SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 854

Detekcija prometnih znakova primjenom Houghove transformacije

Antonija Novokmet

Zagreb, srpanj 2009.

Sadržaj

1. Uvod

Računalni vid je područje umjetne inteligencije čiji je glavni cilj izgradnja umjetnog sustava koji dobiva različite informacije sa slika. Čovjeku je ova vještina prirođena i jednostavna, dok računalima predstavlja popriličan problem. Računalni vid je zanimljivo i široko područje koje pokušava izgraditi sustave za: kontrolu procesa (autonomna vozila, industrijski roboti), prepoznavanje događaja (video nadzor), organizaciju informacija (indeksiranje slika), modeliranje objekata, interakciju računala i čovjeka.

Čest problem u području računalnog vida je određivanje lokacije, orijentacije ili broja određenih objekata. Primjer takovog problema je prepoznavanje prometnih znakova. Prepoznavanje prometnih znakova je tek jedno od potpodručja računalnog vida. Primjene prepoznavanja znakova su u sustavima za pomoć vozačima (eng. *Dirver's Aid – DSS*), autonomnim vozilima (eng. *automated surveillance)*.

Jedna od metoda koja se koristi za prepoznavanje prometnih znakova je Houghova transformacija. Pomoću Houghove transformacije moguće je prepoznavati razne geometrijske oblike. Ovaj rad se bavi kružnom Houghovom transformacijom, te njezinom prilagodbom i primjenom za prepoznavanje okruglih prometnih znakova. Rad ukratko obrađuje različite verzije implementacije Houghove transformacije, te različite filtre koji se koriste za pretprocesiranje slika za Houghovu transformaciju.

2. Houghova transformacija

Houghova transformacija je tehnika pomoću koje se mogu prepoznavati parametarske krivulje. Metoda uspješno detektira nesavršene instance objekata odgovarajuće klase oblika postupkom glasanja.

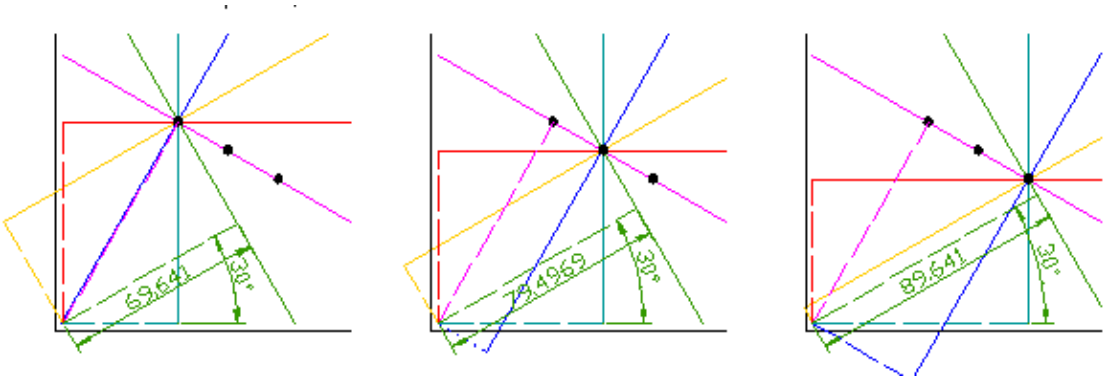
Klasična Houghova transformacija se bavila prepoznavanjem linija u slikama. Kasnije se metoda proširila na prepoznavanje proizvoljnih oblika, najčešće kružnica ili elipsa. Transformaciju za proizvoljne oblike, tj. generaliziranu Houghovu transformaciju (*eng. generalized Hough transform*) izmumili su Richard Duda i Peter Hart 1972. godine. Ova transformacija postaje poznata u području računalnog vida nakon članka "*Generalizing the Hough transform to detect arbitrary shapes*" autora Dana H. Ballard. [9]

2.1. Houghova transformacija za linije

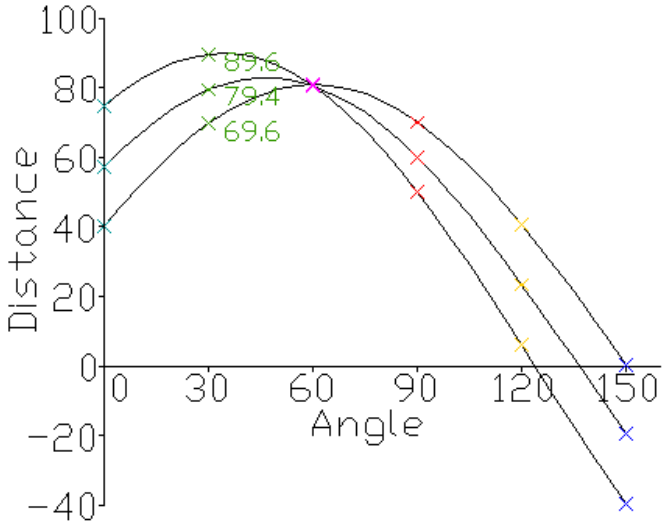
Svaki pravac možemo opisati sa dva parametra: kutom nagiba – θ , te udaljenosti od ishodišta – r. Prema tome pravac možemo zapisati kao:

$$x \cos\theta + y \sin\theta = r \tag{1}$$

Kroz svaki piksel na koordinatama (xi,yi) može proći više linija koje zadovoljavaju formulu (1). Piksel za svaku od tih linija izračunava θ_i i r_i, te povećava akumulacijsko polje na mjestu (θ_i, r_i). Svaki piksel se tim postupkom transformira u jednu krivulju. Svaki od piksela linije će između ostalog glasati i za liniju na kojoj se nalaze, tj. maksimumi akumulacijskog polja određuju parametre linije θ_i i r_i.

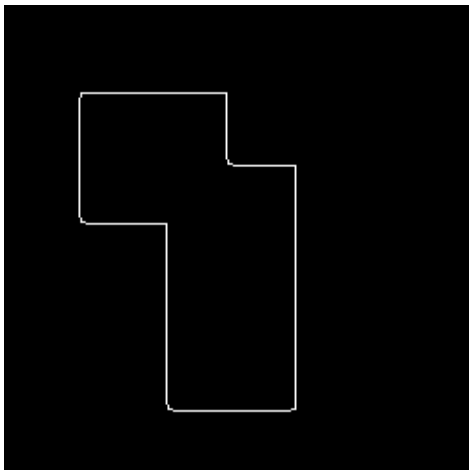


Slika 1 – [9]



Slika 2 – [9]

Slika 1 prikazuje „liniju“ od tri piksela. Svaki graf Slike 1 pokazuje glasove jednog piksela. Istom bojom su obojane linije pod istim kutom. Slika 2 prikazuje akumulacijsko polje dobiveno Houghovom transformacijom za linije Slike 1. Svaka krivulja Slike 2 su glasovi jedne točke. Točka u kojoj se sijeku krivulje (θ= 60, r= 80) određuje parametre linije.



