SVEUČILIŠTE U ZAGREBU
**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

DIPLOMSKI RAD br. 318

# Detekcija obojenih pravokutnih prometnih znakova

Šime Bašić

Zagreb, lipanj 2011.

# Sadržaj


Uvod ... 1

1. Ostali pristupi u detekciji prometnih znakova ... 2

2. Korišteni pristup u detekciji prometnih znakova ... 5

2.1. Segmentacija upotrebom histograma vjerojatnosti boje i Bayesovog teorema ... 5

2.2. Cannyjev detektor rubova ... 9

2.3. Gradijent iz slike u boji ... 10

2.4. Pronalaženje lanaca rubnih piksela ... 11

2.5. Pronalaženje pravocrtnih segmenata ... 12

2.6. Ulančavanje pravocrtnih segmenata ... 13

3. Programska implementacija ... 16

3.1. Implementacija detekcije na temelju boje ... 17

3.2. Implementacija detekcije na temelju rubova ... 18

3.3. Implementacija učenja histograma vjerojatnosti boja ... 24

3.4. Rad s razvojnim okruženjem i ljuskom cvsh ... 25

4. Eksperimentalni rezultati ... 29

4.1. Detekcija žutih prometnih znakova ... 33

4.1.1. Pozitivni rezultati ... 37

4.1.2. Lažno pozitivni rezultati ... 38

4.1.3. Lažno negativni rezultati ... 41

4.2. Detekcija plavih prometnih znakova ... 43

4.2.1. Pozitivni rezultati ... 45

4.2.2. Lažno pozitivni rezultati ... 46

4.2.3. Lažno negativni rezultati ... 48

4.3. Detekcija crvenih prometnih znakova ... 50

4.3.1. Pozitivni rezultati ... 51

4.3.2. Lažno pozitivni rezultati ... 52

4.3.3. Lažno negativni rezultati ... 53

Zaključak ... 54

Literatura ... 55

Sažetak ... 56

# Uvod

Razvoj tehnologije munjevito mijenja mogućnosti modernog vozila. Uvođenje sustava baziranih na senzorima u vozila omogućuje vozaču pregled važnih informacija iz okoline vozila. Takvi sustavi imaju veliki potencijal u povećanju razine sigurnosti, udobnosti i učinkovitosti same vožnje pa je razvoj sigurnijih vozila jedan od ciljeva državnih vlada, proizvođača auta i elektroničke opreme te akademske zajednice. Identifikacija prometnih znakova jedno je od najvećih područja izučavanja inteligentnih transportnih sustava. Ljudska vizualna percepcija ovisi o osobnim fizičkim i mentalnim sposobnostima koje ponekad mogu biti pod utjecajem umora i napetosti. Zbog toga je važno imati sustav za identifikaciju znakova kako bi se pravovremenim informiranjem vozača spriječile prometne nesreće, kako bi se vozača podsjetilo na trenutno ograničenje brzine, odnosno da bi se vozača spriječilo u poduzimanju neprihvatljivih akcija. Automatsko prepoznavanje znakova može služiti u održavanju prometne infrastrukture ili u autonomnoj vožnji bez ljudskog upravljanja. Identifikacija prometnih znakova se sastoji od dvije glavne faze, detekcije i prepoznavanja. U fazi detekcije slika se predprocesira i segmentira ovisno o značajkama znakova kao što su boja ili oblik. Učinkovitost i brzina faze detekcije su važni faktori u cijelom procesu jer smanjuju područje pretraživanja slike i upućuju samo na potencijalne regije slike gdje bi se znakovi mogli nalaziti. U fazi prepoznavanja svaka potencijalna regija se uspoređuje sa skupom značajki da bi se ustanovilo da li regija sadrži znak ili ne. U slučaju pozitivnog odgovora znak se ovisno o dobivenim značajkama svrstava u odgovarajući tip znaka. Detekcija je važniji i zahtjevniji dio jer prometni znakovi mogu biti identificirani samo ukoliko su potencijalne regije slike ispravno i precizno segmentirane. Uspješna detekcija može biti narušena kutom kamere gdje se događa da okrugli znakovi postanu elipse, boja i oblik znaka degenerira uslijed prometnih nesreća i vremenskih neprilika, a složene pozadine dodatno otežavaju uspješnu detekciju. Znakovi su dizajnirani u izraženim bojama stvarajući kontrast s pozadinom kako bi ih vozač lako uočio, ali su te boje osjetljive na varijacije osvjetljenja u sceni. ¸

Cilj ovog rada je predstaviti sustav za detekciju prometnih znakova na temelju boje i oblika znaka. U 1. poglavlju su opisani neki drugi pristupi u detekciji znakova. U 2. poglavlju je dan opis korištenih postupaka u ovom radu. Programska implementacija je dana u 3. poglavlju, dok su eksperimentalni rezultati opisani u 4. poglavlju. Zadnje poglavlje sadrži analizu prednosti i slabosti korištenih postupaka.

# 1. Ostali pristupi u detekciji prometnih znakova

Neki od ostalih pristupa u detekciji prometnih znakova iskorištavaju boju znaka kao diskriminantnu značajku slikovnih regija. Jedan od takvih postupaka je [1] gdje se koristi histogram boja u detekcijskom oknu. Histogram boja u [1] je dvodimenzionalna struktura koja ima onoliko elemenata koliko ima kombinacija vrijednosti komponenata boje u odabranom sustavu boja (npr. za RGB sustav to je $256^3$). U navedenom pristupu detekcijsko okno se pomiče po slici, zatim se na svakoj poziciji okna iz slike izračunava histogram boje te se dobiveni histogram uspoređuje s naučenim histogramom. Ta dva histograma se uspoređuju element po element i ukoliko se značajno razlikuju u broju elemenata manjem od nekog zadanog praga tada se trenutno područje okna u slici smatra potencijalnim znakom. Naučeni histogram je prosjek zbroja histograma boje svih slika iz skupa za učenje iz kojeg je dodatno izračunat i histogram standardne devijacije čiji su elementi korišteni kao dopuštena mjera odstupanja pri usporedbi histograma. Prednosti ovog pristupa su kod detekcije objekata čija je rotacija ili pozicija u sceni nepoznata, a mane su to što se ignorira oblik traženog objekta.

U [2] se koristi također jedan od uobičajenih pristupa u detekciji znakova na temelju boje. Postupak se temelji na transformaciji izvornog sustava boje RGB u neki od ostalih sustava poput HSI. Zatim se svaki piksel uspoređuje sa zadanim pragovima komponenata sustava boja čime se piksel klasificira kao znak ili ne. U [2] je izvorna slika iz RGB sustava transformirana u HSI sustav. Nakon toga su na svaki piksel primijenjeni unaprijed zadani pragovi H, S i I komponente za traženu boju. Ukoliko je svaka komponenta boje piksela odgovarala zadanim granicama tada je piksel klasificiran kao znak. Za spremanje piksela klasificiranih kao znak korištena je binarna slika gdje su pikseli znaka obilježeni bijelom bojom, a svi ostalo crnom. Zatim je za sve bijele piksele iz binarne slike proveden postupak narastanja gdje su svi susjedni bijeli pikseli grupirani u zasebne cjeline. I na kraju je za svaku od tih cjelina određen omeđujući pravokutnik. Postupak je pokazao neželjenu osjetljivost na neke boje što bi se moglo popraviti tako da se zadani pragovi za H, S i I komponente odrede na temelju skupa znakova ciljanog tipa. Također bi se postupak mogao popraviti ako bi se pronađene regije dodatno filtrirale analizom oblika traženih znakova.

Pristup u [3] se također temelji na transformaciji sustava boja, ali uz korištenje dodatnih tehnika kao što su horizontalna i vertikalna projekcija slike. Prvo se početni RGB sustav boja transformira u HSV sustav (HSV slika). Zatim se svaka od $256^3$ različitih kombinacija boje iz HSV sustava kvantizira u neku od osam diskretnih razina boje. Detekcija se provodi tako da se stvori binarna slika za traženu boju znaka gdje će bijelom bojom biti označeni pikseli čija boja na HSV slici nakon kvantizacije odgovara traženoj boji. Zatim se na toj binarnoj slici vrši horizontalna i vertikalna projekcija. Horizontalna projekcija se slično kao i vertikalna provodi tako da se u odgovarajući element polja spremi broj bijelih piksela s odgovarajućom *y* komponentom. Onda se iz projekcijskih polja stvara niz nula i jedinica gdje se jedinica pridružuje elementu polja čija je vrijednost unutar zadanih pragova. Svaki kontinuirani niz jedinica u nekom smjeru se proglašava mogućim znakom ukoliko je taj niz veći od zadanog praga. Onda se stvara kartezijev produkt svih kandidata prethodno dobivenih u vertikalnom i horizontalnom smjeru. Za konačan izbor pozicije znaka uzima se samo jedan kandidat čiji su vertikalni i horizontalni nizovi jedinica najveći među svim ostalim kandidatima.

Pristup u [4] koji se koristi za detekciju okruglih crvenih znakova bi možda mogao u modificiranoj verziji poslužiti za detekciju pravokutnih znakova. Primjenjuje se kombinirani pristup detekcije oblika i boje. Na početku se slika transformira iz RGB sustava u HSV sustav boja. Nakon toga se na HSV sliku primjenjuju pragovi za crvenu boju čime se dobiva binarna slika gdje su crvenom bojom označeni svi pikseli koji pripadaju crvenom rubu znaka, a crnom bojom svi ostali. Zatim se na na svaku grupu susjednih bijelih piksela primjenjuje ispitivanje kružnog oblika da se provjeri da li to područje segmentirano bojom uistinu predstavlja okrugli crveni znak. Ispitivanje kružnog oblika se provodi na način da se odredi jednadžba kružnice kroz tri točke odabirom tri bijela piksela iz odgovarajuće grupe piksela koji bi trebali pripadati rubu znaka. Nakon što je dobivena jednadžba kružnice potrebno je za svaki bijeli piksel iz odgovarajuće grupe provjeriti udaljenost do središta dobivene kružnice. Ukoliko je srednja kvadratna pogreška udaljenosti svih bijelih piksela iz grupe manja od zadanog praga greške tada promatrana regija slike odgovara prometnom znaku.

Drugačiji pristup opisan je u [5] gdje su za detekciju crvenih znakova korištene dvoslojne neuronske mreže uz kombiniranje značajki boje i oblika znaka. Implementirane su dvije neuronske mreže, jedna za detekciju znakova na temelju boje i jedna za detekciju na temelju rubova. Za svaki neuron u izlaznom sloju se provjerava da li je piksel kojeg on

predstavlja potencijalni centar prometnog znaka. Svi pikseli slike imaju svoj pripadajući neuron u ulaznom i izlaznom sloju mreže, a svaki neuron izlaznog sloja je povezan sa svim neuronima ulaznog sloja. Prva neuronska mreža na ulaznom sloju neurona prima boju svih piksela u RGB sustavu, transformira dobivene boje u HSI sustav te uzima samo *hue* komponente. Na izlazu tih neurona dobije se mjera sličnosti primljene *hue* komponente i poznate *hue* vrijednosti za crvenu boju. Na ulaz neurona izlaznog sloja dolazi mjera sličnosti pomnožena s težinom veze između odgovarajućih neurona. Težina veze preferira radijus traženog objekta unutar zadanih granica te eliminira sve neurone ulaznog sloja čiji su pikseli preblizu ili predaleko od piksela promatranog neurona izlaznog sloja. Izlaz neurona izlaznog sloja je zbroj umnožaka mjere sličnosti i težine veze svih neurona ulaznog sloja koji su s tim izlaznim neuronom povezani. Druga neuronska mreža koja detektira mogući centar znaka na temelju rubova prima na ulazu iznos gradijenta za sve piksele slike. Izlaz tih neurona se množi s težinom veze između dvaju slojeva. Veza kažnjava sve neurone čiji su pikseli predaleko od piksela odgovarajućeg izlaznog neurona, a potpuno eliminira neurone čiji su pikseli bliže od zadanog praga. Oba izlaza dviju mreža se kombiniraju kako bi se dobila konačna mjera da li je odgovarajući piksel mogući centar znaka. Piksel će postati centar znaka u slici ako je njegova mjera lokalni maksimum u promatranom dijelu slike.

# 2. Korišteni pristup u detekciji prometnih znakova

## 2.1. Segmentacija upotrebom histograma vjerojatnosti boje i Bayesovog teorema

Prometni znakovi su dizajnirani da budu lako uočljivi pri raznim varijantama dnevne svjetlosti što se primarno postiže upotrebom jakih diskriminantnih boja. Zbog toga se smatra da bi boja znaka mogla biti pouzdana značajka za detekciju znakova. Neke poteškoće koje se mogu javiti su pojave da boja znaka blijedi uslijed duge izloženosti sunčevoj svjetlosti. Boja znaka je drugačija ovisno o vremenskim uvjetima kao što su magla, kiša ili snijeg. Oblačno vrijeme i varijacije osvjetljenja te razne sjene značajno mogu promijeniti nijanse boje znaka.

