

SEMINAR

Primjena histograma boje u računalnom vidu

Izradio:

Tomislav Babić

0036412769

Mentor:

Siniša Šegvić

Sadržaj

| | |
|--|----|
| 1. Uvod | 3 |
| 2. Histogram boje | 4 |
| 3. Integralni histogram | 5 |
| 4. Izvedba detekcije prometnih znakova histogramima boje | 7 |
| 4.1. Implementacija postupka učenja | 8 |
| 4.2. Implementacija postupka detekcije | 9 |
| 5. Rezultati i analiza | 11 |
| 5.1 Primjeri | 15 |
| 6. Zaključak | 17 |
| 7. Literatura | 18 |

1. Uvod

U računalnom vidu, poglavito u području detekcije objekata, jedna od najčešćih korištenih metoda jest kaskada boostanih (ojačanih) Haarovih klasifikatora[1]. Metoda je odlična za detekciju oblika, ali Haarove značajke se koriste isključivo na sivim slikama. Zbog toga se ponekad može dogoditi da dođe do krive detekcije, koja bi se na slikama u boji mogla lako popraviti. Postavlja se pitanje kako i koje značajke koristiti za detekciju na slikama u boji. Kao logičan odgovor, nameće se korištenje histograma boje.

2. Histogram boje

U obradi slike, histogram boje se koristi za mjeru distribucije boja u slici. Kod digitalnih slika, histogram boje se zapravo može predstaviti kao niz spremnika (eng. bin) gdje svaki spremnik sadrži broj slikovnih elemenata (piksela) koji se nalaze u određenom rasponu vrijednosti. Na slici 1. možemo vidjeti histogram izgrađen u prostoru crvene i plave boje.

| | | red | | | |
|------|---------|------|--------|---------|---------|
| | | 0-63 | 64-127 | 128-191 | 192-255 |
| blue | 0-63 | 43 | 78 | 18 | 0 |
| | 64-127 | 45 | 67 | 33 | 2 |
| | 128-191 | 127 | 58 | 25 | 8 |
| | 192-255 | 140 | 47 | 47 | 13 |

Slika 1. Histogram u prostoru crvene (red) i plave (blue) boje. Slika preuzeta s wikipedije [5].

Histogram boje se može napraviti u bilo kojem prostoru boja, iako će se u ovom radu koristiti prostor boja RGB. U tom slučaju histogram se, umjesto kao dvodimenzionalna matrica (slika 1.), može predstaviti kao trodimenzionalno polje.

Činjenica da histogram daje informacije o distribuciji boja u slici mu daje neke prednosti i mane prilikom detekcije i prepoznavanja objekata. Zbog toga primjena histogra može biti korisna kod objekata čija se pozicija ili rotacija u sceni ne poznaje. Isto tako možemo zaključiti da je najveća mana korištenja histograma to što ignorira oblik objekata.

3. Integralni histogram

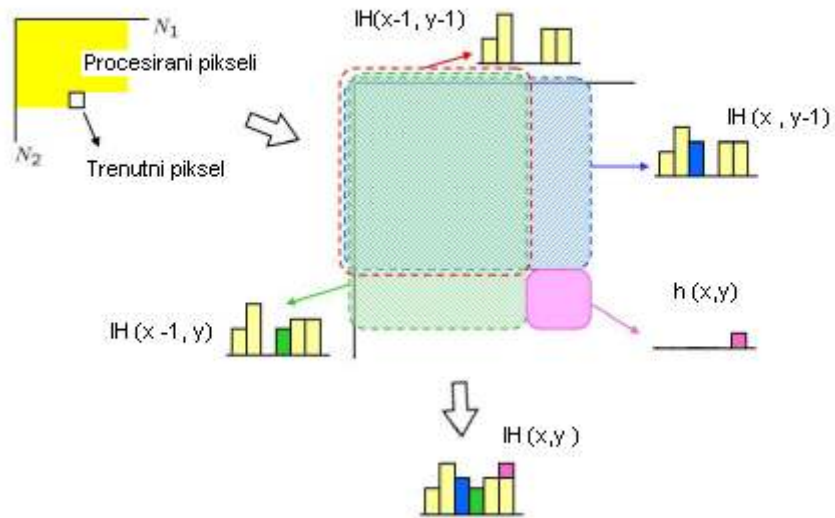
Kod detekcije objekata čija je pozicija u sceni (slici) nepoznata, potrebno je pomičnim detekcijskim oknom, sustavno pretražiti sliku. Okno određene veličine se pomiče kroz cijelu sliku. Nakon što je prošlo cijelu sliku, okno se povećava te ponovno počinje od početka slike. Postupak se ponavlja sve dok okno ne dosegne određenu specificiranu veličinu (primjerice okno se povećava dok ne prekrije cijelu sliku). Na svakoj poziciji okna detekcijskim postupkom se provjerava postoji li traženi objekt u slici. Računanje histograma iznova za svaku poziciju detekcijskog okna je vrlo skup postupak. Zato se uvodi koncept integralnog histograma[2], po uzoru na integralnu sliku[1].