Slika 3 – Ulazna slika za Houghovu transformaciju

Slika 4 – Akumulacijsko polje Slike 3

2.2. Houghova transformacija za kružnice

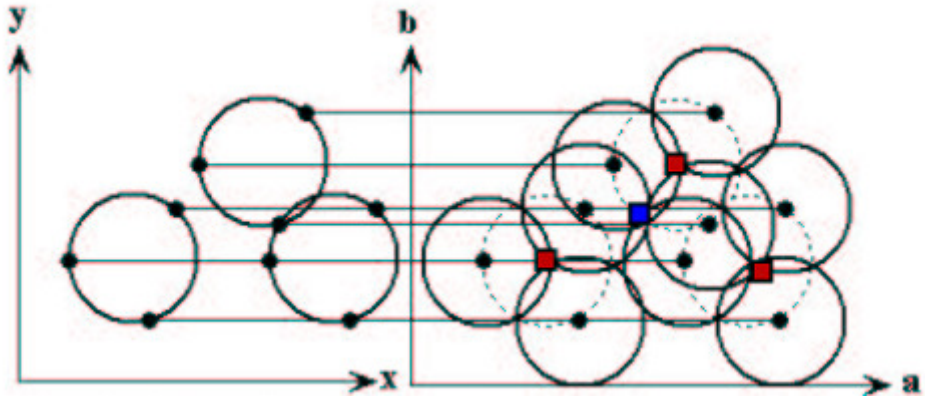
Houghova transformacija za kružnice (*eng. Circle Hough Transform – CHT*) je modificirana verzija Houghove transformacije. Houghova transformacija za kružnice se koristi za transformaciju skupa obilježja iz prostora slike u skup akumuliranih glasova u prostoru parametara.

Jednadžba kružnice:

$$(x-a)^2 + (y-b)^2 = r^2$$

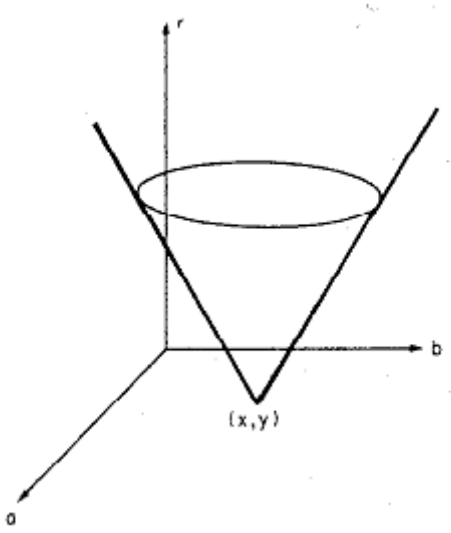
(x,y) predstavlja koordinate piksela na slici, (a,b) koordinate središta kružnice, te r polumjer kružnice.

Svaka točka (x,y) se u parametarskom prostoru transformira u skup kružnica sa središtem u koordinatama (x,y). Recimo da poznajemo polumjer kružnice, svaka točka te početne kružnice tada se transformira u novu kružnicu. Kada se ovaj postupak provede nad svim točkama kružnice dobiva se skup kružnica istog polumjera koje se sijeku u jednoj točki čije koordinate određuju koordinate središta početne kružnice (Slika 5).



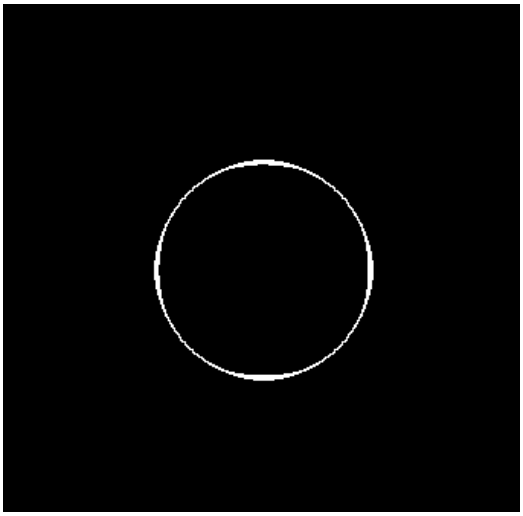
Slika 5 – Prikaz transformacije nekoliko točaka više kružnica [6]

Ako imamo sliku koja predstavlja rubove početne slike , moguće je provesti postupak akumulacije. Za svaki piksel koji predstavlja rub možemo se pitati: ako piksel leži na kružnici koja bi krivulja prikazala moguća mjesta središta kružnice. Ta krivulja je plašt stošca (Slika 6) .

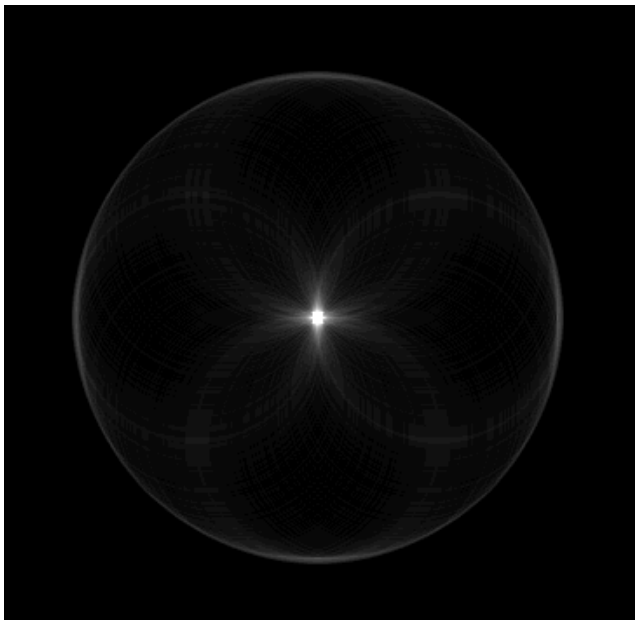


Slika 6 – Prikaz parametarskog prostora [1]

Važan dio prepoznavanja Houghovom transformacijom je prepoznavanje maksimuma u akumulacijskom polju. Prepoznavanjem maksimuma potrebno je odrediti točnu lokaciju središta kružnice, te točan broj kružnica.



Slika 7 – Kružnica



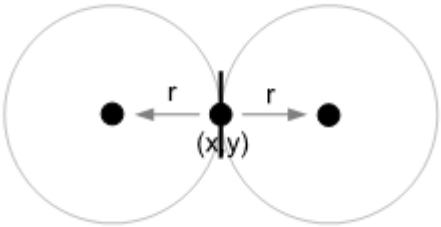
Slika 8 – Prikaz akumulatorskog polja

Slika 7 prikazuje kružnicu polumjera r0 čiji pikseli glasuju te na temelju koje se dobiva Slika 8. Slika 8 prikazuje vrijednosti akumulatorskog polja nijansama sive boje, pri čemu vrijedi pravilo: što svjetlija boja - veća vrijednost akumulacijskog polja. Dobiveno akumulacijsko polje je za r=r0.