U ovom radu korišten je pristup u detekciji znakova upotrebom uvjetne vjerojatnosti na same piksele slike [6]. Dva histograma vjerojatnosti boje, histogram vjerojatnosti boje znaka i boje pozadine, korišteni su kao spremnici za učenje vjerojatnosti i kasnije za primjenu vjerojatnosnih zakona. Oba histograma imaju $256^3$ elemenata što je jednako broju različitih boja u RGB sustavu. Učenje vjerojatnosti se provodi na odabranom skupu slika za učenje. Slike za učenje su pribavljene kamerom montiranom na autu te se na njima nalaze znakovi od interesa i pozadinski elementi poput ceste, drveća ili zgrada. Prethodno se iz odgovarajućih datoteka dohvaćaju informacije o poziciji i dimenzijama svih znakova od interesa na slikama iz skupa za učenje. U procesu učenja vjerojatnosti iterira se po svim pikselima svih slika za učenje te se za svaki piksel koji pripada traženom prometnom znaku uveća pripadajući element histograma vjerojatnosti boje znaka čiji je indeks jednak boji promatranog piksela. Isto se napravi za sve piksele iz slika za učenje koji ne pripadaju traženom znaku, odnosno pripadaju pozadini te se tada uveća pripadajući element histograma vjerojatnosti boje pozadine čiji je indeks jednak boji promatranog piksela. Nakon što je obrađena i zadnja slika za učenje u elementima histograma nalazi se broj piksela znaka, odnosno piksela pozadine odgovarajuće boje iz cijelog skupa slika za učenje. Prethodno opisano može se pojasniti i na slici 1. koja trenutno predstavlja sliku iz skupa za učenje. Neka je traženi tip znaka kojeg želimo detektirati na skupu slika za testiranje upravo pravokutni žuti znak iz slike 1. označen crvenim okvirom. Cilj je opisane informacije o pikselima slike koji pripadaju traženom tipu znaka spremiti u odgovarajući

histogram. Da bi se saznalo koji pikseli pripadaju znaku, a koji pozadini konzultira se pripadajući redak odgovarajuće datoteke gdje je zapisana informacija o poziciji i dimenziji znaka u promatranoj slici. Upravo ta informacija određuje dimenzije i poziciju crvenog okvira na slici 1. gdje je sada jasno da se pikselima znaka smatraju svi pikseli unutar crvenog okvira, a sve ostalo je pozadina. Da bi se u elementima histograma dobile vrijednosti vjerojatnosti iz intervala [0, 1] svaki element histograma vjerojatnosti boje znaka, odnosno pozadine se dijeli sa zbrojem vrijednosti svih elemenata histograma vjerojatnosti boje znaka, odnosno pozadine. Nakon ove faze u svakom elementu histograma zapisana je uvjetna vjerojatnost koja kaže koliko je vjerojatno da znak, odnosno pozadina sadrži piksel čija je boja jednaka indeksu odgovarajućeg elementa histograma.


Slika 1. Ilustracija učenja histograma vjerojatnosti boje

Time je gotovo učenje vjerojatnosti i pristupa se testiranju, odnosno segmentaciji na skupu slika za testiranje pri čemu se naučene uvjetne vjerojatnosti primjenjuju u Bayesovom teoremu (1).

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)} \tag{1}$$

Za pojašnjenje detekcija na temelju boje i ruba u nastavku će poslužiti slika 1. u ulozi testne slike. Proces segmentacije započinje tako da se iterira po svim pikselima slika

za testiranje i ukoliko je lijeva strana izraza (4) za boju promatranog piksela veća od desne strane tada se taj piksel smatra dijelom prometnog znaka te se ulazi u rekurziju gdje se dalje ispituju njegovi susjedi.

$$P(\omega_O \mid x) > P(\omega_B \mid x) \tag{2}$$

$$\frac{P(x \mid \omega_O) \cdot P(\omega_O)}{P(x)} > \frac{P(x \mid \omega_B) \cdot P(\omega_B)}{P(x)} \tag{3}$$

$$\frac{P(x \mid \omega_O)}{P(x \mid \omega_B)} > \frac{P(\omega_B)}{P(\omega_O)} = \Theta \tag{4}$$

U izrazima (2), (3) i (4) $\omega_O$ označava prometni znak, $\omega_B$ pozadinu, a $x$ boju piksela. Izrazi (3) i (4) su dobiveni primjenom Bayesovog teorema (1) na izraz (2). Uvjetne vjerojatnosti $P(\omega_O \mid x)$, odnosno $P(\omega_B \mid x)$ iz izraza (2) kažu koliko je vjerojatno da boja $x$ promatranog piksela pripada znaku, odnosno pozadini. Ukoliko je lijeva vjerojatnost u (2) veća od desne znači da je vjerojatnije da je promatrani piksel dio znaka pa se u tom slučaju taj piksel prihvaća kao element znaka. Vjerojatnosti $P(x \mid \omega_O)$ i $P(x \mid \omega_B)$ iz izraza (3) i (4) se uzimaju iz odgovarajućih elemenata histograma vjerojatnosti boje znaka i pozadine. Omjer apriorne vjerojatnosti pozadine $P(\omega_B)$ i znaka $P(\omega_O)$ u izrazu (4) pridružen je *theti* ($\Theta$). Konfiguriranjem *thete* moguće je jednostavno mijenjati učinkovitost algoritma. Nakon što postupak pronađe susjedni piksel za kojeg je također zadovoljen izraz (4) ide se razinu dublje u rekurziju. Na povratku iz rekurzije nakon što je pronađena grupa susjednih piksela za koje vrijedi izraz (4) ispituje se veličina grupe. Ukoliko je grupa piksela dovoljno velika tada se sprema u memoriju kao potencijalna pozicija prometnog znaka. Zatim se nastavlja iteracija po ostalim pikselima slike gdje se traži sljedeći neposjećeni piksel za kojeg vrijedi izraz (4). Piksel se označava posjećenim ako se za njega u rekurziji ispostavi da vrijedi (4). Rekurzivno pronalaženje susjednih piksela za koje vrijedi (4) slično je algoritmu narastanja uz primjenu poplavljivanja gdje se provjeravaju svi susjedni pikseli promatranog piksela. Pronađene grupe piksela predstavljaju konačan rezultat detekcije znakova na temelju boje. Nakon što su pronađene grupe piksela u binarnoj slici se svi posjećeni pikseli označe bijelom, a svi ostali pikseli crnom bojom (Slika 2.).

Slika 2. Binarna slika dobivena primjenom izraza (4) na piksele slike 1.

Na binarnoj slici (Slika 2.) vidi se jedna istaknuta grupa piksela koja pripada upravo žutom pravokutnom znaku. Na kraju je još za potrebe testiranja potrebno iscrtati pravokutni okvir oko pronađenih grupa piksela koje zadovoljavaju dimenzije pravokutnih prometnih znakova (Slika 3.).


Slika 3. Konačan rezultat detekcije znakova na temelju boje za sliku 1.

Na slici 3. se vidi da je algoritam uspio pronaći žuti pravokutni znak na temelju boje.

## 2.2. Cannyjev detektor rubova

Cannyjev detektor rubova [7] jedan je od najpoznatijih postupaka detekcije rubova u slici čije se autorstvo pripisuje J. Cannyju. Cannyjev detektor sačinjavaju četiri faze: glađenje, izračun gradijenta, stanjivanje rubova, primjena pragova. Na početku se slika u boji transformira u sivu sliku te se obrađuje Gaussovim filtrom kako bi se uklonio neizbježni šum. Efikasnost i veličina Gaussovog filtra je određena parametrom *sigma* ($\sigma$), tj. standardnom devijacijom. Povećanjem parametra *sigma* uzima se u obzir šire susjedstvo oko promatranog piksela što posljedično rezultira povećanim filtrom, a time i dugotrajnijim procesom glađenja. Na izlazu prve faze dobije se blago zamućena slika. U drugoj fazi detektora se za svaki piksel zaglađene slike računa iznos i smjer pružanja vektora gradijenta. Iznos gradijenta se za svaki piksel računa kao razlika intenziteta susjednih piksela u *x*, odnosno *y* smjeru. U trećoj fazi se provodi stanjivanje rubova na slici iznosa gradijenta. Stanjena slika se u četvrtoj fazi obrađuje s dva zadana relativna praga koji određuju da li je piksel dio ruba ili ne čime se eliminiraju nijanse sive boje te se dobiva binarna slika (Slika 4.). Algoritam ulančavanja pravocrtnih segmenata prima na ulazu u fazu pronalaženje lanaca rubnih piksela navedenu binarnu sliku za daljnju obradu.

Slika 4. Izlaz Cannyjevog detektora ruba za sliku 1.

## 2.3. Gradijent iz slike u boji

Na slikama za testiranje je uočeno da su boje nekih znakova u jakom kontrastu s pozadinom što je dovelo do ideje da bi se rubovi u slici možda mogli učinkovitije pronaći iz slike u boji [8], a ne iz sive slike kao kod Cannya. Da bi se to postiglo potrebno je za svaki piksel odrediti iznos i smjer vektora gradijenta iz slike u boji. Postupak započinje slično kako i kod Cannya tako da se slika u boji zagladi Gaussovim filtrom kako bi se uklonio šum u slici. Glađenje se provodi posebno na svakoj komponenti RGB sustava boje. U sljedećoj fazi gradijent se računa svojstvenom dekompozicijom dvodimenzionalne matrice *Q* (6) čiji su elementi kombinacija komponenti gradijenta odgovarajućeg piksela na svakom od RGB slojeva.

$$J = \begin{bmatrix} r_x & r_y \\ g_x & g_y \\ b_x & b_y \end{bmatrix} \tag{5}$$

$$Q = J^T \cdot J = \begin{bmatrix} r_x^2 + g_x^2 + b_x^2 & r_x \cdot r_y + g_x \cdot g_y + b_x \cdot b_y \\ r_x \cdot r_y + g_x \cdot g_y + b_x \cdot b_y & r_y^2 + g_y^2 + b_y^2 \end{bmatrix} \tag{6}$$

U izrazu (5) $r_x$ i $r_y$, $g_x$ i $g_y$, odnosno $b_x$ i $b_y$ predstavljaju komponente gradijenta promatranog piksela na crvenom, zelenom, odnosno plavom sloju dobivene oduzimanjem intenziteta odgovarajućih susjednih piksela. Rezultat svojstvene dekompozicije matrice *Q* je umnožak matrica (7) gdje je *D* dijagonalna matrica koja na dijagonali sadrži svojstvene vrijednosti matrice *Q*, a stupci matrice *R* su svojstveni vektori matrice *Q*.

$$Q = R^T \cdot D \cdot R \tag{7}$$

Svojstvene vrijednosti dobiju se računanjem korijena karakterističnog polinoma nastalog rješavanjem determinante (8).

$$\left| Q - \lambda \cdot I \right| = 0 \tag{8}$$

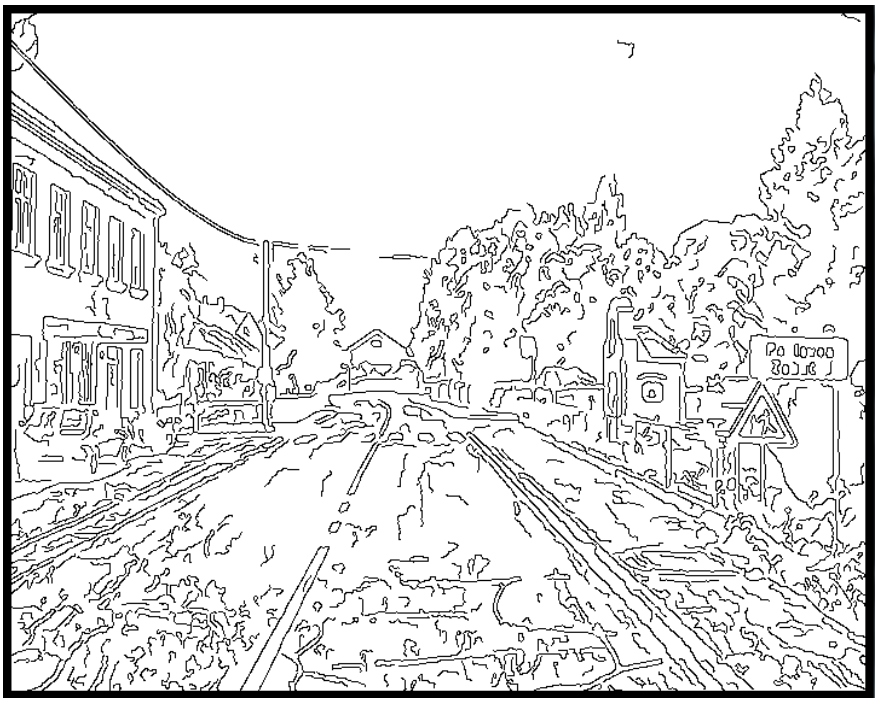
Svojstveni vektori se dobiju uvrštavanjem pronađenih svojstvenih vrijednosti u izraz (9).

$$Q \cdot \vec{v} = \lambda \cdot \vec{v} \tag{9}$$

Za iznos, odnosno smjer gradijenta promatranog piksela uzima se najveća svojstvena vrijednost matrice *Q*, odnosno svojstveni vektor matrice *Q* koji pripada najvećoj svojstvenoj vrijednosti. Dobivena slika iznosa gradijenta prosljeđuje se fazi stanjivanja Cannyjevog detektora na daljnju obradu.