Integralni histogram se može predstaviti kao dvodimenzionalna matrica dimenzija slike za koju se i računa. Tada je svaki element u toj matrici zapravo izgrađeni histogram slike od njezinog početnog elementa do pozicije elementa koji odgovara poziciji u matrici. Označimo s $IH(x, y)$ element integralnog histograma na poziciji x i y , a $H(x_1, y_1, x_2, y_2)$ histogram dijela slike izgrađen između dvije točke čije pozicije odgovaraju pozicijama x_1 i y_1 , odnosno x_2 i y_2 . Tada vrijedi:

$$IH(x, y) = H(0, 0, x, y). \quad (1)$$

Označimo s $h(x,y)$ funkciju koja računa histogram za jedan slikovni element na poziciji x , y u slici.. Tada se histogram slike može računati i propagacijom na sljedeći način:

$$IH(x,y) = IH(x, y-1) + IH(x-1, y) - IH(x-1, y-1) + h(x,y). \quad (2)$$



Slika 2. Na slici možemo vidjeti ilustraciju propagacije histograma. Slika preuzeta iz [2].

Kada iz integralnog histograma želimo dobiti histogram slike počevši od točke x_1, y_1 do točke x_2 i y_2 , tada se to može napraviti na sljedeći način:

$$H(x_1, y_1, x_2, y_2) = IH(x_2, y_2) - IH(x_1-1, y_2) - IH(x_2, y_1-1) + IH(x_1-1, y_1-1). \quad (3)$$

4. Izvedba detekcije prometnih znakova histogramima boje

U radu je implementiran iznimno jednostavan postupak učenja. Na skupu slika za učenje nalaze se isključivo slike prometnih znakova opasnosti. Za svaku sliku u skupu slika za učenje računa se njezin histogram boje. Na kraju se izračuna prosječna vrijednost svih histograma boje, te standardna devijacija. Dobiveni histogrami se spremaju, kako bi se kasnije mogli koristiti u postupku detekcije.

U postupku detekcije, na skupu slika za detekciju pomičnim oknom se prolazi kroz svaku sliku. Za svaku poziciju okna računa se histogram slike, te uspoređuje s histogramima dobivenim u postupku učenja.

Označimo s D_i , i -ti spremnik u histogramu detekcijskog okna, A_i , i -ti spremnik u prosječnom histogramu, te s S_i , i -ti spremnik u histogramu standardne devijacije. Neka je c_i rezultat usporedbe između i -tih spremnika. Sada možemo na sljedeći način izgraditi klasifikator:

$$c_i = \begin{cases} 1, & \text{ako } D_i < (A_i \pm S_i) \\ 0, & \text{inače} \end{cases} \quad (4)$$

Prilikom usporedbe moguće je specificirati broj spremnika koji ne moraju zadovoljavati kriterij usporedbe. Neka je n broj spremnika u svim histogramima, a d specificirani broj klasifikatora koji ne moraju zadovoljavati kriterij usporedbe. Prethodno opisani klasifikator C , se onda može zadati i ovako:

$$C = \begin{cases} 1, & \text{ako } (n-d) \leq \sum_{i=0}^n c_i \\ 0, & \text{inače} \end{cases} \quad (5)$$

Pomoću parametra d se može kontrolirati osjetljivost postupka. Što je taj broj veći, to će biti i veći broj prijavljenih detekcija.