2.3. Varijacije Houghove transformacije za kružnice

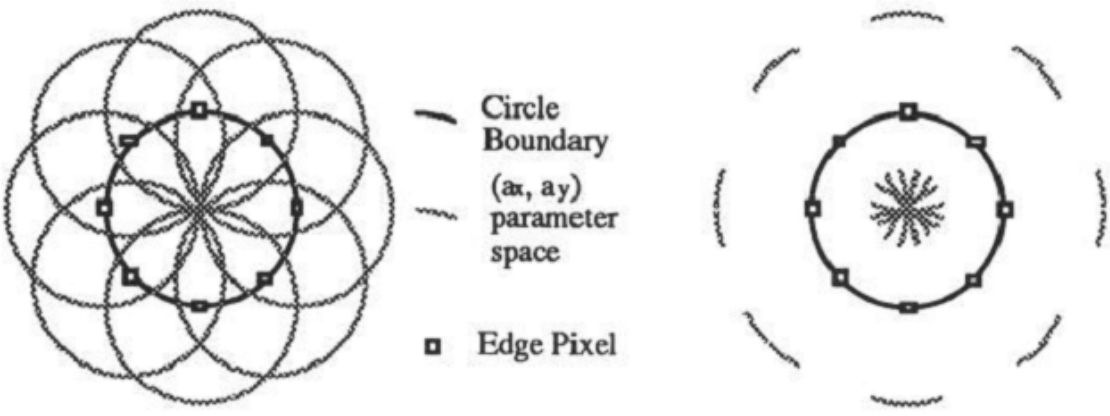
2.3.1. Orijentacija rubova

Možemo primijetiti da ne postoji potreba da glasovi piksela budu cjelovite kružnice (ako poznajemo r), tj. stožac (ako ne poznajemo r). Ako imamo informaciju o smjeru ruba dovoljno je da piksel glasa u smjeru normale na rub. U idealnoj situaciji središte kružnice se nalazi na liniji koju određuje normala na rub. Time parametarski prostor plašta stošca postaje dvostruka linija.

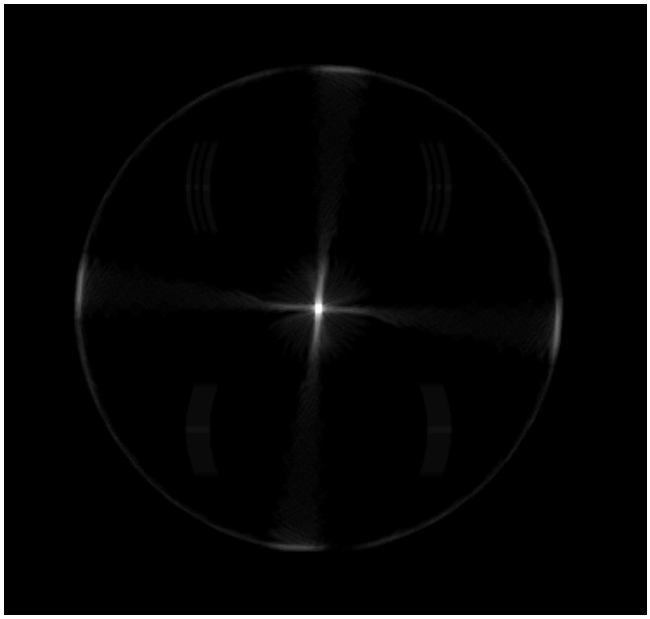


Slika 9 [3]

U realnim primjerima informacija o smjeru normale često nije potpuno točna, tada u obzir uzimamo i neki otklon θ. Svaki piksel glasa u smjeru gradijenta $\pm\theta$. Ova transformacija naziva se standardna Houghova transformacija (*eng. Standard Hough transform*) [2]. Parametarski prostor ove transformacije je dio plašta stošca.



Slika 10 – Usporedba Houghove transformacije sa i bez korištenja informacije o orijentaciji ruba [4]



Slika 11 – Akumulacijsko polje Slika 7 uz korištenje informacije o orijentaciji ruba, te $\theta = \frac{\pi}{8}$

2.3.2. Gerig i Klein Hougova transformacija

Za nepoznati polumjer akumulacijsko polje je 3D polje. Metoda Gerig i Klein uklanja potrebu za trodimenzijskim poljem. Umjesto jednog 3D polja koristi se tri dvodimenzijska polja. Dva dvodimenzijska polja se koriste za koordinate središta kružnice, te jedno polje za polumjer kružnice. [2]

Struktura ovog algoritma je vrlo jednostavna. Kako parametarski prostor predstavlja plašt stošca , za zadani r=r0 , parametarski prostor postaje kružnica polumjera r0. Time postaje nepotrebno akumulirati vrijednost za svaki polumjer. Ova metoda se može nadopuniti informacijom o smjeru ruba.

2.3.3. Brza Houghova transformacija

Brza Houghova transformacija (*eng. Fast Hough Transform* – FHT) je metoda koja za akumulaciju koristi višedimenzionalnog stabla čiji čvorovi imaju maksimalno četvero djece (*eng. Quadtree*). [2]

FHT se zasniva na korištenju stvaranja hiperravnine (*eng. Hyperplane*). Glasovi u parametarskom prostoru tvore hiperravninu. Ovaj pristup nije prikladan za pronalaženje kružnica zbog problema nelinearnosti među parametrima u

formulaciji hiperravnini. Zbog izbjegavanja ovog problema i povećanja učinkovitosti koristi se druga formulacija koja upotrebljava informaciju o orijentaciji ruba. Ova transformacija se naziva promijenjena brza Houghova transformacija (*eng. modified Fast Hough Transform - MFHT*). [2]

MFHT metoda koristi činjenicu da je parametarski prostor sa informacijom sa informacijom o normali ruba; dvije linije u 3D prostoru koje prolaze kroz točke (xi, yi). Za određivanje da li je jedna od tih dvaju linija sječe sa hiperkockom uspoređuje se vertikalna udaljenost od središta hiperkocke do linija sa duljinom dijagonale kocke. Ako je dijagonala manja, hiperkocka dobiva jedan glas.

3. Pretprocesiranje slike

Houghova transformacija se provodi nad pretprocesiranim slikama. Korišten je Cannyjev detektor rubova i HSL filter.

3.1. Cannyev detektor rubova

Rubovi su definirani kao nagle promjene u intenzitetu susjednih piksela i predstavljaju granice objekata na slici. Cannyev detektor rubova je jedan od najkorištenijih i najtočnijih detektora rubova. Upravo zbog njegove točnosti korišten je za pretprocesiranje slika za Houghovu transformaciju.