## 2.4. Pronalaženje lanaca rubnih piksela

Ulaz u fazu pronalaženja lanaca rubnih piksela [9] je izlaz Cannyjevog detektora. S detektorom rubova započinje algoritam detekcije znakova na temelju oblika uz pomoć rubova slike. Nakon Cannyjevog detektora poznato je samo koji pikseli slike pripadaju rubu, ali za sljedeće faze u detekciji znakova potrebno je pronađene piksele rubova organizirati u neprekinute lance. Neprekinuti lanci rubnih piksela stvaraju se rekurzivno tako da se funkcija koja sprema piksele lanca u memoriju poziva za susjedne piksele pronađenog piksela ruba. Ukoliko je jedan od susjeda promatranog piksela također dio ruba tada se funkcija dalje poziva za njegove susjede i tako sve dok za trenutni piksel postoje susjedni pikseli ruba. Ukoliko je broj piksela pronađenog lanca manji od zadanog praga tada se taj lanac odbacuje. Nakon ove faze poznati su svi bitni lanci piksela ruba te se prosljeđuju sljedećoj fazi pronalaženja pravocrtnih segmenata. Na slici 5. iscrtani su svi prihvaćeni lanci rubnih piksela dobiveni analizom slike rubova.



Slika 5. Rezultat metode pronalaženja lanaca rubnih piksela za sliku 1.

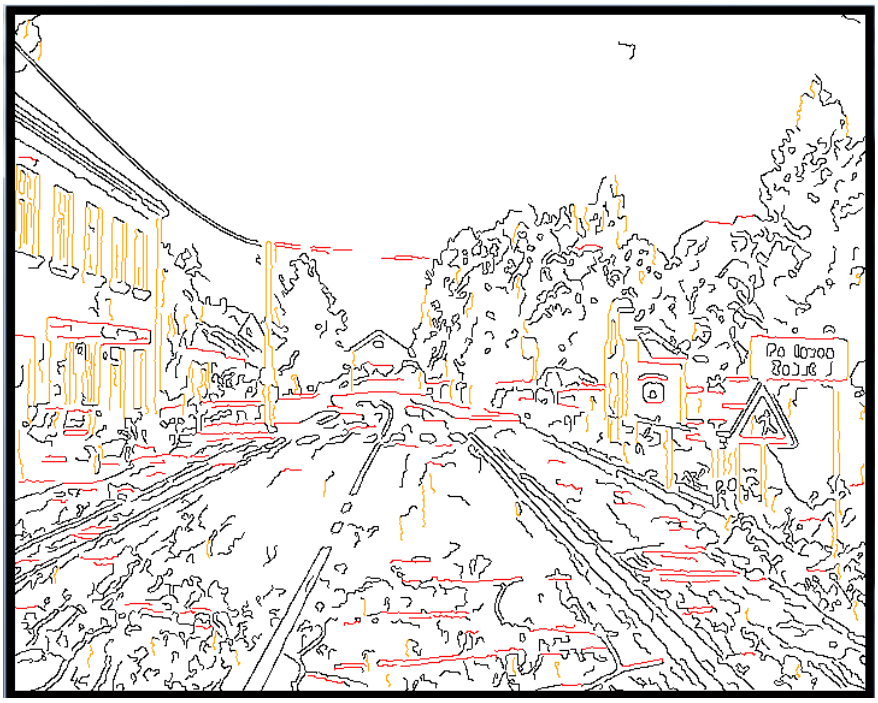
## 2.5. Pronalaženje pravocrtnih segmenata

Cilj ove metode [9] je u prethodno dobivenim lancima rubnih piksela pronaći samo pravocrtne segmente pod kutom od 0° i 90° što je u skladu s oblikom pravokutnih znakova. Metoda započinje tako da se kroz početni i završni piksel svakog lanca povuče imaginarni pravac. Onda se traži piksel lanca s najvećom udaljenosti do tog imaginarnog pravca prema izrazu (10).

$$d = \left| \frac{A \cdot x + B \cdot y + C}{D} \right| \tag{10}$$

$$A = y_1 - y_2 \quad B = x_2 - x_1 \quad C = y_2 \cdot x_1 - y_1 \cdot x_2 \quad D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

U izrazima iznad $x_1$, $x_2$, $y_1$ i $y_2$ su koordinate početnog i završnog piksela lanca. Ako je najveća pronađena udaljenost do lanca veća od zadanog praga tada se lanac dijeli na dva dijela na mjestu piksela s najvećom udaljenosti te se postupak rekurzivno ponavlja za oba dijela lanca. Ukoliko je pronađena maksimalna udaljenost do lanca manja od zadanog praga koji označava maksimalnu dopuštenu devijaciju tada se taj dio lanca smatra pravocrtnim te se dalje provjerava da li taj segment lanca zadovoljava ograničenja duljine i kuta. Kut pravocrtnog segmenta jednostavno se odredi funkcijom arkus tangens budući da su poznati početni i završni pikseli segmenta. Pravocrtni segment se prihvaća i sprema u memoriju samo ako je njegova duljina veća od zadanog ulaznog parametra minimalne duljine te ako dobiveni kut uz odgovarajuće odstupanje ulazi u jednu od sljedeće dvije kategorije: 0° ili 90°. Na slici 6. prikazana je izlazna slika ove faze gdje su bojama označeni prihvaćeni pravocrtni segmenti pod kutovima od 0° i 90°.



Slika 6. Rezultat metode pronalaženja pravocrtnih segmenata za sliku 1.

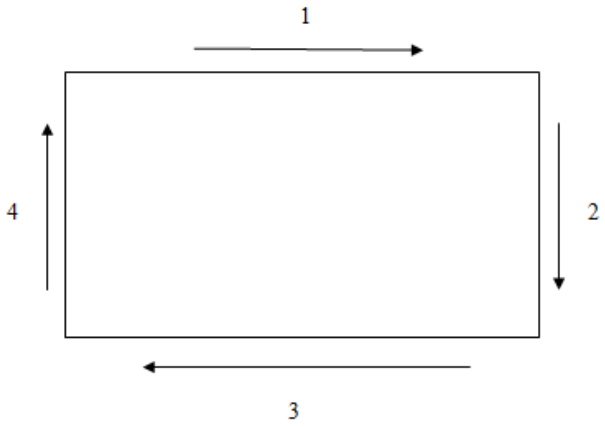
## 2.6. Ulančavanje pravocrtnih segmenata

Ulančavanje pravocrtnih segmenata je zadnji korak u detekciji prometnih znakova na temelju oblika, odnosno rubova. Metoda na ulazu prima pronađene pravocrtne segmente iz prethodnog koraka. Cilj ove metode je iz pronađenih pravocrtnih segmenata konstruirati pravokutne likove koji bi trebali odgovarati pravokutnim znakovima u slici. To se radi tako da se na početku za svaki segment odrede njemu susjedni segmenti ovisno o zadanim pragovima međusobnih udaljenosti. Zatim se rekurzivno pristupa ulančavanju segmenata gdje se susjedni horizontalni i vertikalni segmenti nadovezuju jedan na drugoga sve dok se nadovezivanjem ne zatvori struktura slična pravokutniku. Ako su horizontalne, odnosno vertikalne stranice pravokutnog lika stvorenog nadovezivanjem slične duljine tada se smatra da taj lik označava prometni znak. Detaljnije, ulančavanje započinje tako da se za odabrani horizontalni segment pronađe susjedni vertikalni segment nakon čega se kreće u izradu prve vertikalne stranice pravokutnika. Zatim se na taj vertikalni segment nadovezuje nula ili više susjednih vertikalnih segmenata prije nego se pronađe susjedni horizontalni segment. Nakon pronalaska navedenog horizontalnog segmenta započinje izrada prve horizontalne stranice pravokutnika. Na pronađeni horizontalni segment nadovezuje se nula ili više susjednih horizontalnih segmenata prije nego se pronađe susjedni vertikalni segment. Nakon pronalaska vertikalnog segmenta započinje izrada druge vertikalne

stranice pravokutnika. Na pronađeni vertikalni segment nadovezuje se nula ili više susjednih vertikalnih segmenata prije nego se pronađe susjedni horizontalni segment. Nakon pronalaska horizontalnog segmenta provjerava se da li su konstruirane vertikalne stranice pravokutnog lika slične duljine. Ako je prethodno istinito pristupa se izradi druge, tj. zadnje horizontalne stranice pravokutnika, odnosno ako nije istinito pristupa se novoj izradi druge vertikalne stranice pravokutnika. Pri izradi druge horizontalne stranice pravokutnika na pronađeni horizontalni segment nadovezuje se nula ili više susjednih horizontalnih segmenata sve dok se ne dođe do početnog horizontalnog segmenta čime se konstrukcija pravokutnika zatvara. Ukoliko duljine stranica pravokutnog lika zadovoljavaju zadane pragove duljina tada se taj lik deklarira kao prometni znak. Slika 7. prikazuje rezultat ulančavanja gdje su odabrani segmenti (označeni plavom bojom) žutog pravokutnog znaka nadovezani jedan uz drugoga u pravokutnu strukturu.


Slika 7. Rezultat metode ulančavanja pravocrtnih segmenata za sliku 1.

Na slici 8. se vidi kako se ulančavanje provodi ako se kreće od segmenta označenog s 1 pa na desnu stranu u smjeru segmenta 2. U tom smjeru redom se ulančavaju segmenti označeni brojevima 1, 2, 3 i 4.




Slika 8. Princip ulančavanja pravocrtnih segmenata

Još je samo potrebno zbog testiranja iscrtati pravokutne okvire oko prihvaćenih pravokutnih likova (Slika 9.).


Slika 9. Konačan rezultat detekcije znakova na temelju rubova za sliku 1.

# 3. Programska implementacija

Algoritam detekcije znakova razvijan je unutar ljuske cvsh (*computer vision shell*). Uloga ljuske je pojednostavljenje rada s algoritmima računalnog vida, odnosno njihova učinkovita primjena na ulazne slike i videa. Ljuska cvsh i programski dio ovog diplomskog rada napisani su u programskom jeziku C++.

Glavna programska komponenta koja je ujedno ulaz u proces detekcije znakova nalazi se u datoteci **alg_signDetection.cpp**. U njoj je implementirana klasa `alg_signDetection` koja ima metodu `alg_signDetection::process()` gdje se izvršava cjelokupna detekcija prometnih znakova na temelju boje, odnosno rubova pozivanjem odgovarajućih metoda.

```
virtual void process(
     const img_vectorAbstract& src,
     const win_event_vectorAbstract& events,
     int msDayUtc);
```

Obrada slike odabranim korisničkim algoritmom započinje tako da se algoritam postavi aktivnim i onda ljuska poziva virtualnu metodu `process()` odabranog algoritma. Argument `src` u pozivu metode `process()` označava ulaznu sliku na kojoj se primjenjuju metode detekcije znakova. Unutar metode `process()` moguće je odabrati detekciju znakova na temelju boje, rubova ili objedinjeno na temelju i boje i rubova.

Za rad s pikselima u programskoj implementaciji korištena je struktura `Piksel`.

```
struct Piksel{
     int x;
     int y;
};
```

Struktura je definirana u datoteci **ext_BayesTheorem.hpp**. Prije same detekcije potrebno je za potrebe testiranja saznati pozicije i dimenzije označenih traženih znakova u ulaznoj slici. Za svaku ulaznu sliku postoji redak u datoteci **sekvence3_boja.seq** gdje su zapisane dimenzije i pozicije svih označenih znakova u toj slici. Boja iz naziva datoteke ovisi o boji traženih znakova iz trenutnog testnog skupa slika. Podatci o označenim znakovima spremaju se u vektor `rectangles`.

```
std::vector< std::vector<int> > rectangles;
```

Elementi vektora `rectangles` su također vektori veličine četiri elementa gdje svaki od tih vektora sadrži visinu, širinu te koordinate gornjeg lijevog kuta okvira označenog znaka.