4.1. Implementacija postupka učenja

Na početku je implementirana klasa `ColorHist` u koja služi za predstavljanje histograma boje. Klasa je definirana brojem spremnika, veličinom raspona spremnika i samim spremnicima. Deklaracija klase:

```
class ColorHist{
public:
    int nBin;           //number of bins in histogram for a color
    double rangeBin;
    vector<float> bin;

public:
    ColorHist();
    ColorHist(int n);
    ~ColorHist();

    void createBins( int n);
    void calculate(const img_wrap& imgSrc);
    void increaseBin (int r, int g, int b);
    void normalize(int width, int height);
    void addHistogram( ColorHist &hist);
    void subHistogram( ColorHist &hist);
    void mulHistogram( float num);
    void powHistogram( float num);
    void absHistogram ();
    int cmpHistogram (ColorHist &hist);
};
```

Kao što se može vidjeti u klasi su definirane javne funkcije za stvaranje spremnika, izračun histograma za zadanu sliku, povećanje vrijednosti odgovarajućeg spremnika histograma, dodavanje i oduzimanje dva histograma, množenje histograma s konstantnom vrijednošću, potenciranje histograma, izračun apsolutne vrijednosti histograma, te usporedba dva histograma.

Sam postupak učenja implementiran je u komponenti `alg_histLrn` ljuske `cvsh`. U komponenti su privatno definirani objekti `*sumHist` i `*sqSumHist` tipa `ColorHist` koji služe za izračun sume svih histograma, odnosno sume kvadrata svih histograma. Također je privatno definirana putanja adrese na koju se žele spremiti rezultati varijablom `path`, kao i broj slika `n_imgs` koje se obrađuju, te broj spremnika `nBins` za histograme.

Pozivom funkcije `process(...)` započinje postupak učenja. Na početku se za svaku sliku računa histogram, koji se zatim normalizira s obzirom na veličinu slike. Također se računa

kvadrat normaliziranog histograma. Normalizirani histogram se pridodaje sumi histograma za sve prethodno obrađene slike, a njegov kvadrat se dodaje sumi kvadrata histograma. Nakon što su sve slike obrađene. Poziva se funkcija `finish(...)`, koja računa prosječnu vrijednost histograma, varijancu i standardnu devijaciju, te iste sprema na zadanu adresu.

4.2. Implementacija postupka detekcije

Za implementaciju postupka detekcije korištena je klasa `ColorHist` koja je opisana u prethodnoj točki ovog rada. Uz nju je definirana i klasa `class IntegralHist` kojom je implementiran integralni histogram. Deklaracija klase:

```
class IntegralHist
{
public:
    int width;
    int height;
    vector <ColorHist> elem;

    IntegralHist (const img_wrap& imgSrc, int nBins);
    IntegralHist (int x, int y, int nBins);
    ~IntegralHist ();

    void calculate (const img_wrap& imgSrc);
    void computeIntersection(const rect rc, ColorHist &hist);
};
```

Uz konstruktore i destruktore u klasi su definirane i dvije funkcije. Pozivom funkcije `calculate(...)` će se izračunati integralni histogram za zadanu sliku kao što je opisano u (2). Funkcija `computeIntersection(...)` prima referencu na rezultatni histogram, a po objektu `rc` prima podatke na kojem području se nalazi detekcijsko okno. Zatim se u skladu s (3) izračuna rezultat tj. traženi histogram.

Za detekciju histogramima je odgovorna komponenta `alg_histDet` u ljusci `cvsh`. U konstruktoru te komponente se učitava prosječni histogram i standardna devijacija histograma koji su dobiveni u postupku učenja.