Cannyeva detekcija rubova sastoji se od par koraka: prigušenje šuma, računanje gradijent inteziteta, stanjivanje rubova, te usporedba sa pragom. Ulaz u Cannyev detektor rubova je crno-bijela slika.

1. Prigušenje šuma – Ovaj korak smanjuje detaljnost slike, zamagljuje ju. Prigušenje šuma postiže se Gaussovim filtrom. Veličina Gaussovog filtra je jedan od parametara koji se predaje Cannyjevom detektoru rubova.
2. Računanje gradijent intenziteta – Gradijent je vektor koji pokazuje smjer najveće promjene inteziteta. Računa se vodoravna i okomita komponenta gradijenata , te se na temelju formule (2) dobiva ukupni intenzitet.

$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2} \tag{2}$$

Pomoću formule (3) računa se smjer gradijenta.

$$\mathbf{\Theta} = \arctan\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right) \tag{3}$$

Izračunati kut se zaokružuje na četiri vrijednosti: 0°, 45°, 90° i 135°.

3. Stanjivanje rubova – Rubovi se stanjuju do veličine jednog piksela.
4. Usporedba s pragom – Cannyjev detektor koristi dva praga: gornji i donji prag (*eng. low threshold, high threshold*). Pikseli intenziteta većeg od gornjeg praga prihvaćaju se kao rubovi, pikseli vrijednosti nižeg od donjeg praga se odbacuju, a pikseli između se dodatno razmatraju: uspoređuju se sa gornjim pragom vrijednosti susjednih 8 piksela.

Slika 12 – Početna slika

Slika 13 – Crno – bijela slika, ulazna slika Cannyjevog detektora rubova

Slika 14 – Prepoznati rubovi

Slika 15 – Usporedba s pragom

3.2. HSL filter

Ovaj filter djeluje u HSL prostoru boja. Čuva piksele čija je boja u ili izvan zadanog opsega boje. Ovaj filter prevodi sliku iz RGB prostora u HSL pomoću formula (4), (5), (6) gdje r, g, b predstavljaju boju u RGB prostoru u intervalu [0,1], max maksimalnu od tih vrijednosti, te min minimalnu.

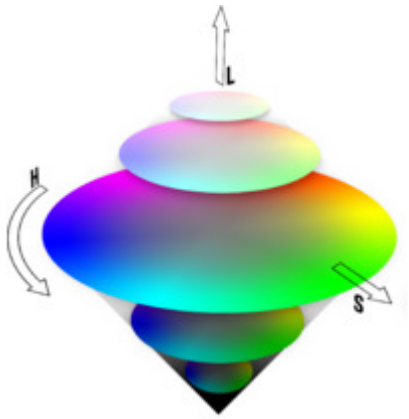
Svaka boja predstavljena je sa tri komponente:

- Nijansa (*eng. hue*)
- Zasićenje (*eng. saturation*)
- Osvjetljenje (*eng. lightness*)

$$h = \begin{cases} 0 & ako\ je\ max = min \\ \left(60° \times \dfrac{g-b}{max-min} + 360°\right) mod\ 360°, & ako\ je\ max = r \\ 60° \times \dfrac{b-r}{max-min} + 120°, & ako\ je\ max = g \\ 60° \times \dfrac{r-g}{max-min} + 240°, & ako\ je\ max = b \end{cases} \tag{4}$$

$$l = \frac{1}{2}(max + min) \tag{5}$$

$$s = \begin{cases} 0 & ako\ je\ max = min \\ \dfrac{max-min}{max+min} = \dfrac{max-min}{2l}, & ako\ je\ l \leq \dfrac{1}{2} \\ \dfrac{max-min}{2-(max+min)} = \dfrac{max-min}{2-2l}, & ako\ je\ l > \dfrac{1}{2} \end{cases} \tag{6}$$



Slika 16 – Prikaz HSL prostora

Slika 17 – HSL filter nad Slikom 14

4. Programsko ostvarenje

Algoritam Houghove transformacije je implementiran u programskom jeziku C#. Za pretprocesiranje slike korištena je biblioteka Aforge.Net. Uz sam algoritam izrađeno je i korisničko sučelje koje pokazuje sve faze obrade slike, te omogućava promjenu važnijih parametara.

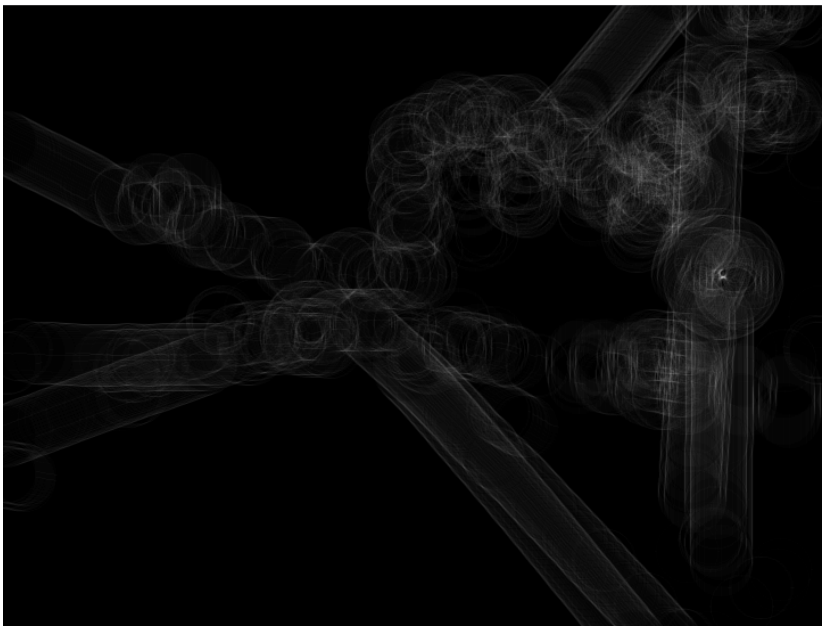
4.1. Houghova transformacija

Algoritam transformacije:

```
 1: za svaki piksel slike
 2:   ako je piksel postavljen onda
 3:     akumuliraj_polje_oko_piksela(piksel);
 4:
 5: akumuliraj_polje_oko_piksela(piksel):
 6:   delta_x=0;
 7:   delta_y=0;
 8:   //kut između dva susjedna piksela kružnice
 9:   delta_theta=1/polumjer;
10:   i=0;
11:   dok je i<(pi/2):
12:     delta_x=r*sin(i);
13:     delta_y=r*cos(i);
14:     akumuliraj_polje(x+delta_x, y+delta_y);
15:     akumuliraj_polje(x+delta_x, y-delta_y);
16:     akumuliraj_polje(x-delta_x, y+delta_y);
17:     akumuliraj_polje(x-delta_x, y-delta_y);
18:     i+=delta_theta;
```

Ovaj algoritam se ponavlja za svaki promatrani polumjer. Algoritam stvara akumulacijsko polje. Za ulaznu Sliku 12, te r=28, akumulacijsko polje prikazuje Slika 18.