## 3.1. Implementacija detekcije na temelju boje

Prije same detekcije u članske varijable `signHist_` i `backgndHist_` klase `alg_signDetection` učitavaju se naučeni histogrami vjerojatnosti boje znaka i pozadine iz odgovarajućih datoteka. Svaki histogram zahtijeva 128 MB memorije.

```
std::vector<double> signHist_;
std::vector<double> backgndHist_;
```

Detekcija prometnih znakova na temelju boje (poglavlje 2.1.) provodi se pozivom metoda `rekFindSignPixels()` i `findSignPixels()` koje se nalaze u datoteci **ext_BayesTheorem.cpp**.

```
void rekFindSignPixels(int i, int j);
void findSignPixels(double theta_, img_wrap& imgBin);
```

Detekcija znakova započinje pozivom metode `findSignPixels()` kojoj se prosljeđuje ulazni parametar `theta_` koji označava *thetu* iz izraza (4) i binarna slika `imgBin`. Metodi se također preko eksternih deklaracija (`extern`) iz datoteke **ext_BayesTheorem.hpp** prosljeđuju histogrami vjerojatnosti boje i pikseli ulazne slike iz pozivajuće klase `alg_signDetection`.

```
extern double* signHist;
extern double* backgndHist;
extern int* pBits_;
```

Polje `signHist` sadrži vjerojatnosti pojavljivanja odgovarajuće boje traženog znaka, polje `backgndHist` sadrži vjerojatnosti pojavljivanja odgovarajuće boje pozadine, dok polje `pBits_` sadrži piksele ulazne slike. U petlji se iterira po svim pikselima slike i kad se pronađe piksel za čiju boju je zadovoljen izraz (4) tada se za taj piksel poziva rekurzivna funkcija `rekFindSignPixels()`. Uvjetne vjerojatnosti iz izraza (4) se za odgovarajuću boju uzimaju iz histograma vjerojatnosti boje znaka

`signHist`, odnosno pozadine `backgndHist`. Funkcija `rekFindSignPixels()` se principom poplavljivanja poziva rekurzivno za sve susjedne piksele promatranog piksela čija boja također zadovoljava izraz (4). Dubina rekurzije je zbog kapaciteta stoga i smanjenja broja lažno pozitivnih detekcija ograničena parametrima `maxGroupSize` i `maxDepth`. Ukoliko je broj susjednih piksela čija boja zadovoljava izraz (4) nakon povratka iz rekurzije veći od zadanog praga veličine grupe `minGroupSize` tada se ta grupa piksela sprema u spremnik `piks` i predstavlja potencijalnu poziciju prometnog znaka.

```
Piksel* piks;
```

Parametri `minGroupSize` i `maxGroupSize` određeni su razmatranjem dimenzija traženih znakova iz skupa slika za testiranje. Svaki piksel slike posjećuje se samo jednom što se postiže upisom oznake za posjećene piksele u polje `signPixels`. Prije samog izlaska iz funkcije `findSignPixels()` puni se binarna slika `imgBin` na temelju označenih piksela iz polja `signPixels`.

```
int* signPixels;
```

Označavanje grupe pravokutnim okvirom za potrebe testiranja provodi se tako da su koordinate donjeg lijevog kuta okvira određene *x*, odnosno *y* koordinatom piksela s najmanjom *x*, odnosno piksela s najmanjom *y* koordinatom u grupi. Koordinate gornjeg desnog kuta okvira određene su *x*, odnosno *y* koordinatom piksela s najvećom *x*, odnosno piksela s najvećom *y* koordinatom u grupi.

## 3.2. Implementacija detekcije na temelju rubova

Uspješnost detekcije znakova na temelju rubova najviše ovisi o uspješnosti pronalaženja rubova znaka što se postiže detektorom rubova. Zbog toga se ulazna slika na početku obrađuje Cannyjevim algoritmom [7] nakon čega se dobiva slika rubova. Prve dvije faze Cannyjevog algoritma, glađenje i izračun gradijenta, moguće je umjesto na sivoj slici provesti na slici u boji (poglavlje 2.3.). Glađenje slike i izračun gradijenta iz slike u boji ostvaruje se pozivom metoda `colorFilter()` i `colorGradient()` iz datoteka **ext_gladjenje.cpp**, odnosno **ext_gradijent.cpp**.

```
int colorFilter(
      double sigma,
      const img_wrap& imgSrc,
      double* imgFirstSmooth,
      double* imgFinalSmooth);

int colorGradient(
      int margin,
      const double* imgFinalSmooth,
      img_wrap& imgGrad,
      img_wrap& imgPhi);
```

U deklaraciji metode `colorFilter()` parametar `imgSrc` označava ulaznu sliku u boji, `imgFirstSmooth` je slika `imgSrc` zaglađena jednodimenzionalnim Gaussovim filtrom samo u vertikalnom smjeru, dok je slika `imgFinalSmooth` dobivena glađenjem zaglađene slike `imgFirstSmooth` jednodimenzionalnim Gaussovim filtrom u horizontalnom smjeru. U deklaraciji metode `colorGradient()` argument `margin` označava marginu slike, `imgGrad` je slika u koju će se spremiti izračunate amplitude vektora gradijenta, dok je `imgPhi` slika u koju će se spremiti izračunati smjerovi vektora gradijenta. Obje metode vraćaju novu vrijednost margine slike. Unutar metode `colorGradient()` iterira se po pikselima slike te se računaju komponente gradijenta na svakom od RGB slojeva. Zatim se za promatrani piksel poziva metoda `eigen()` gdje se računa svojstvena dekompozicija matrice *Q* (6).

```
void eigen(double q1, double q2, double q3, double q4,
           double * e1, double * e2, double * v);
```

Parametri `q1`, `q2`, `q3` i `q4` predstavljaju elemente matrice *Q*, `e1` i `e2` su pokazivači na komponente svojstvenog vektora koji odgovara najvećoj svojstvenoj vrijednosti `v` matrice *Q*. Pozivom funkcije `quad()` dobije se vrijednost većeg korijena karakterističnog polinoma, odnosno dobije se veća svojstvena vrijednost matrice *Q*.

```
double quad(double a, double b, double c);
```

Parametri `a`, `b` i `c` su koeficijenti kvadratne jednadžbe, tj. karakterističnog polinoma. Nakon što je dobivena slika rubova potrebno je piksele ruba organizirati u povezane strukture. Pozivom metode `findEdgeChains()` iz neorganiziranih piksela ruba dobivenih Cannyjevim detektorom stvaraju se lanci piksela ruba.

```
void findEdgeChains(
     int margin,
     const img_wrap& imgThreshold,
     img_wrap& imgSegment);

void rekFindEdgeChains(
     int margin,
     int i,
     int j,
     img_wrap& imgSegment);
```

Metoda `findEdgeChains()` prima argumente `margin` koji označava marginu slike, sliku rubova `imgThreshold` dobivenu iz Cannyjevog detektora i sliku `imgSegment` u koju će se otisnuti pronađeni lanci piksela ruba. Unutar metode `findEdgeChains()` iterira se po svim pikselima slike `imgThreshold` te se za svaki piksel ruba poziva rekurzivna metoda `rekFindEdgeChains()`. Metoda se dalje poziva za susjedne piksele koji pripadaju rubu. Ako je nakon povratka iz rekurzije pronađeno piksela ruba više od zadanog praga `minChainLength` tada se ti pikseli spremaju u vektor `chains` kao vektor `Piksel` elemenata.

```
std::vector< std::vector<Piksel> >chains;
```

Metode za pronalaženje lanaca rubnih piksela nalaze se u datoteci **ext_findChains.cpp**. Zatim je u dobivenim lancima potrebno pronaći samo pravocrtne segmente kako bi se kasnije mogli konstruirati pravokutni likovi slični oblicima prometnih znakova. Pravocrtni segmenti se stvaraju pozivom metode `findStraightSeg()` iz datoteke **ext_findLines.cpp**.

```
void findStraightSeg(
     win_ann& annLines,
     double lineSize,
     double maxDev);
```

Vektor lanaca `chains` dostupan je metodi `findStraightSeg()` preko eksterne deklaracije u datoteci **ext_findChains.hpp**.

```
extern std::vector< std::vector<Piksel> >chains;
```

Metodi se prosljeđuju anotacija `annLines` u koju se spremaju pronađeni pravocrtni segmenti kako bi se kasnije iscrtali na odabranoj slici, minimalna prihvatljiva duljina pravocrtnog segmenta `lineSize` te maksimalna dopuštena devijacija od pravca `maxDev`. Unutar metode `findStraightSeg()` za svaki lanac iz `chains` poziva se rekurzivna metoda `rekFindStraightSeg()`.

```
void rekFindStraightSeg (
      int a,
      int first,
      int last);
```

Argument `a` označava indeks promatranog lanca u vektoru `chains`, `first` i `last` su indeksi piksela `a`-tog lanca iz `chains` koji označavaju promatrani dio lanca. Metoda za promatrani dio lanca provjerava da li se u tom dijelu nalaze pravocrtni segmenti. To se radi tako da se kroz početni i završni piksel dijela lanca provuče pravac. Zatim se pozivom metode `findMaxDev()` pronađe udaljenost i indeks piksela s najvećom udaljenosti do navedenog pravca (10).

```
MaxDev findMaxDev (int a, int first, int last);
```

Metoda `findMaxDev()` prima iste argumente s kojima je pozvana metoda `rekFindStraightSeg()`. Tip povratnog rezultata je struktura `MaxDev`.

```
struct MaxDev{
      double value;
      int x;
      int y;
      int index;
};
```

Struktura `MaxDev` u `index` sprema indeks piksela s najvećom udaljenosti u promatranom dijelu lanca, `x` i `y` su koordinate navedenog piksela, a `value` predstavlja udaljenost tog piksela do imaginarnog pravca. Ukoliko je najveća pronađena udaljenost manja od argumenta `maxDev` proslijeđenog metodi `findStraightSeg()` i ako je duljina promatranog dijela lanca veća od proslijeđenog argumenta `lineSize` tada se za promatrani dio lanca pristupa računanju kuta. Kut se računa pomoću funkcije arkus tangens `atanl`. Ako je dobiveni kut jednak 0° ili 90° tada se taj dio lanca sprema u vektor pravocrtnih segmenata `lines`, a kut segmenta u vektor kutova `angles`.

```
std::vector< std::vector<Piksel> >lines;
std::vector<int> angles;
```

Kut segmenta se računa uz određeno odstupanje koje se nalazi u varijabli `angleOffset`. Spremljeni pravocrtni segment se po tipu ne razlikuje od lanca rubnih piksela. Pravocrtnom segmentu s indeksom *i* iz `lines` odgovara kut s indeksom *i* iz `angles`. Ako je najveća pronađena udaljenost iz `MaxDev` veća od argumenta `maxDev`

tada se metoda `rekFindStraightSeg()` poziva rekurzivno za dva dijela promatranog lanca nastala diobom na mjestu piksela s najvećom udaljenosti.

U sljedećem koraku pristupa se ulančavanju pravocrtnih segmenata u pravokutne likove koji bi trebali predstavljati potencijalne pozicije prometnih znakova. Ulančavanje započinje pozivom metode `chainStraightSeg()` iz datoteke **ext_chainSegments.cpp**.

```
void chainStraightSeg();
```

Pravocrtni segmenti i pripadajući kutovi dostupni su metodi `chainStraightSeg()` preko eksternih deklaracija iz datoteke **ext_findLines.hpp**.

```
extern std::vector< std::vector<Piksel> >lines;
extern std::vector<int> angles;
```

Unutar metode `chainStraightSeg()` prvo se iz vektora `lines` izdvajaju horizontalni i vertikalni segmenti u posebne strukture `lines90` i `lines0`.

```
std::vector< std::vector<Piksel> > lines90;
std::vector< std::vector<Piksel> > lines0;
```

Zatim se za svaki par pravocrtnih segmenata provjerava susjedstvo te se odgovarajuća oznaka susjedstva upisuje u vektor `neighbours`.

```
std::vector< std::vector<string> > neighbours;
```

Oznake susjedstva su `LD` (*left down*) što označava susjedstvo vertikalnog i horizontalnog segmenta gornjeg lijevog kuta pravokutnika, `RD` (*right down*) što označava susjedstvo vertikalnog i horizontalnog segmenta gornjeg desnog kuta pravokutnika, `LU` (*left up*) što označava susjedstvo vertikalnog i horizontalnog segmenta donjeg lijevog kuta pravokutnika, `RU` (*right up*) što označava susjedstvo vertikalnog i horizontalnog segmenta donjeg desnog kuta pravokutnika, `R` (*right*) i `L` (*left*) što označava susjedstvo dvaju horizontalnih segmenata, `U` (*up*) i `D` (*down*) što označava susjedstvo dvaju vertikalnih segmenata te `0` čime se označava da odgovarajući segmenti nisu susjedni. Parovi vertikalnih i horizontalnih pravocrtnih segmenata smatraju se susjedima ako je njihova međusobna udaljenost manja od `maxPairOfSegsDist`. Parovi dvaju vertikalnih ili dvaju horizontalnih segmenata smatraju se susjedima ako je njihova međusobna udaljenost manja od `maxPairOfSegsDist` te ako je otklon krajnjih piksela odabranog segmenta od krajnjih piksela promatranog segmenta manji od `maxSegDev`.