Pozivom funkcije `process(...)` može započeti detekcija na pojedinoj slici. Ta funkcija će pozvati funkciju odgovornu za detekciju, te primiti njezine rezultate i predati ih dalje ljusci `cvsh`. Sam postupak detekcije se obavlja unutar funkcije:

```
void detectHist( img_wrap & img, ColorHist *avgHist, ColorHist *stdDev,
                vector<rect> &detections, double scaleFactor)
```

Funkcija prima sliku koja se treba obraditi, prosječni histogram i standardnu devijaciju, vektor u koji se spremaju rezultati tj. detekcije, te faktor kojim se okno povećava. Postupak započinje s računanjem integralnog histograma slike, te definiranjem maksimalne vrijednosti veličine okna. Zatim se ulazi u petlju:

```
for (double scale = 1.0; scale <= scaleLimit; scale = scale * scaleFactor)
```

U svakom novom prolazu kroz petlju, detekcijsko okno se povećava za zadani faktor. Nakon što mu je definirana veličina, okno zatim započinje s pomicanjem kroz sliku po x i y osi, sve dok ne dođe do kraja slike. Pomicanje okna kroz sliku je definirano s dvije petlje.

Za svaku poziciju okna poziva se funkcija `computeIntersection(...)` koja iz integralnog histograma izračuna histogram okna. Histogram se zatim normalizira s obzirom na veličinu okna. Nakon toga je potrebno izračunati udaljenost histograma okna i prosječnog histograma. To se učini tako da se od histograma okna oduzme prosječni histogram, a zatim se od rezultata izračuna apsolutna vrijednost. Dobiveni rezultat će biti histogram u kojem je u svakom spremniku vrijednost jednaka apsolutnoj razlici odgovarajućih spremnika u histogramu okna i prosječnom histogramu. Takav rezultat je onda potrebno usporediti s histogramom standardne devijacije.

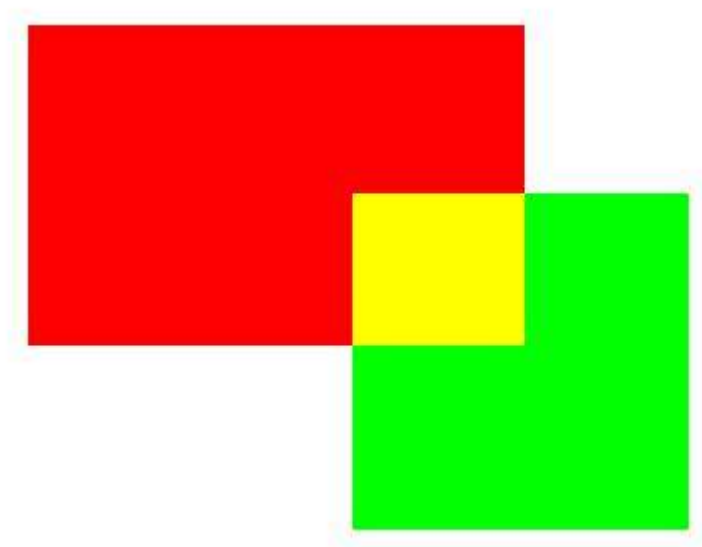
Ukoliko su svi spremnici rezultata manji od odgovarajućih spremnika u standardnoj devijaciji, onda se u oknu nalazi traženi objekt. Podaci o poziciji objekta se tada spremaju u objekt `detections`.

Po završetku rada funkcije `detectHist()` završava i postupak detekcije za zadanu sliku. Na kraju se još samo vrši grupiranje detekcija koje se nalaze u istom području, kako bi se smanjio broj prijavljenih detekcija.

5. Rezultati i analiza

Pri analizi rezultata korištena su dva skupa slika. Skup za učenje se sastoji od 344 slike na kojima ima 428 znakova. Skup za detekciju se sastoji od 332 slike na kojima ima 354 znakova. Slike su dimenzija 720x576, međutim pretraživanje se izvodilo samo interesantnom dijelju slike, dimenzija 360x288. Iz originalne slike su izbačena lijeva polovica te gornja i donja četvrtina slike.

Veličina histograma koji se koristio iznosila je 8x8x8 spremnika. Iz toga slijedi da je veličina integralnog histograma iznosila: $360 \times 288 \times 512 = 50,625 \times 2^{20}$. Prosječna duljina obrade jedne slike iznosila je 3.3 sekunde. Korišteno računalo je Intelov quad-core procesor takta 2.4 Ghz s 4GB radne memorije.



Slika 3. Na slici je prikazana mogući odnos između traženog objekta na slici i prijavljene detekcije. Crveni pravokutnik predstavlja traženi objekt, a zeleni pravokutnik detekciju. Tada žuti pravokutnik predstavlja površinu preklapanja.