Algoritam se neznatno mijenja ako se koristi informacija o orijentaciji ruba. Razlika je u 10. i 11. retku, varijabla i se postavlja na vrijednost kuta orijentacije ruba, te se petlja iterira ne iterira do pi/2, nego do kuta theta, čija je vrijednost prvotno postavljena na pi/8.



Slika 18 – Akumulacijsko polje Slike 12 za r=28

4.2. Izdvajanje maksimuma akumulacijskog polja

Nakon provedene transformacije potrebno je izdvojiti maksimume iz akumulacijskog polja. Elementi akumulacijskog polja oko pravog središta kružnice također imaju visoku vrijednost. Cijelo područje visoke vrijednosti potrebno je tretirati kao jednu cjelinu.

Implementirani algoritam pronalazi jedan element koji je veći od zadanog praga i sve elemente koji su povezani s njim, te pronalazi središte cijelog područja.

Zadani prag je iz intervala [0,1]. Predstavlja dio (postotak) kružnice koji treba glasati za središte kružnice. Npr. prag od 0.5 znači da se u akumulacijskom polju traže vrijednosti veće od 0.5*2*r*π, tj. elemente za koje je glasalo više od pola kružnice.

Algoritam pronalaska:

```
1: pronadi_kruznice():
2:  za sve elemente (i,j) akumulacijskog polja
3:    ako je vrijednost elementa veća od praga onda:
4:      //globalne varijable
5:      xmin=0;
6:      xmax=širina_slike;
7:      ymin=0;
8:      ymax=duljina_slike;
```

```
 9:      nadi_podrucje(i,j);
10:      dodaj_kruznicu((xmax+xmin)/2,
12:                     (ymax+ymin)/2,polumjer));
13:
14: nadi_podrucje(x,y):
15:   ako x ili y izlaze izvan okvira slike onda: vrati se;
16:  ako je akumulacijsko_polje[x,y]<prag onda: vrati se;
17:  akumulacijsko_polje[x,y]=0;
18:  ako je x < xmin onda: xmin=x;
19:  ako je x > xmax onda: xmax=x;
20:  ako je y < ymin onda: ymin=y;
21:  ako je y > ymax onda: ymax=y;
22:  nadi_podrucje(x,y+1);
23:  nadi_podrucje(x,y-1);
24:  nadi_podrucje(x+1,y);
25:  nadi_podrucje(x-1,y);
```

4.3. Korištenje programa

Primjer korištenja klase houghTransform:

```
 1: houghTransform hough = new houghTransform(image);
 2: hough.threshold = 0.4d;
 3: hough.Trans(rmin, rmax, rstep, 0);
 4:
 5: Image = hough.HoughMapToImage();
 6:
 7: foreach (Circle circle in hough.houghCircle)
 8:  {
 9:    Rectangle rectangle = new Rectangle(circle.x -
10:          circle.r, circle.y - circle.r, circle.r * 2,
11:          circle.r * 2);
12:    graphics.DrawRectangle(pen, rectangle);
13:  }
14:
15: houghEdgeHSL = new houghTransform(edgeImage,redImage);
```

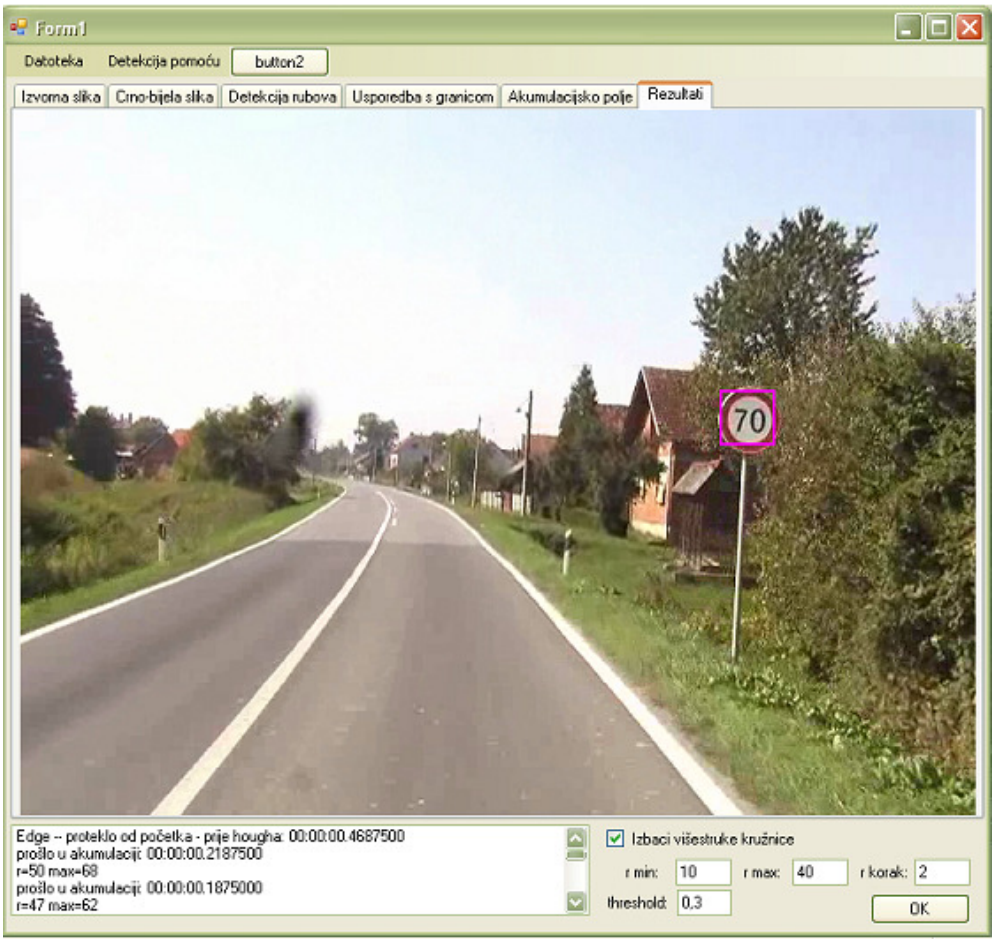
Redak 1 je konstruktor kojemu se predaje pretprocesirana slika formata 24 bitnog RGB ili 8 bitnog indeksiranog. Redak 2 postavlja prag prepoznavanja, dok redak 3 poziva transformaciju za određeni broj polumjera, četvrti parametar je minimalna veličina područja u akumulacijskom polju većeg od praga (korisno za HSL filtriranu sliku, zbog kružnice debljeg obruba i područje maksimuma je veće veličine).

Redak 7 poziva pretvorbu akumulacijskog polja u sliku.

Reci 7-13 pokazuju iscrtavanje pronađenih kružnica.

Redak 15 pokazuje konstruktor u slučaju korištenja kombinacije slike rubova i HSL filtrirane slike.