Nakon što su poznati susjedni pravocrtni segmenti počinje iteriranje po svim horizontalnim segmentima u potrazi za susjednim vertikalnim segmentima. Svaki put kada se za horizontalni segment pronađe njemu susjedni vertikalni segment pozove se rekurzivna metoda `rekChainStraightSeg()`.

```
void rekChainStraightSeg(
     int k,
     string mark,
     string type,
     int turn);
```

Argument `k` u pozivu metode `rekChainStraightSeg()` označava indeks promatranog horizontalnog, odnosno vertikalnog segmenta iz vektora `lines0`, odnosno `lines90` za koji se u trenutnom pozivu metode traži susjedni segment određen oznakom `mark`, `mark` je oznaka traženog susjedstva, `type` se postavlja na `hor` (*horizontal*), odnosno `ver` (*vertical*) kada je promatrani segment horizontalan, odnosno vertikalan, `turn` označava smjernice da li se oznake susjedstva pri pozivu metode komplementarno mijenjaju ili ne. Komplementarno mijenjanje bi značilo npr. ako je za promatrani vertikalni segment pronađen njegov susjedni horizontalni segment gdje je njihovo susjedstvo označeno oznakom `LD` tada će se pri sljedećem pozivu metode tražiti susjedstvo `RD` jer su ta dva prethodno pronađena segmenta dijelovi gornjeg lijevog kuta pravokutnika pa je sada potrebno pronaći desni gornji kut pravokutnika. Ulančavanje pravocrtnih segmenata rekurzivnim pozivima provodi se na način kako je opisano u poglavlju 2.6. uz napomenu da su susjedstva određena oznakama u vektoru `neighbours`. Jedan od uvjeta prihvaćanja grupe ulančanih pravocrtnih segmenata je da se vertikalne stranice stvorenog pravokutnog lika po duljini razlikuju manje od `maxLengthDiff`. Sve grupe prihvaćenih segmenata koje tvore pravokutne likove spremaju se u vektor `signs`.

```
std::vector<int> signs;
```

Na kraju je potrebno pronaći pravokutne okvire kojima će se označiti grupe ulančanih pravocrtnih segmenata za potrebe testiranja. To se za određenu grupu radi na način da je lijeva stranica okvira određena najmanjom *x* koordinatom segmenata lijeve stranice ulančanog pravokutnog lika, donja stranica je određena najmanjom *y* koordinatom segmenata donje stranice ulančanog pravokutnog lika, desna stranica okvira je određena najvećom *x* koordinatom segmenata desne stranice ulančanog pravokutnog lika, a gornja

stranica je određena najvećom *y* koordinatom segmenata gornje stranice ulančanog pravokutnog lika.

## 3.3. Implementacija učenja histograma vjerojatnosti boja

Programska komponenta gdje se obavlja učenje vjerojatnosti nalazi se u datoteci **alg_learnHist.cpp**. U navedenoj datoteci nalazi se implementacija klase `alg_learnHist` koja posjeduje virtualnu metodu `alg_learnHist::process()` gdje je implementirano cjelokupno učenje vjerojatnosti.

```
virtual void process(
     const img_vectorAbstract& src,
     const win_event_vectorAbstract& events,
     int msDayUtc);
```

Nakon što se algoritam učenja vjerojatnosti postavi aktivnim ljuska poziva odgovarajuću virtualnu metodu `process()`. Argument `src` u pozivu metode `process()` označava ulaznu sliku iz koje se uzimaju boje piksela znakova i pozadine. U poglavlju 2.1. opisan je proces učenja vjerojatnosti boje. Histogrami u koje će se spremati pikseli znaka i pozadine i koji će kasnije sadržavati uvjetne vjerojatnosti boje definirani su kao članske varijable klase `alg_learnHist`.

```
std::vector<double> signHist_;
std::vector<double> backgndHist_;
```

Prije obrade ulazne slike potrebno je saznati pozicije i dimenzije označenih znakova kako bi se znalo gdje tražiti piksele znaka, a gdje piksele pozadine u slici. Potrebne informacije nalaze se u odgovarajućim datotekama gdje za svaku sliku postoji redak u kojem su zapisane dimenzije, tip znaka te koordinate gornje lijeve točke okvira znaka za sve tražene znakove u slici. Navedene informacije pohranjuju se za nastavak učenja u vektor `rectangles`.

```
std::vector< std::vector<int> > rectangles;
```

Vektor `rectangles` sadrži vektore veličine četiri elementa za svaki od traženih znakova u slici gdje svaki od tih vektora sadrži visinu, širinu te koordinate gornjeg lijevog kuta okvira pripadajućeg znaka. U radu su za učenje vjerojatnosti boje korištena dva različita skupa slika za učenje pa je bilo potrebno za vrijeme promjene skupova spremiti u memoriju trenutno stanje histograma i ponovno ga učitati. Zbog toga je bilo nužno

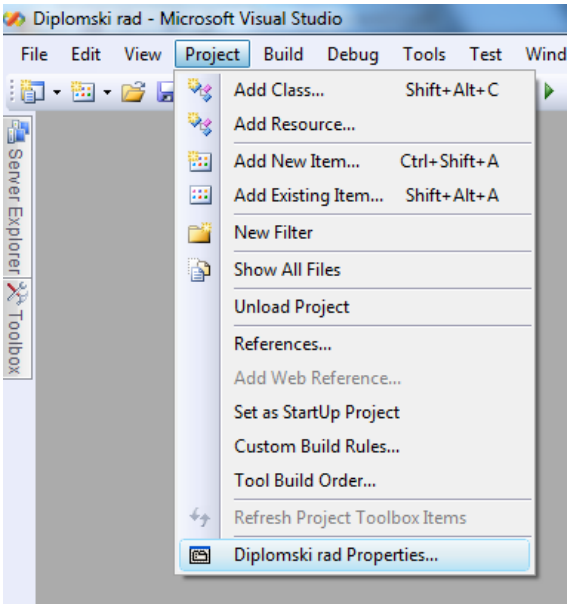
implementirati metodu poput `initLearnHist()` kojom će se spremljeni podatci iz memorije učitati u histograme.

```
void initLearnHist();
```

Kod promjene skupa za učenje, tj. nakon završetka inkrementiranja odgovarajućih elemenata histograma vjerojatnosti boje na temelju prvog skupa, informacije u histogramima se spremaju u datoteku. Prije nastavka učenja na drugom skupu informacije iz datoteke se pozivom metode `initLearnHist()` učitavaju natrag u histograme i učenje se nastavlja na drugom skupu slika.

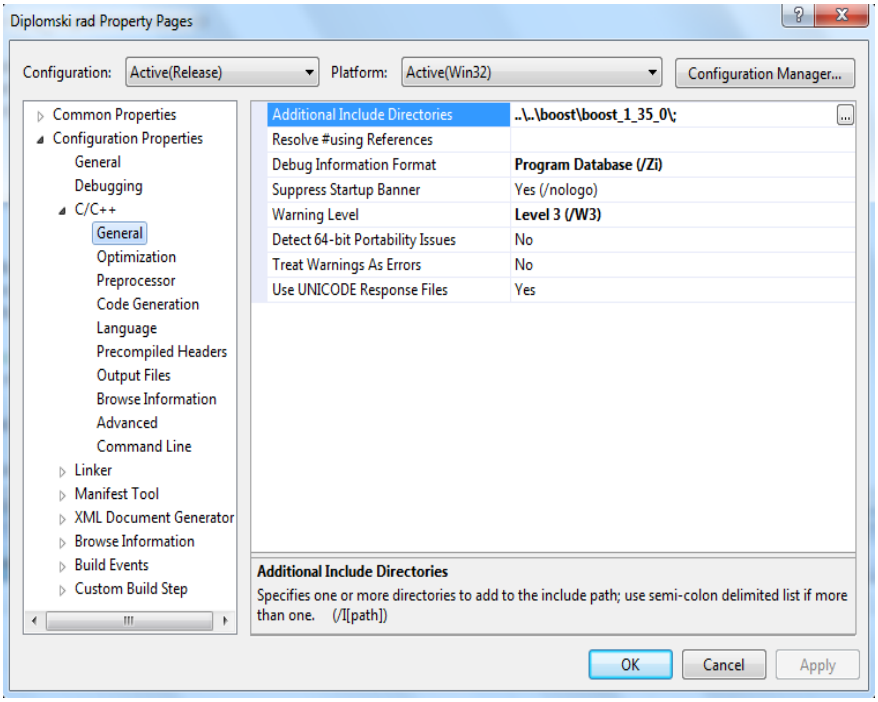
## 3.4. Rad s razvojnim okruženjem i ljuskom cvsh

U izradi programske implementacije detekcije znakova i učenja vjerojatnosti korišteno je razvojno okruženje Visual Studio 2008 Professional Edition. U nastavku će biti opisani koraci u konfiguriranju razvojnog okruženja potrebni za prevođenje i pokretanje korisničkog algoritma unutar ljuske cvsh. Ljuska cvsh je dizajnirana na način da se korisnički algoritmi zajedno s ljuskom prevedu u jedinstveni izvršni kôd. Svi potrebni koraci konfiguracije odvijati će se u kartici *ime_projekta Properties* do koje se dolazi klikom na *Project* iz alatne trake (Slika 10.). Pretpostavlja se da je korisnik već napravio novi projekt u koji je dodao sve komponente izvornog koda ljuske cvsh i odabranog korisničkog algoritma.



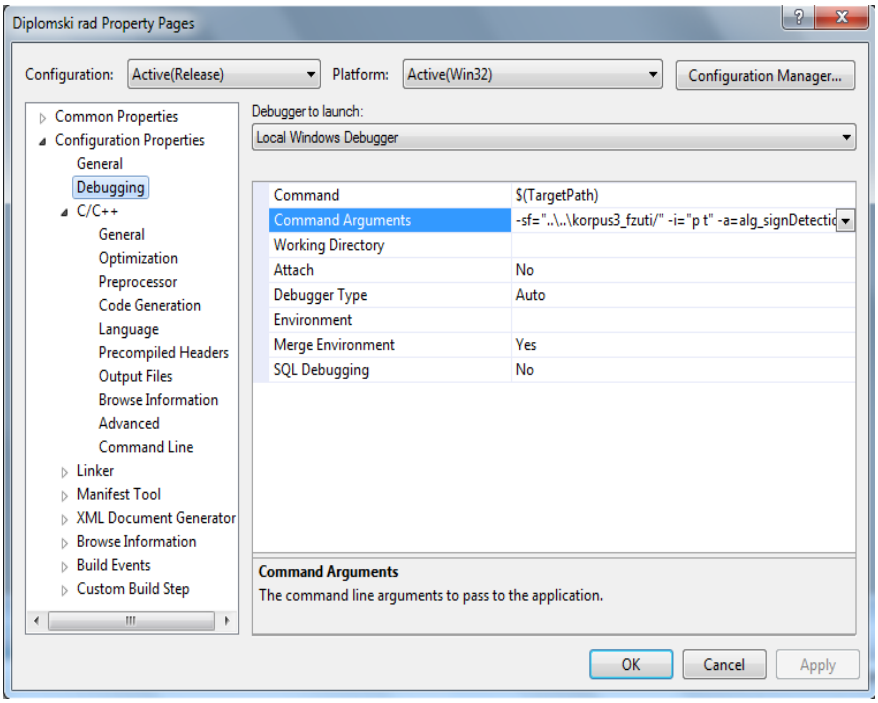

Slika 10. Kartica *ime_projekta Properties*

Za prevođenje koda potrebno je imati odgovarajuće elemente izvornog koda biblioteke Boost koji se mogu preuzeti na lokaciji http://www.boost.org/. Biblioteka Boost se u projekt uključuje tako da se pod *Project → ime_projekta Properties → Configuration Properties → C/C++ → General → Additional Include Directories* upiše putanja do lokacije gdje je biblioteka instalirana (Slika 11.). Moguće je upisati i relativnu putanju u odnosu na datoteku **ime_projekta.vcproj**, odnosno **ime_projekta.sln**.




