neke od njih.
core - Ovaj modul sadrži osnovne strukture, C strukture i operacije, optimizacijske algoritme, klase za međusobno funkcioniranje s OpenGL-om i DirectX-om, sloj za hardversko ubrzanje, te još neke stvari.
imgproc - Ovo je modul koji sadrži podršku za filtriranje slika, geometrijske transformacije slika, funkcije za crtanje, histograme, analizu pokreta i praćenje objekata, te pronalaženje osobina i objekata na slici.
video - Sadrži podršku za analizu pokreta te praćenje objekata na videu.
ml - U ovom modulu sadržane su klase koje podržavaju područje strojnog učenja, npr. EM-algoritam spomenut u ovom diplomskom radu.
stitching - Daje podršku za spajanje slika.
cuda - Brojni moduli za rad s CUDA API-jem za paralelno računanje
videostab - Daje podršku za stabilizaciju video snimki.
Od dodatnih modula, za područje računalnog vida dosta su bitni moduli **fuzzy**, **optflow**, te **reg**.

Bibliografija

[1] *Spatial Filtering*, (2012), http://www.coe.utah.edu/~cs4640/slides/Lecture5.pdf.

[2] Fida El Baf, Thierry Bouwmans i Bertrand Vachon, *A Fuzzy Approach for Background Subtraction*, ICIP 2008, San Diego, United States. (2008), 2648–2651.

[3] Fida El Baf, Thierry Bouwmans i Bertrand Vachon, *Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey*, Recent Patents on Computer Science, Bentham Science Publishers **1** (2008), br. 3, 219–237.

[4] M. A. Fischler i R. C. Bolles, *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography.*, Commun. ACM **6** (1981), br. 24, 381–395.

[5] P. Z. Graszka, *Background Modeling in Video Sequences*, Advances in Systems Science. Proceedings of the International Conference on Systems Science (2014), 195–205.

[6] Cerman Martin, *Background Subtraction Using Running Gaussian Average: A Color Channel Comparison*, (2014), http://www.prip.tuwien.ac.at/teaching/cvpr_se/Cerman2014.pdf.

[7] N. I. Rao, H. Di, i G. Xu, *Panoramic background model under free moving camera*, Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery **1** (2007), 639–643.

[8] Nishu Singla, *Motion Detection Based on Frame Difference Method*, (2014), http://www.ripublication.com/irph/ijict_spl/ijictv4n15spl_10.pdf.

[9] Zhen Tang, Zhenjiang Miao i Yanli Wan, *Background Subtraction Using Running Gaussian Average and Frame Difference*, Entertainment Computing – ICEC 2007 **4740 of the series Lecture Notes in Computer Science** (2007), 411–414.

[10] C. Tomasi i T. Kanade, *Detection and tracking of point features.*, Technical report, Carnegie Mellon University (1991), https://www.ces.clemson.edu/~stb/klt/tomasi-kanade-techreport-1991.pdf.

[11] Kwang Moo Yi, Kimin Yun, Soo Wan Kim, Hyung Jin Chang, Hawook Jeong i Jin Young Choi, *Detection of Moving Objects with Non-stationary Cameras in 5.8ms: Bringing Motion Detection to Your Mobile Device*, 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2013), 27–34.

Sažetak

U ovom radu upoznajemo čitatelja s osnovama područja detekcije pokreta. Ovo područje spada pod računalni vid koji je jedna od gorućih grana računarstva današnjice. Postoje brojne primjene računalnog vida u robotici, automatizaciji, računalnom sučelju, te mnogim drugima. Izabiru se metode koje su prilično lako shvatljive akademskom čitatelju koji nije imao prijašnjih susreta s područjem, te se spominje jedna kompliciranija radi kontrasta. Govori se o glavnim preprekama i izazovima koje potencijalne metode moraju savladati da bi bile upotrebljive za praktičnu primjenu. Kao zaključak rada navodi se prijedlog jednostavne metode za implementaciju u specifičnim kontroliranim uvjetima. Time se pokazuje da unatoč tome što u području postoji na stotine pristupa, metoda i varijacija ne postoji jedna najbolja za sve primjene. Glavna nit vodilja kod odabira metode ipak na kraju mora biti minimizacija zahtjeva na računalne resurse, a da metoda ipak daje iskoristive rezultate.

Summary

In this thesis we introduce the reader to the basics of motion detection. Motion detection is part of computer vision, which is one of the thriving branches of contemporary computer science. There are many applications for computer vision in robotics, automatization, computer interface, and many others. We choose methods which are easily understandable to academic reader with no previous experience in the field, and we mention one which is a bit more complicated, for contrast. We talk about main obstacles and challenges which potential methods must overcome in order to be viable for application. As a conclusion to thesis we propose a simple method for implementation in controlled conditions. Thus, we show that despite the fact that motion detection has hundreds of approaches, methods and their variations, there isn't a single one that is best for all applications. In general, as we chose a method to use in specific case, we must try to minimize computer resource consumption, while keeping the method results usable for intended purpose.

Životopis

Rođen sam 21. srpnja 1985. godine u Zagrebu. Upisujem osnovnu školu Josipa Račića u Zagrebu 1992. godine. Za vrijeme osnovne škole bavim se plivanjem te hokejem na travi. Upisujem prirodoslovno-matematički razred XIII. gimnazije u Zagrebu 2000. godine. Za vrijeme srednje škole još godinu dana se bavim hokejem na travi, a nakon toga se počinjem baviti borilačkom vještinom Kung-Fu. Godine 2004. završavam gimnaziju te polažem maturu s odličnim uspjehom, te iste godine upisujem Prirodoslovno matematički fakultet u Zagrebu. Godine 2011. upisujem Diplomski sveučilišni studij Računarstva i matematike. Godine 2014./2015. uz studij bavim se samostalnim razvojem komercijalne mobilne aplikacije. Trenutna područja interesa su razvoj softvera za mobilne platforme, umjetna inteligencija, te matematika koja stoji iza svih vrsta igara (sport, računalne, stolne).