Jednostavno korištenje algoritma moguće je uz korisničko grafičko sučelje. Sučelje prikazuje sve faze obrade slike, te nudi jednostavnu promjenu metode detekcije i parametara Houghove transformacije. Nakon provedene transformacije prikazuju se korisne informacije o vremenskoj potrošnji i maksimumima akumulacijskog polja.



Slika 19 – Korisničko sučelje

5. Rezultati

Postoji mnogo kriterija po kojima se algoritmi za prepoznavanje znakova, tj. algoritmi općenito, mogu ocijeniti. Neki od njih su: točnost, robusnost, vremenska, te prostorna složenost.

Točnost se mjeri tako što se uspoređuje centar i polumjer pronađene kružnice sa pravim centrom i polumjerom znaka uz neko dozvoljeno odstupanje.

Algoritme je potrebno testirati na reprezentativnom skupu slika. Skup slika treba pokazati prednosti i mane pojedine varijacije algoritma. Skup slika nad kojima je testiran program sadrži slike koje nemaju nijedan, ili pak imaju jedan ili više okruglih znakova, znakove slabije vidljivosti; tj. oštrine i osvjetljenja, objekte slične boje i oblika okruglim znakovima.

Zbog većeg broja filtera pomoću kojih se pretprocesira slika, bilo je potrebno naći optimalne parametara za pojedine filtre, te optimalne parametre za samu Houghovu transformaciju.

Mjereno je više elemenata brzine algoritma; ukupno vrijeme, vrijeme potrebno za akumulaciju jednog polumjera, te ukupno vrijeme kada se razmatra samo jedan polumjer.

Moguće preinake za povećanje brzine:

- Smanjivanje slike
- Smanjivanje broja polumjera
- Prilagođavanje filtera, povećanje praga detektora rubova ili smanjenje opsega boje HSL filtera, rezultat je smanjenje broja piksela koji će glasati
- Korištenje podatka o orijentaciji

Ako je potrebno prepoznati znakove u videu, te je potrebno svaki okrugli znak prepoznati bar jednom, preporučljivo je smanjiti raspon promatranog polumjera, čak i na samo jedan polumjer. U videu znakovi mijenjaju veličinu, te pravilnim odabirom jednog polumjera, svi znakovi bi trebali biti prepoznati, a algoritam dobiva višestruko ubrzanje.

5.1. Klasična sa Canny detektorom rubova

Klasična transformacija je prikladna za slike koje su oštre. Granica Canny detektora ruba se tada može postaviti na veću vrijednost. Time se dobiva manje piksela na ulaznoj slici za Hougovu transformaciju, i samim time veću brzinu transformacije. Zbog veće granice rubovi znaka na lošijim slikama se izgube. Smanjivanjem granice dolazi do velikog broja lažnih detekcija. U ovim slučajevima prikladnija je neka druga transformacija.

Točno prepoznatih znakova je 71%, na 8% slika detektirani su znakovi koji to nisu (*eng. false positive*).

Parametri za koje je izračunata točnost:

Canny:
LowThreshold=75
HighThreshold=80
GaussianSigma=9

Houghova transformacija:
rmin=15, rmax=50, rstep=3
Threshold=0.4

Brzina za sliku veličine 720x576 piksela:
Ukupno=2s
Pretprocesiranje slike=0.4s
Akumulacija za jedan polumjer = 0.15s
Ukupno za jedan polumjer = 0.6s

5.2. Korištenje informacije o orijentaciji i Canny detektor rubova

Prednost korištenja podatka o orijentaciji ruba je brže izvođenje algoritma, te točnije određivanje središta. Zbog toga što ta informacija nije uvijek točna, manji broj piksela glasa za pravo središte kružnice, dio glasova koje bi pravo središte dobilo se izgubi. Zbog tog razloga potrebno je smanjiti prag za pronalaženje kružnica. Nakon što se smanji prag algoritam je osjetljivi, te pronalazi kružnice tamo gdje ih ne bi pronašao sa običnom transformacijom i većim pragom.

Pokazalo se da je tada posebno osjetljiv na područja u kojem se nalazi više paralelnih linija.

Točno prepoznati znakovi: 58%

Krive detekcije: 7% slika

Parametri za koje je izračunata točnost:

Canny:
LowThreshold=75
HighThreshold=80
GaussianSigma=9

Houghova transformacija:
rmin=15, rmax=50, rstep=3
Threshold=0.2

Brzina za sliku veličine 720x576 piksela:
Ukupno=1s
Pretprocesiranje slike=0.4s
Akumulacija za jedan polumjer = 0.04s
Ukupno za jedan polumjer = 0.5s

5.3. Pomoću filtera boje

Inačica Houghove transformacije koja koristi kao ulaz sliku dobivenu HSL filterom pogodna je za slike u kojima se vidi boja, te ne postoje drugi objekti iste boje. Ova inačica implementirana je samo za znakove sa crvenim obrubom, tj. znakove izričitih naredbi.

Broj ispravnih detekcija moguće je povećati proširenjem boje koja se filtrira, ali tim postupkom dolazi do velikog broja krivih detekcija.

Točno prepoznati znakovi: 54%

Krive detekcije: 34% slika

Parametri za koje je izračunata točnost:

HSL filter:
Hue = (335, 10);

Saturation = (0.09, 1);
Luminance = (0.15, 0.833);

Houghova transformacija:
rmin=15, rmax=50, rstep=3
min=5
Threshold=0.5

Brzina za sliku veličine 720x576 piksela:
Ukupno=1s
Pretprocesiranje slike=0.1s
Akumulacija za jedan polumjer = 0.07s
Ukupno za jedan polumjer = 0.2s

5.4. Pomoću filtera boje i detektora rubova

Ova varijanta koristi kao ulaz dvije slike, jednu na kojoj je filtrirana boja, te drugu koja predstavlja rubove. Razmatraju se pikseli čija je vrijednost na obje slike veća od nule.

U odnosu na prošle verzije smanjen je prag Cannyevog detektora rubova, te je povećan opseg boje koja se filtrira.

Pretprocesiranje slike u ovoj inačici traje najduže, ali zbog malog broja piksela za koje je potrebno vršiti akumulaciju, ukupno vrijeme je najmanje. Brzina je pogodna ako se razmatra veći broj polumjera. Ova varijanta se pokazala kao najtočnija i najbrža za veći broj razmatranih polumjera.

Točno prepoznati znakovi: 81%

Krive detekcije: 23% slika

Parametri za koje je izračunata točnost:

HSL filter:
Hue = (250, 30);
Saturation = (0.09, 1);
Luminance = (0.2, 0.8);

Canny:

LowThreshold=15

HighThreshold=20

GaussianSigma=9

Houghova transformacija:

rmin=15, rmax=50, rstep=3

Threshold=0.3

Brzina za sliku veličine 720x576 piksela:

Ukupno=0.9s

Pretprocesiranje slike=0.5s

Akumulacija za jedan polumjer = 0.03s

Ukupno za jedan polumjer = 0.6s

Uz povećani opseg filtrirane boje dobiva se veći postotak prepoznatih znakova, ali i dosta veći broj neispravnih detekcija.