Slika 11. Povezivanje s bibliotekom Boost

Implementirani algoritmi detekcije znakova i učenja vjerojatnosti u svom radu koriste određene parametre koji im se prosljeđuju putem naredbenog retka. U ovom radu algoritam je testiran samo na slikama, ali ljuska u naredbenom retku prihvaća i videa. Parametri naredbenog retka definiraju se pod *Project → ime_projekta Properties → Debugging → Command Arguments* (Slika 12.).




Slika 12. Unos naredbenog retka

Neki od osnovnih parametara koje je moguće definirati u naredbenom retku ljuske su: parametar sf (*source file*) koji označava putanju do izvorne slike ili direktorija sa slikama koje se obrađuju, a (*algorithm*) je naziv odabranog korisničkog algoritma, i (*interactive*) je prva naredba koja se prosljeđuje korisničkom sučelju te određuje način pokretanja projekta, c (*config*) omogućuje konfiguraciju odgovarajućih parametara korisničkog algoritma.

Primjer kako bi u ovom radu trebao kod detekcije izgledati valjani naredbeni redak:

-sf="..\..\korpus3_fzuti/" -i="p t" -a=alg_signDetection

Argumentu sf je u gornjem primjeru pridružena relativna putanja do direktorija koji sadrži slike za testiranje. Neke od opcija koje prihvaća parametar i (*interactive*) su p n (*process next*) obradi sljedeću, p p (*process previous*) obradi prethodnu ili p t (*process this*) obradi trenutnu. Pomoću opcije c (config) moguće je postaviti vrijednosti određenih parametara algoritma prije samog pokretanja. Konkretno, u ovom radu moguće je prilikom detekcije postaviti vrijednosti *thete*, *sigme*, gornjeg i donjeg relativnog praga kod Cannyjevog algoritma, minimalne duljine pravocrtnog segmenta i maksimalne devijacije piksela od imaginarnog pravca:

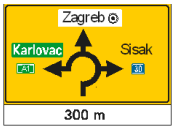
-c="<theta_><sigma_><thHi_><thLo_><lineSize_><maxDev_>“

Korisničko sučelje prilikom rada s algoritmom podržava kretanje kroz ispitne slike ili okvire videa primjenom istih naredbi koje prihvaća parametar i (*interactive*). Iz programa se izlazi zadavanjem naredbe q (*quit*) korisničkom sučelju. Također je moguće postaviti vrijednosti parametara algoritma zadavanjem naredbe c (*config*) korisničkom sučelju na isti način kao i u naredbenom retku.

# 4. Eksperimentalni rezultati

Testiranje ostvarenih metoda provedeno je na prijenosnom računalu HP Compaq nx 7300 s dvojezgrenim procesorom takta 1.6 GHz i 1024 MB radne memorije. Učinkovitost opisanog pristupa ispitana je nezavisno za žute, plave, odnosno crvene znakove na njihovim odgovarajućim skupovima slika za testiranje. Svi žuti znakovi iz skupa slika za testiranje pravokutnog su oblika pa je bilo moguće na cijelom skupu primijeniti oba opisana postupka detekcije na temelju boje i rubova. Plavi znakovi iz skupa slika za testiranje su kružnih i pravokutnih oblika pa je na cijelom skupu smisleno primijeniti samo postupak detekcije na temelju boje. Testni skup slika s crvenim znakovima sadržava samo trokutaste crvene znakove pa je na cijelom skupu također bilo smisleno primijeniti samo postupak detekcije na temelju boje. Iako se naslov rada odnosi na pravokutne znakove, segmentacija bojom je neovisna o obliku prometnog znaka pa je bilo razumno primijeniti predloženu detekciju bojom na trokutaste i okrugle znakove. Postupak detekcije na temelju rubova dao je slabije rezultate od postupka detekcije na temelju boje. Detekcija na temelju ruba dobivenog iz slike u boji dala je nešto bolje rezultate od detekcije na temelju ruba dobivenog iz sive slike. Uočeno je da se oba postupka puno bolje ponašaju na slikama s bližim i većim znakovima. Kako bi se ispitala učinkovitost detekcije većih i bližih znakova iz testnog skupa slika sa žutim znakovima izdvojen je jedan puno manji skup slika u kojima se nalaze samo veći i bliži žuti pravokutni znakovi. Kako bi se i na testnom skupu slika s plavim znakovima mogla primijeniti detekcija na temelju rubova također je izdvojen jedan manji skup slika u kojima se nalaze samo veći, odnosno bliži pravokutni plavi znakovi. Za ispitivanje učinkovitosti detekcije na temelju boje prvo je bilo potrebno naučiti uvjetne vjerojatnosti za primjenu izraza (4) na piksele slike. Učenje je provedeno posebno za žute, plave, odnosno crvene znakove na njihovim odgovarajućim skupovima slika za učenje. Skup slika za učenje žutih znakova sadrži 1795 punih slika na kojima se nalaze samo žuti pravokutni znakovi, skup slika za učenje plavih znakova sadrži 754 pune slike u kojima se ne nalaze plavi znakovi te 420 punih i 754 isječene slike na kojima se nalaze okrugli i pravokutni plavi znakove, dok skup slika za učenje crvenih znakova sadrži 2700 isječenih slika na kojima se nalaze samo trokutasti crveni znakovi i 2700 punih slika na kojima se ne nalaze crveni znakovi. U nastavku su posebno za svaku grupu prikazani svi tipovi znakova zastupljeni u skupu za učenje i/ili u skupu za testiranje. Slovo U,

odnosno T ispod slike znaka označava da je odgovarajući tip znaka prisutan u skupu za učenje, odnosno u skupu za testiranje.

| | | | | | |
|---|---|---|---|---|---|
|  |  |  |  |  |  |
| C79 (T,U) | C80 (U) | C81 (T,U) | C82 (T,U) | D03 (U) | D04 (T,U) |
|  |  |  |  |  |  |
| D09 (T,U) | D10 (T,U) | D12 (T,U) | D17 (U) | D05 (T,U) | C77 (U) |
|  |  |  | | | |
| D16 (T) | D06 (T) | D14 (T) | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |
| B45 (U) | B50 (T,U) | B52 (U) | B51 (U) | B53 (U) | B48 (U) | B46 (U) |
|  |  |  |  |  |  |  |
| B60 (T,U) | B61 (T,U) | B59 (T,U) | B57 (U) | B62 (T,U) | C01 (U) | C02 (T,U) |
|  |  |  |  |  |  |  |
| C06 (U) | C03 (U) | C18 (U) | C19 (U) | C35 (T,U) | C70 (U) | C75 (U) |
|  |  |  |  |  |  |  |
| C77 (U) | C78 (U) | C86 (U) | C94 (U) | C95 (U) | C91 (U) | C79 (T) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |
| B01 (T,U) | A01 (T,U) | A03 (U) | A04 (U) | A05 (T,U) | A06 (U) | A07 (U) | A08 (T,U) | A09 (T,U) |
|  |  |  |  |  |  |  |  |  |
| A10 (T,U) | A11 (T,U) | A12 (U) | A13 (U) | A14 (U) | A15 (U) | A16 (U) | A17 (T,U) | A20 (T,U) |
|  |  |  |  |  |  |  |  |  |
| A22 (U) | A23 (U) | A24 (U) | A25 (U) | A26 (U) | A27 (T,U) | A32 (U) | A33 (T,U) | A34 (T,U) |
|  |  |  |  |  |  |  | | |
| A35 (U) | A39 (U) | A41 (U) | A43 (U) | A44 (T,U) | A45 (T,U) | A46 (U) | | |

Pune slike su cjelovite slike snimljene kamerom montiranom na autu koje osim prometnih znakova sadrže uobičajenu cestovno okruženje poput drveća, kuća i sl., dok isječene slike sadrže samo traženi znak te su dobivene iz punih slika izdvajanjem regije znaka. Pošto za učenje crvenih i plavih znakova nije postojalo dovoljno punih slika trebalo je manjak nadomjestiti isječenim slikama. Problem kod isječenih slika je taj što na njima ne postoji pozadina pa je iz njih bilo nemoguće naučiti histogram vjerojatnosti boje pozadine. Zato su u skupove za učenje crvenih, odnosno plavih znakova stavljene pune slike koje ne sadrže tražene znakove te je za isječene slike korišten nešto drugačiji postupak učenja. Za razliku od učenja iz poglavlja 2.1. u izmijenjenom učenju za svaku isječenu sliku postoji pripadajuća puna slika bez traženog znaka u kojoj se područje veličine isječene slike izdvoji na mjestu gdje je ta isječena slika izdvojena iz vlastite pune slike te se svi ostali pikseli izvan te izdvojene regije koriste u osvježavanju histograma

vjerojatnosti boje pozadine. Učinkovitost detekcije znakova na temelju boje ne ovisi puno o tome koji tipovi, odnosno oblici znakova se nalaze u skupu za učenje i u skupu za testiranje, već je bitnije da su iste boje prisutne na znakovima za testiranje i na znakovima iz skupa za učenje. Kao mjere za ocjenu učinkovitosti implementiranih postupaka korišteni su odziv (11) i preciznost (12).

$$\frac{TP}{TP + FN} \tag{11}$$

$$\frac{TP}{TP + FP} \tag{12}$$

U (11), odnosno (12) TP (*true positive*) označava broj pozitivnih detekcija gdje su pronađeni traženi znakovi, FN (*false negative*) označava broj lažno negativnih detekcija gdje nisu pronađeni traženi znakovi, a FP (*false positive*) označava broj lažno pozitivnih detekcija gdje su pronađeni objekti koji ne pripadaju prometnim znakovima. U ovom radu se pozitivnom detekcijom smatra samo slučaj gdje se pravokutni okvir dobiven detekcijom i pravokutni okvir dobiven iz datoteke koja sadrži informacije o traženim znakovima u trenutnoj slici međusobno preklapaju za više od 60%. Potrebno je reći da je u ovom radu naglasak bio na većem odzivu pa je za određenu metodu najbolji rezultat onaj gdje je za kombinaciju vrijednosti odgovarajućih parametara postignut najveći odziv. Sljedeći parametri imali su konstantne vrijednosti za vrijeme svih testiranja:

- `maxDepth`=3000
- `thHi_`=0.5
- `thLo_`=0.3
- `minChainLength`=15
- `maxDev_`=2.5
- `maxPairOfSegsDist`=15
- `maxSegDev`=4
- `maxLengthDiff`=20
- `lineSize_`=15
- `angleOffset`=12

Parametri `minGroupSize` i `maxGroupSize` ovise o trenutnom skupu slika za testiranje. Određeni su dimenzijama traženih znakova iz trenutnog skupa slika za testiranje. U potrazi za boljim rezultatima odziva i preciznosti samo su parametrima `theta_` i `sigma_` mijenjane vrijednosti. U nadolazećim tablicama se za svaki testni skup nalaze zapisi o vrijednostima parametara kojima su postignuti najbolji rezultati primjenom odgovarajuće metode. Metoda temeljena na rubu testirana je za gradijent dobiven iz slike u

boji i iz sive slike. Kod testnih skupova slika gdje je bilo smisleno primijeniti obje metode detekcije redak u tablicama gdje se nalazi zapis o rezultatu kombinacije dviju metoda ostvaren je pomoću vrijednosti parametara `theta_` i `sigma_` kojima su na tom istom skupu ostvareni najbolji rezultati primjenom svake metode zasebno. U nastavku su također za određeni skup slika i za odgovarajuću metodu detekcije ilustrirane mjere učinkovitosti za nekoliko odabranih vrijednosti parametara `sigma_`, odnosno `theta_`. Slike u nastavku koje pokazuju pozitivne, lažno pozitivne i lažno negativne rezultate detekcije temeljene na boji dobivene su na pripadajućem cijelom testnom skupu za iznos parametra `theta_` iz retka odgovarajuće tablice. Slike na kojima se nalaze pozitivni i negativni rezultati detekcije temeljene na rubu dobivene su na pripadajućem manjem izdvojenom skupu slika za iznos parametra `sigma_` i za vrstu gradijenta kojima je na promatranom skupu postignut bolji odziv za što postoji zapis u odgovarajućoj tablici. Svi znakovi detektirani s obje metode su prije označavanja na ulaznoj slici filtrirani na temelju dimenzija.