Vrlo često detekcije neće pronaći traženi objekt u cijelosti, već samo njegov dio. Takav slučaj je prikazan na slici 3. Stoga su za klasifikaciju detekcija korištene četiri kategorije čiji je opis dan u nastavku (u zagradama su oznake koje će se koristiti u tabličnim prikazima rezultata):

Ispravna detekcija (T) – detekcija će se svrstati u ovu kategoriju, ukoliko površina preklapanja detekcije i traženog objekta (znaka) prekriva bar 80% površine objekta i 60% površine detekcije. Ukoliko više ispravnih detekcija pronade jedan te isti znak, bit će prijavljena samo jedna od njih.

Djelomična detekcija (P) – detekcija će se svrstati u ovu kategoriju ukoliko se središte detekcije nalazi unutar objekta ili ako površina preklapanja detekcije i objekta prekriva bar 10% površine objekta. Ispravna detekcija ne može biti djelomična detekcija. Ukoliko više djelomičnih detekcija pronade jedan te isti znak, bit će prijavljena samo jedna od njih.

Pogrešna detekcija (F) – ukoliko detekcija ne zadovoljava kriterije za ispravnu ili djelomičnu detekciju, tada se svrstava u ovu kategoriju.

Promašaj (M) – ukoliko objekt nije detektiran, tada je riječ o promašaju.

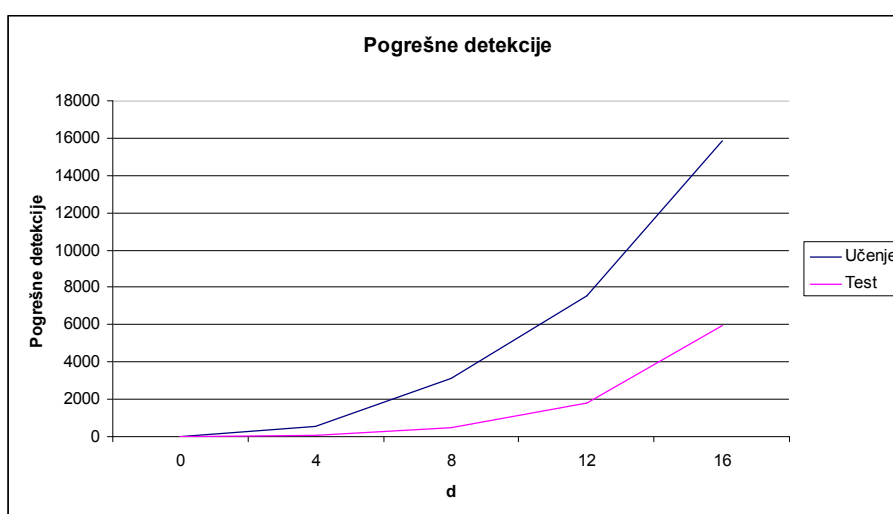
U sklopu 4. poglavlja smo već spomenuli da se može kontrolirati osjetljivost postupka detekcije. Povećanjem broja spremnika (parametar d) koji ne trebaju zadovoljavati kriterij usporedbe s histogramima dobivenim u postupku učenja se povećava i osjetljivost (5). U tabeli 1. možemo vidjeti podatke o detekcijama na skupu za učenje, dok u tabeli 2. možemo vidjeti podatke na skupu za testiranje. Iz obje tablice možemo vidjeti da s porastom parametra d , raste i broj ispravnih i djelomičnih detekcija, ali da također iznimno brzo raste i broj pogrešnih detekcija. Porast pogrešnih detekcija se također lijepo vidi na slici 4.

| T | P | F | M | UKUPNO | D |
|----------|----------|----------|----------|---------------|----------|
| 0 | 4 | 4 | 424 | 428 | 0 |
| 11 | 122 | 585 | 295 | 428 | 4 |
| 54 | 251 | 3118 | 123 | 428 | 8 |
| 71 | 282 | 7544 | 75 | 428 | 12 |
| 79 | 305 | 15847 | 4 | 428 | 16 |

Tabela 1. Rezultati na skupu za učenje. T- ispravne detekcije. P- djelomične detekcije. F- pogrešne detekcije. M - promašaji.

| T | P | F | M | UKUPNO | D |
|----|-----|------|-----|--------|----|
| 0 | 0 | 0 | 354 | 354 | 0 |
| 6 | 25 | 51 | 323 | 354 | 4 |
| 36 | 140 | 464 | 178 | 354 | 8 |
| 66 | 215 | 1801 | 73 | 354 | 12 |
| 74 | 254 | 5925 | 26 | 354 | 16 |

Tabela 2. Rezultati na skupu za testiranje. T- ispravne detekcije. P- djelomične detekcije. F- pogrešne detekcije. M - promašaji.