HSL filter:

Hue = (200, 30);

Saturation = (0.02, 1);

Luminance = (0.1, 0.8);

Točno prepoznati znakovi: 93%

Krive detekcije: 75% slika

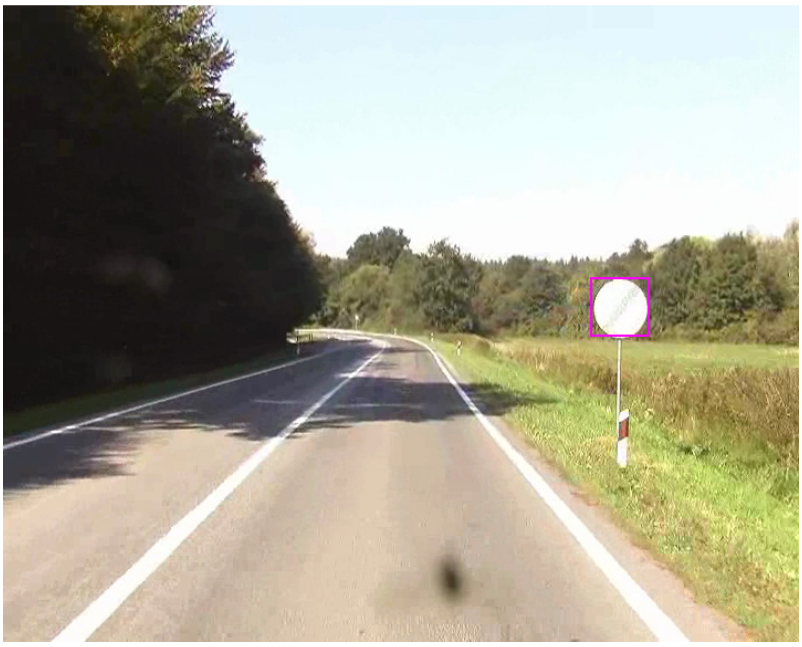
Tablica 1 - Usporedba dobivenih rezultata

	Canny detektor	Canny-orijentacija	HSL filter	HSL filter i Canny
Točnost	71%	58%	54%	81%
Kriva detekcija	8%	7%	34%	23%
Vrijeme obrade	2s	1s	1s	0.9s
Vrijeme obrade za jedan polumjer	0.6s	0.5s	0.2s	0.6s

5.5. Primjeri uspješne detekcije

Slika 20 - Uspješno detektiran znak

Slika 21 - Uspješno detektiran znak



Slika 22 - Uspješno detektiran znak

Slike 20, 21 i 22 prikazuju ispravnu detekciju znakova. Sliku 22 ispravno prepoznaje inačica sa Canny detektorom rubova i inačica koja koristi informaciju o orijentaciji ruba. Korišteni su parametri iz poglavlja 5.1 – 5.4 .

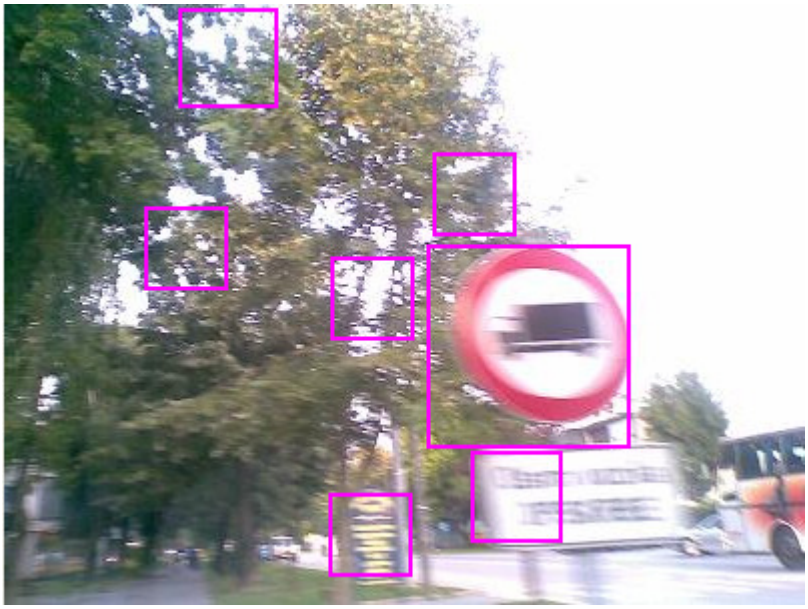
Slika 23 – Uspješna detekcija pomoću kombinacije HSL filtera i Canny detektora rubova

Slika 23 prikazuje ispravnu detekciju zamućenog znaka pomoću kombinacije filtera. Također i sam HSL filter ispravno detektira ovaj znak. Problem kod Cannyjevog detektora rubova je što rubovi samog znak nisu dovoljno izražajni, te se nakon usporedbe sa pragom dio kružnice odbacuje. Prilagođavanjem parametara (rmin=20, rmax=50, rstep=2, threshold=0.3) dobiva se ispravna detekcija (slika 26). Zbog mnogo rubova od lišća, te smanjenog praga prepoznavanja dolazi do neispravnih detekcija. U ovom slučaju rješenje bi moglo biti povećanje minimalnog razmatranog polumjera.

Slika 24 - Rubovi Slike 23

Slika 25 - Rubovi nakon usporedbe

sa pragom Slike 23



Slika 26 - Detekcija pomoću Canny detektora rubova

5.6. Primjeri neuspješne detekcije

Primjeri krive detekcije:

Slika 27 - Neispravno detektiran znak

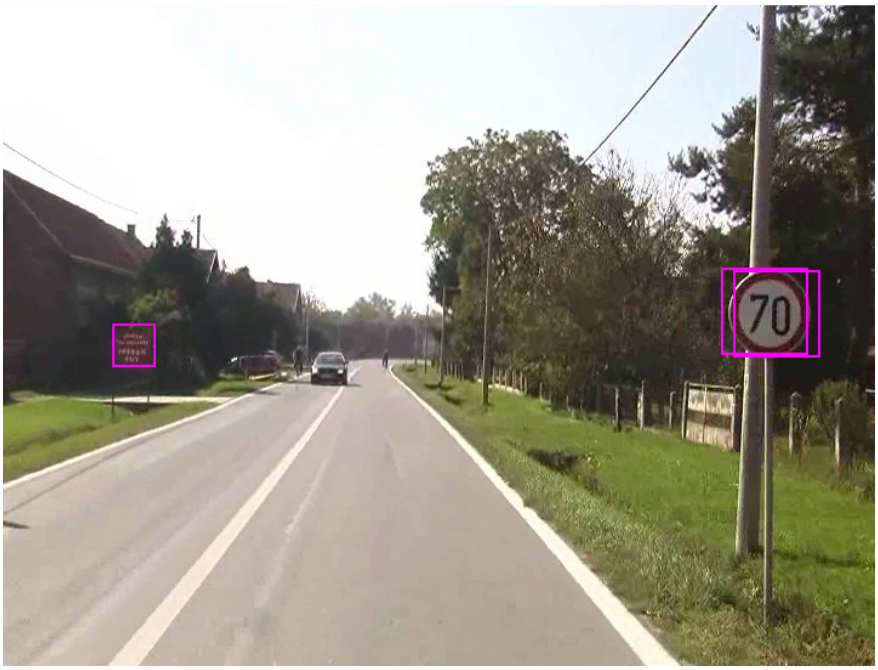
Slika 27 je primjer krive detekcije drugog objekta polukružnog oblika.