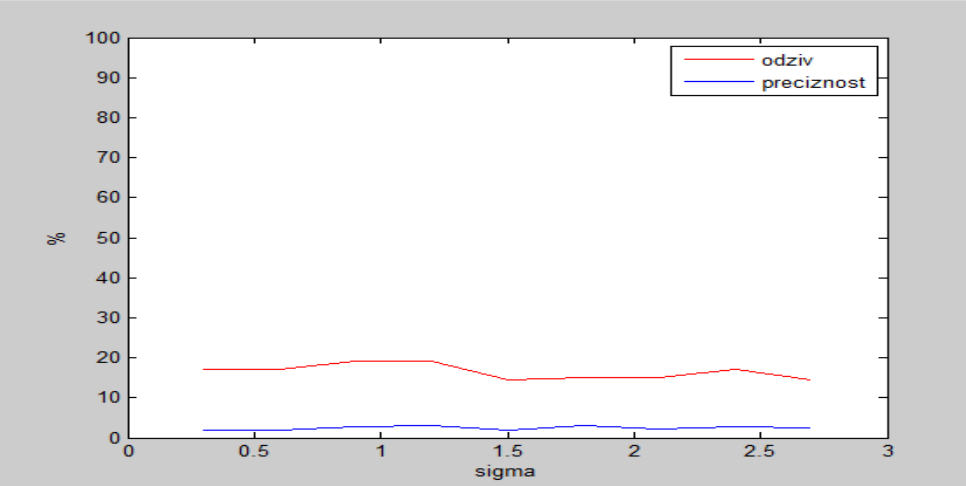
## 4.1. Detekcija žutih prometnih znakova

Cjeloviti skup slika za testiranje žutih znakova sadrži 145 slika na kojima se nalaze 152 tražena žuta pravokutna znaka. Parametri koji ovise o trenutnom testnom skupu imali su sljedeće postavke za skup od 145 slika sa žutim znakovima:

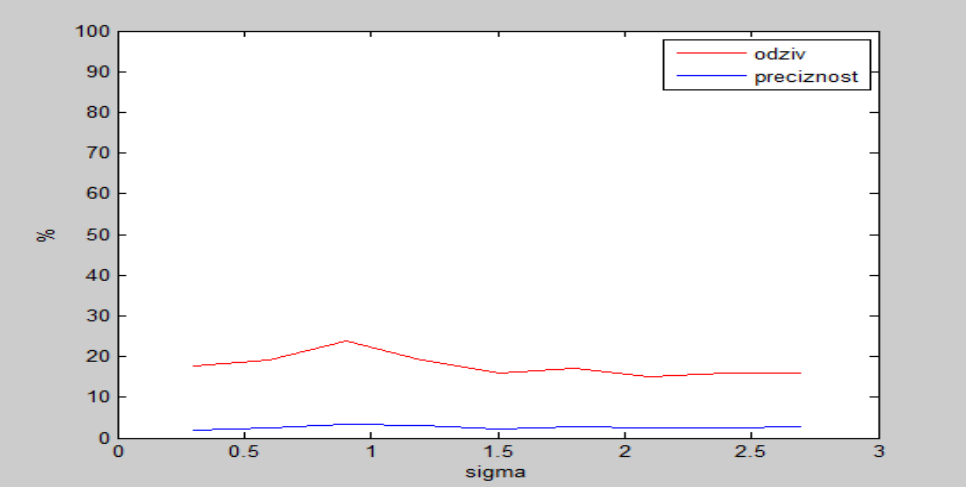
- `maxGroupSize`=132*150
- `minGroupSize`=50

Tablica 1. Najbolji ostvareni rezultati na skupu od 145 slika

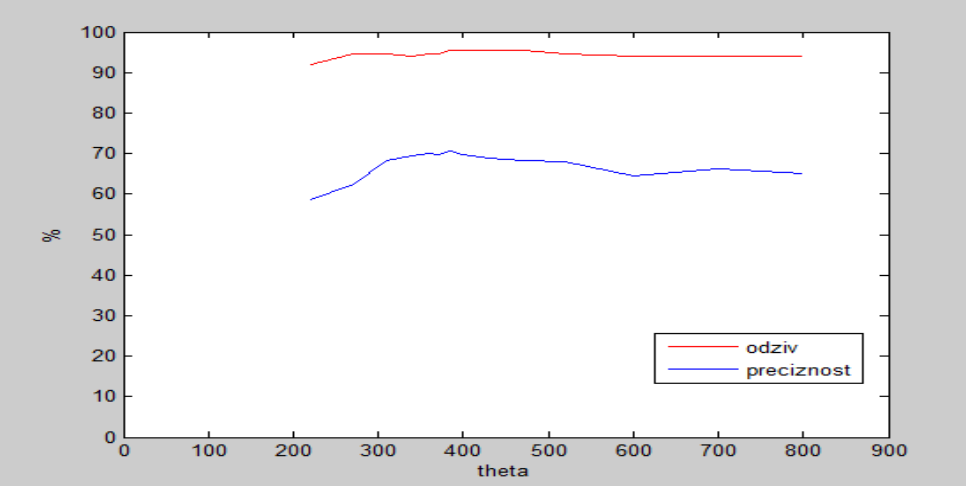
| **detekcija na temelju** | **sigma** | **theta** | **odziv** | **preciznost** |
|---|---|---|---|---|
| ruba dobivenog iz slike u boji | 1.2 | ne koristi se | 19.1% | 3.1% |
| ruba dobivenog iz sive slike | 0.9 | ne koristi se | 23.7% | 3.3% |
| boje | ne koristi se | 385 | 95.4% | 70.7% |
| boje + ruba dobivenog iz slike u boji | 1.2 | 385 | 98% | 13.4% |
| boje + ruba dobivenog iz sive slike | 0.9 | 385 | 97.4% | 11.9% |




Slika 13. Rezultati detekcije na temelju ruba dobivenog iz slike u boji na skupu od 145 slika




Slika 14. Rezultati detekcije na temelju ruba dobivenog iz sive slike na skupu od 145 slika



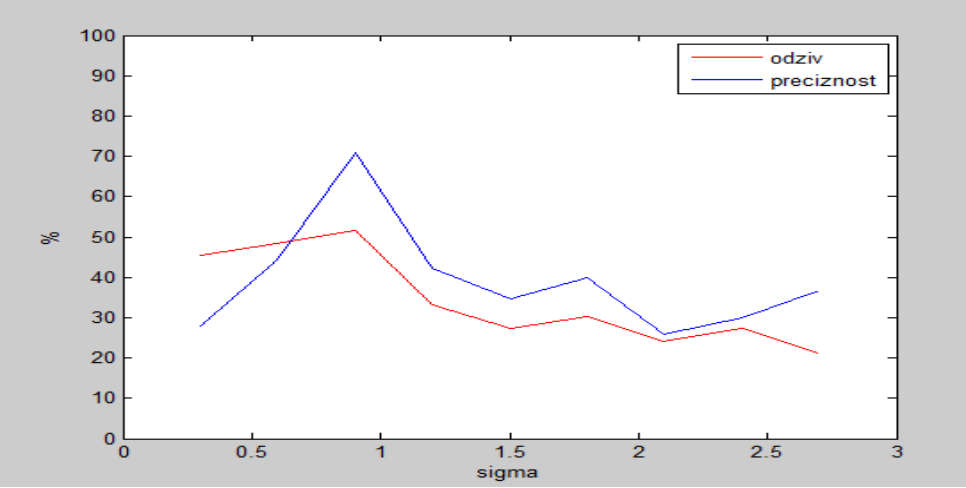

Slika 15. Rezultati detekcije na temelju boje na skupu od 145 slika

Iz skupa od 145 slika sa žutim znakovima izdvojen je manji skup od 32 slike s 33 tražena žuta pravokutna znaka. Izdvojene su samo slike gdje se nalaze bliži i veći znakovi. Parametri koji ovise o trenutnom testnom skupu imali su sljedeće postavke za skup od 32 slike sa žutim znakovima:

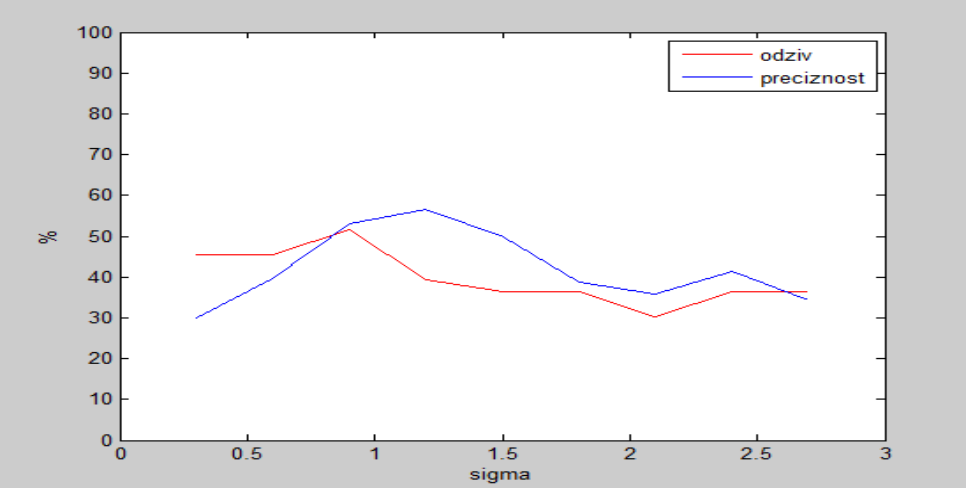
- `maxGroupSize`=132*150
- `minGroupSize`=300

Tablica 2. Najbolji ostvareni rezultati na skupu od 32 slike

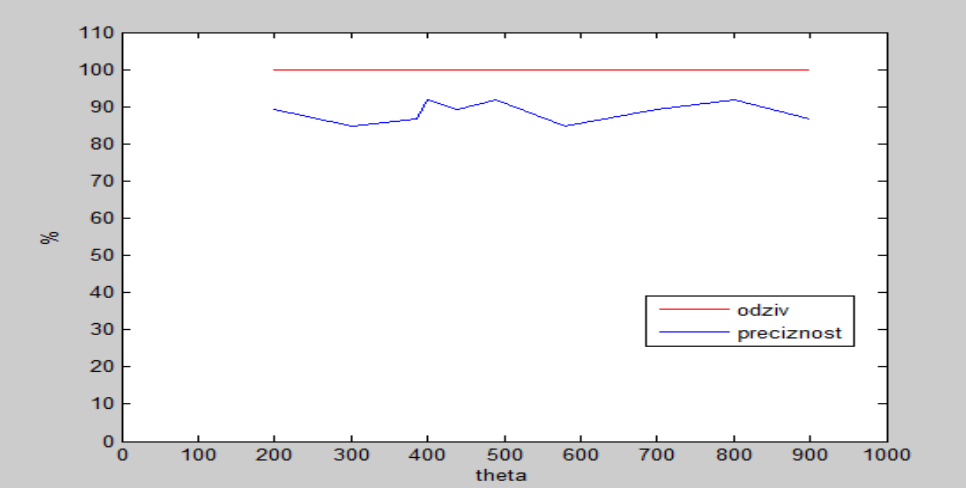
| **detekcija na temelju** | **sigma** | **theta** | **odziv** | **preciznost** |
|---|---|---|---|---|
| ruba dobivenog iz slike u boji | 0.9 | ne koristi se | 51.5% | 70.8% |
| ruba dobivenog iz sive slike | 0.9 | ne koristi se | 51.5% | 53.1% |
| boje | ne koristi se | 490 | 100% | 91.7% |
| boje + ruba dobivenog iz slike u boji | 0.9 | 490 | 100% | 76.7% |
| boje + ruba dobivenog iz sive slike | 0.9 | 490 | 100% | 64.7% |




Slika 16. Rezultati detekcije na temelju ruba dobivenog iz slike u boji na skupu od 32 slike




Slika 17. Rezultati detekcije na temelju ruba dobivenog iz sive slike na skupu od 32 slike



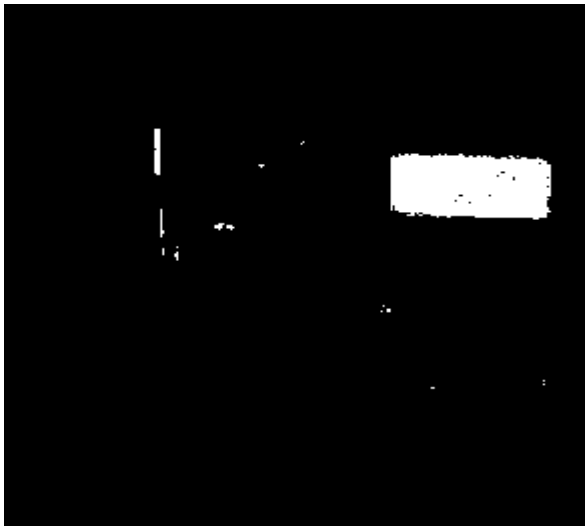

Slika 18. Rezultati detekcije na temelju boje na skupu od 32 slike

Tablica 3. Trajanje postupaka

| boja znaka | skup za evaluaciju | detekcija na temelju | sigma | theta | trajanje |
|---|---|---|---|---|---|
| žuti | skup 1: 145 slika sa 152 tražena znaka | boje | / | 385 | 1.212 s |
| žuti | skup 1: 145 slika sa 152 tražena znaka | ruba dobivenog iz sive slike | 0.9 | / | 0.838 s |