Slika 4. Graf prikazuje ovisnost pogrešnih detekcija o parametru d.

Kao mjera kvalitete nekog sustava mogu se koristiti odziv i preciznost. Željeli bismo pokazati također pokazati kako te mjere ovise o broju djelomičnih detekcija. Označimo s T broj ispravnih detekcija, s P broj djelomičnih detekcija, s F broj pogrešnih detekcija, te s M broj promašaja. Tada se odzivi (R_t i R_p) i preciznost (P_t i P_p) mogu računati na sljedeće načine:

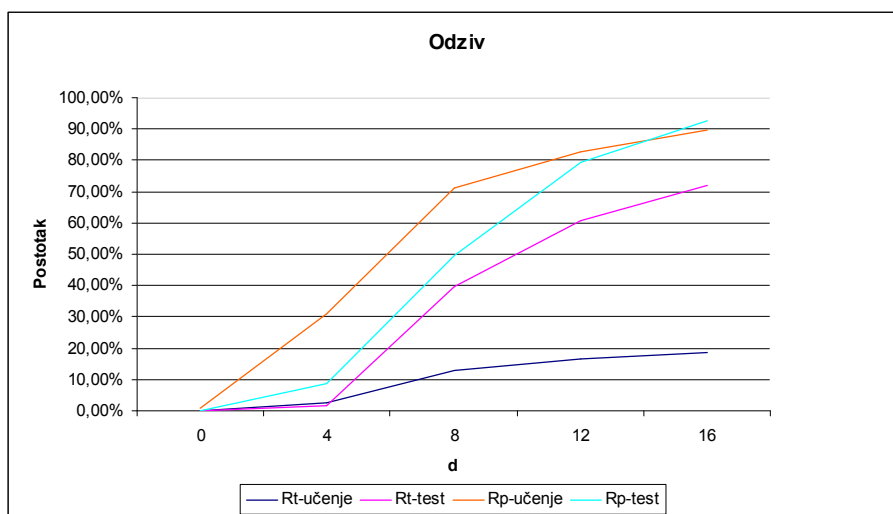
$$R_t = \frac{T}{T + P + M} \quad (6)$$

$$R_p = \frac{T + P}{T + P + M} \quad (7)$$

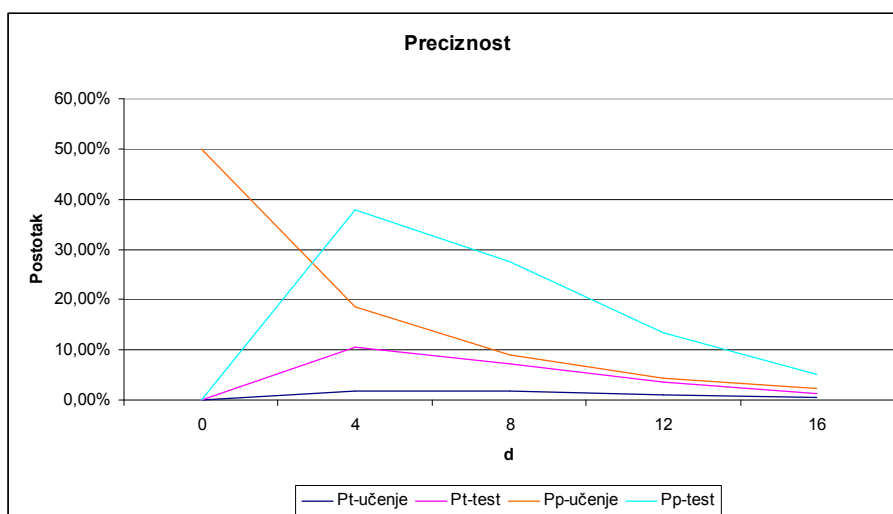
$$P_t = \frac{T}{T + F} \quad (8)$$

$$P_p = \frac{T + P}{T + P + F} \quad (9)$$

Na slici 5. možemo vidjeti kako se s promjenom parametra d mijenja odziv. Na slici 6. možemo vidjeti kako se s promjenom d mijenja broj pogrešnih detekcija mijenja preciznost.