Slika 28 prikazuje krivu detekciju pomoću Cannyevog detektora rubova. Nepostojeći znakovi se detektiraju u području gdje postoji mnoštvo izražajnih

rubova. Povećanjem minimalnog razmatranog polumjera ili praga Houghove transformacije neispravne detekcije nestaju.

Slika 28 - Neispravno detektirani znakovi

Slika 29 - Rubovi Slike 27



Slika 30 – Neispravna i ispravna detekcija pomoću HSL filtera

Slika 30 prikazuje krivu detekciju pomoću HSL filtera. Do neispravne detekcije dolazi zbog drugog objekta crvene boje. Kombinacija Cannyevog detektora rubova i HSL filtera rješava ovaj problem.

Primjeri propuštene detekcije:

Do propuštene detekcije dolazi zbog velike udaljenosti znaka, stapanja znaka sa pozadinom, ili nemogućnosti prepoznavanje boje znaka.

Slika 31 - Propuštena detekcija pomoću HSL filtera

Slika 32 - Ispravna detekcija Slike 29 pomoću Canny detektora rubova

Slika 31 prikazuje znak čija je boja pretamna. Zbog toga HSL filter, te kombinacija HSL filtera i Cannyjevog detektora ne prepoznaje znak. Cannyjev detektor rubova, sa i bez informacije o rubu ispravno prepoznaje znak. HSL filter je potrebno dodatno podesiti da bi točno prepoznao znak, potrebno je povećati opseg boje koja se prepoznaje.

Slika 33 - Propuštena detekcija

6. Zaključak

Houghova transformacija je jednostavna metoda za prepoznavanje kružnica. Lako se implementira, te ima zadovoljavajuće rezultate. Metoda je otporna na smetnje. Prednost joj je i to što nije potrebno prethodno učenje. Sama transformacija za manji opseg razmatranih polumjera je relativno brza.

U radu je testirano više verzija Houghove transformacije. Svaka od testiranih varijacija ima svoje prednosti i mane. Poznavajući kvalitetu slika, te područje izabire se najbolja varijacija. Ako skup slika nad kojima se koristi program je snimljena u isto vrijeme, npr. video, za bolje performanse detekcije moguće je prilagoditi parametre filtera i Houghove transformacije. Najtočnijom verzijom transformacije, tj. pretprocesiranja slike pokazala kombinacija prepoznavanja rubova i HSL filtera, točnosti 81%. Ova verzija je i najbrža. Inačica koja koristi Cannyev detektor rubova je druga po točnosti, 71%, njezina prednost nad kombinacijom prepoznavanja rubova i HSL filtera je manja detekcija neznakova (*eng. false positive*).

Hougova transformacija pokazala se kao solidan algoritam za prepoznavanje okruglih znakova. Metoda nije savršena, te ima prostora za poboljšanje. Preporučljivo ju je koristiti kao dopunu nekoj drugoj metodi.

7. Literatura

[1] Ballard,D.H. : *„Generalizing the Hough transform to detect arbitrary shapes*“, pp. 111-122, 1980.

[2] Yuen,H.K. ,Princen,J. , Illingworth,J. , Kittler, J. : „*A comparative study of Hough transform methods for circle finding“*, In Proc. AVC, pp. 169-174., 1989.

[3] Smereka,M. , Duleba,I. : „*Circular object detection using a modified Hough transform*“, Int. J. Appl. Math. Comput. Sci., Vol. 18, No. 1, pp. 85–91, 2008.

[4] Atherton,T.J. , Kerbyson,D.J. : *„The Coherent Circle Hough Transform“*

[5] Duda,R.O., Hart,P.E. : „*Use of the Hough Transformation to detect lines and curves in pictures*“, 1971.

[6] Rhody,H. : „*Hough Circle Transform*“, prezentacija , Chester F. Carlson Center for Imaging Science, 2005.

[7] Bradski, G. , Kaehler,A. :„*Learning OpenCV*“, O'Reilly Media,SAD,rujan 2008.

[8] „*Hough Transform*“, s Interneta, http://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm, 1.6.2009.

[9] „*Hough Transform*“, s Interneta, http://en.wikipedia.org/wiki/Hough_transform, 1.6.2009.

[10] Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

[11] Poynton, C. , “*What are HSB and HLS?” Color FAQ*, s Interneta, http://www.poynton.com/notes/colour_and_gamma/ColorFAQ.html#RTFToC36 , 10.6.2009.

[12] „HSL color space“, s Interneta, http://en.wikipedia.org/wiki/HSL_color_space, 10.6.2009.

Detekcija prometnih znakova primjenom Houghove transformacije

Sažetak

Ovaj rad razmatra prepoznavanje okruglih prometnih znakova Houghovom transformacijom. Pretprocesiranje slika i Hougova transformacija je izvršena na više različitih načina; Canny detektorom rubova sa i bez informacije o orijentaciji ruba, ulaznom slikom filtriranom HSL filterom, te kombinacijom Canny detektora rubova i HSL filtera. Algoritam je implementiran u programskom jeziku C#, te je korištena biblioteka Aforge.Net. U radu je prikazana usporedba inačica, te su komentirane prednosti i mane pojedine inačice. Uspješnost algoritama je prikazana i komentirana.

Ključne riječi

Računalni vid, prepoznavanje prometnih znakova, Houghova transformacija, Cannyjev detektor rubova, HSL

Traffic sign detection using Hough tranform

Abstract

This work considers traffic sign detection using Hough trasformatin. Image preprocessing and Hough transfom is performed in several different ways; with Canny edge detecting with and without edge orientation information, with image filtered by HSL filter and combination of Canny edge detector and HSL filter. Alogrithm has been implemented in programming language C#. Aforge.Net library has been used. This work shows comparison of implemented version. Advantages and disadvantages of each version has been commented. The efficiency of algorithms has been presented and commented.

Key words

Computer vision, traffic sign detection, Hough transform, Canny edge detector, HSL