Slika 5. Graf prikazuje ovisnost odziva o promjeni parametra d . Oznaka Rt znači da su se djelomične detekcije pribrojile ispravnim detekcijama u izračunu odziva (6). Oznaka Rp znači da su se djelomične detekcije pribrojile promašajima (7).

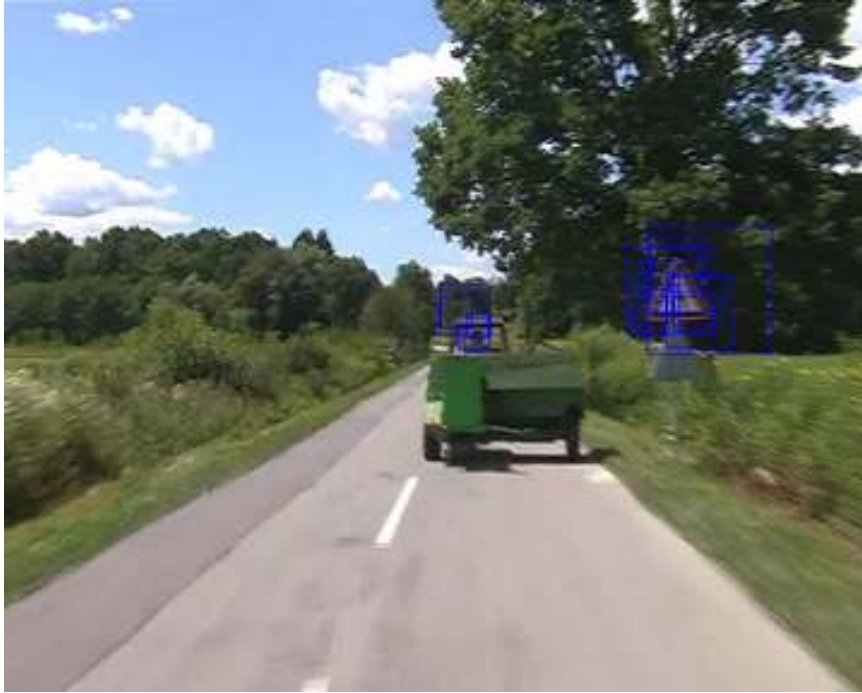


Slika 6. Graf prikazuje ovisnost preciznosti o promjeni parametra d . Oznaka Pt (8) znači da su se djelomične detekcije pribrojile ispravnim detekcijama u izračunu preciznosti. Oznaka Pp znači da su se djelomične detekcije pribrojile pogrešnim detekcijama (9).

Iz priloženih podataka i slika možemo vidjeti kako se kao najbolje vrijednosti za parametar d nameću vrijednosti u intervalu od 8 do 12. Tada je odziv Rp relativno visok (oko 80%), a preciznost još nije drastično pala zbog uslijed povećanja broja pogrešnih detekcija.

5.1 Primjeri

Svi primjeri u nastavku su dobiveni korištenjem parametra $d = 12$. Na slici 6. možemo vidjeti primjer uspješne detekcije prometnog znaka. Također možemo vidjeti i velik broj djelomičnih detekcija. Također možemo vidjeti i primjer pogrešne detekcije koja se vrlo često javlja, iako je područje dominantno zelene boje. Razlog tome je što zbog parametra d nije nužno da se svi spremnici u histogramu detekcije poklapaju s naučenim histogramom, pa se to najviše odnosi na spremnike crvene i zelene boje. Također u samom postupku učenja vrlo često u pozadini znakova nalaze se tamno zelene boje prirode.



Slika 7. Primjer uspješne detekcije znaka i pogrešne detekcije.

Na slici 7. možemo vidjeti velik broj tipičnih pogrešnih detekcija na prijelazu krova kuće. Razlog tome je naravno što će postupak detekcije biti osjetljiv na pojavljivanje kombinacija crvene i bijele ili svijetlo sive boje. Takve kombinacije boja se pojavljuju upravo kod prometnih znakova. Također detekcije se javljaju u neposrednoj blizini ili čak na samom znaku, stoga se neke od njih mogu okarakterizirati kao djelomične detekcije. Dotični znak je teško detektirati zbog žute, umjesto bijele unutrašnjosti. To vrijedi i za ostale znakove s žutom unutrašnjosti.



Slika 8. Primjer pogrešnih detekcija na krovovima.

Na slici 8. možemo vidjeti primjer promašenog znaka. Razlog promašaja je u tome što se znak nalazi u sjeni te su tipične boje prometnog znaka opasnosti puno slabije izražene.



Slika 9. Primjer promašenog znaka.

6. Zaključak

U sklopu ovoga rada, cilj je bio vidjeti kako i koliko dobro se histogrami boje mogu koristiti za detekciju prometnih znakova. Metoda osmišljena i napravljena u ovom radu je iznimno jednostavna, bez ikakvog korištenja naprednih metoda u raspoznavanju uzoraka, umjetnoj inteligenciji ili strojnom učenju. Rezultat toga je značajan broj pogrešnih detekcija, dok broj ispravnih detekcija nije izrazito velik. Međutim, metoda je vrlo često pronalazila dijelove znaka, što se vidi u broju djelomičnih detekcija.

Uz to valja spomenuti kako zbog velikog broja spremnika unutar histograma metoda postaje jako zahtjevna vremenski i prostorno. Prostorna složenost se može zaobići korištenjem manjeg broja spremnika i korištenjem manjih slika. Vremenska složenost se može zaobići tehnikama paralelnog programiranja. Naime, spremnici unutar histograma ne ovise jedan o drugome, osim u postupku izrade integralnog histograma.

U radu se moglo vidjeti kako histogrami donose dobru informaciju o bojama, ali isto tako da nemaju nikakvu informaciju o obliku traženog objekta. U kombinaciji s naprednijim metodama, kao što je već spomenuto boostanje, histogram boje svakako može poslužiti kao korisna i učinkovita značajka.

7. Literatura

- [1] Viola, P., Jones, M., Robust Real-Time Face Detection, International Journal of Computer Vision 57(2), 137–154, <http://lear.inrialpes.fr/people/triggs/student/vj/viola-ijcv04.pdf>, 2004.
- [2] Porikli F., Integral histogram: a fast way to extract histograms in cartesian space, s Interneta, <http://www.merl.com/papers/docs/TR2005-057.pdf> , 2005.
- [3] Laptev, I., Improving object detection with boosted histograms, Image and Vision Computing 27 pp. 535–544, http://www.irisa.fr/vista/Papers/2009_ivc_laptev.pdf, 2009.
- [4] Mauthner, T., Bischof, H., A Robust Multiple Object Tracking for Sport Applications, s Interneta, <http://www.icg.tugraz.at/Members/mauthner/Publications/OAGM07>, 2007.
- [5] Color Histogram – Wikipedia, http://en.wikipedia.org/wiki/Color_histogram, 2007.

Primjena histograma u računalnom vidu

Sažetak

U ovom radu se razmatra mogućnost korištenja histograma boje u računalnom vidu, konkretno u području detekcije objekata odnosno prometnih znakova. Implementirana je jednostavne metode učenja i detekcije. Također je opisan i implementiran koncept integralnog histograma kao metode za brzu izgradnju histograma. U sklopu rada također je napravljena analiza dobivenih rezultata.

Ključne riječi:

Histogram, boja, računalni vid, prometni znakovi, detekcija, integralni histogram

Application of color histogram in computer vision

Abstract

This paper considers possibility of using color histograms in computer vision, specifically in the domain of object detection with reference to traffic sign detection. Simple methods of learning and detection were implemented. Also, the concept of integral histogram, as a method of fast histogram calculation, was described and implemented. Evaluation and analysis was done, as well.

Key words:

Histogram, color, computer vision, traffic signs, detection integral histogram.