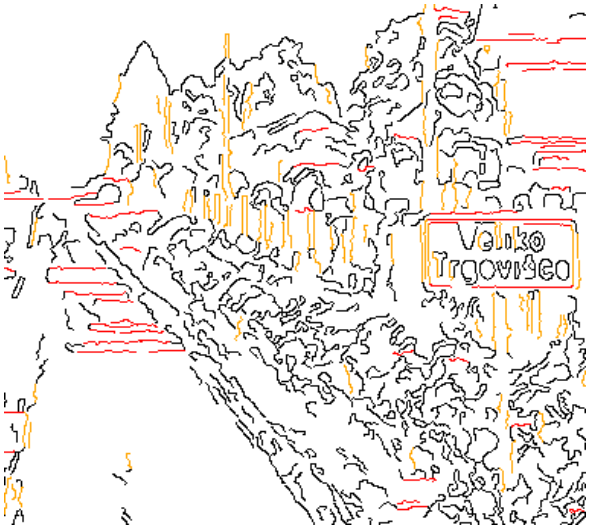
### 4.1.1. Pozitivni rezultati


Slika 19. Pozitivna detekcija na temelju boje



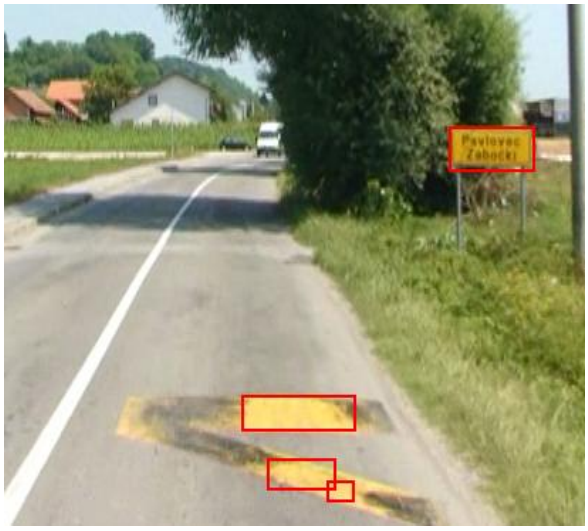
Slika 20. Binarna slika nastala detekcijom znakova u slici 19.


Slika 21. Pozitivna detekcija na temelju ruba




Slika 22. Slika segmenata nastala detekcijom znakova u slici 21.

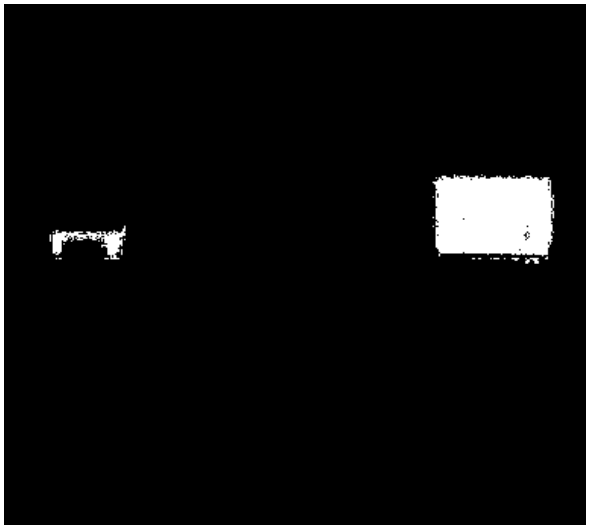
### 4.1.2. Lažno pozitivni rezultati



Slika 23. Lažno pozitivna detekcija na temelju boje

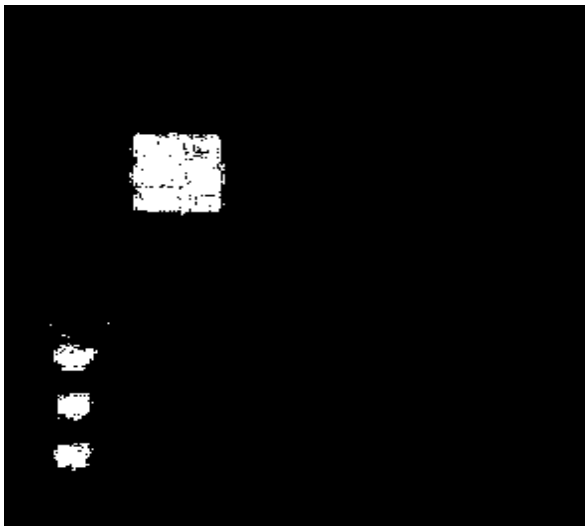
Slika 24. Binarna slika nastala detekcijom znakova u slici 23.

Slika 25. Lažno pozitivna detekcija na temelju boje

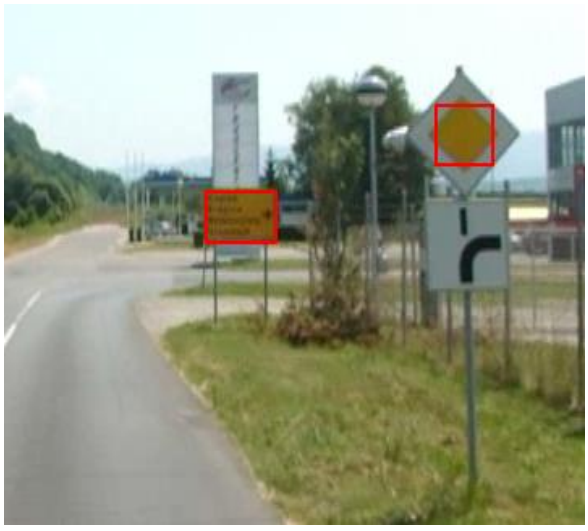


Slika 26. Binarna slika nastala detekcijom znakova u slici 25.

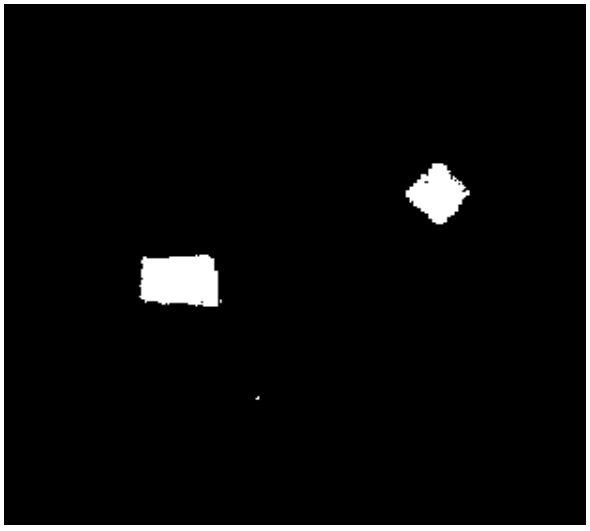

Slika 27. Lažno pozitivna detekcija na temelju boje



Slika 28. Binarna slika nastala detekcijom znakova u slici 27.

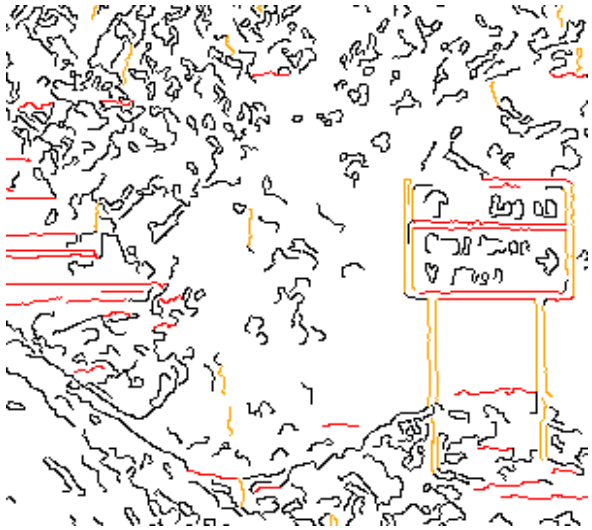


Slika 29. Lažno pozitivna detekcija na temelju boje



Slika 30. Binarna slika nastala detekcijom znakova u slici 29.


Slika 31. Lažno pozitivna detekcija na temelju ruba



Slika 32. Slika segmenata nastala detekcijom znakova u slici 31.

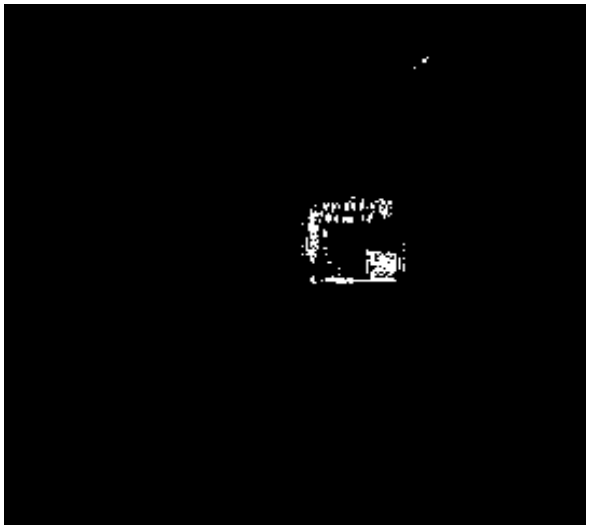
Na slici 23. algoritam je uz ispravno detektirani žuti pravokutni znak neispravno detektirao žute dijelove ceste koji nisu dio prometnog znaka. Tu se vidi nedostatak postupka detekcije na temelju boje gdje će u ovom slučaju veći žuti segmenti slike biti proglašeni prometnim znakom bez razmatranja oblika tih žutih regija. Ista stvar se je dogodila na slici 25. gdje je kao znak prepoznat žuti dio kamiona, na slici 27. su kao znak prepoznati žuti dijelovi stupića, na slici 29. prepoznat je žuti dio romboidnog znaka koji nije u skupu traženih znakova. Na slici 31. detekcijom na temelju ruba neispravno je prepoznat dio slike ispod žutog znaka zbog jakog kontrasta tamnog zasjenjenog drveća u odnosu na svijetlu travu i stupove žutog znaka. Nedostatak metode temeljene na rubu je zanemarivanje boje prepoznate regije slike.

### 4.1.3. Lažno negativni rezultati


Slika 33. Lažno negativna detekcija na temelju boje

Slika 34. Binarna slika nastala detekcijom znakova u slici 33.

Slika 35. Lažno negativna detekcija na temelju boje

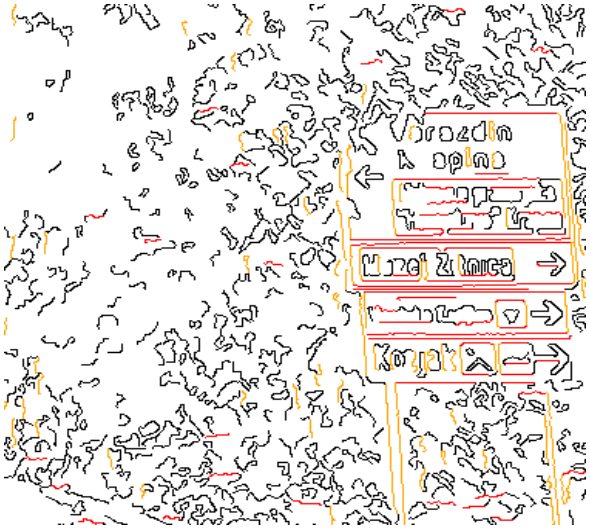


Slika 36. Binarna slika nastala detekcijom znakova u slici 35.


Slika 37. Lažno negativna detekcija na temelju boje

Slika 38. Binarna slika nastala detekcijom znakova u slici 37.


Slika 39. Lažno negativna detekcija na temelju ruba



Slika 40. Slika segmenata nastala detekcijom znakova u slici 39.

Na slici 33. žuti znak nije prepoznat jer za odabrani parametar `theta_` dosta piksela znaka ne zadovoljava izraz (4) pa je regija znaka rascjepkana na male segmente od kojih nijedan ne odgovara dimenzijama prometnog znaka što se vidi na odgovarajućoj binarnoj slici. Na slici 35. prepoznat je samo dio žutog znaka jer u skupu slika za učenje nije bilo dovoljno žutih znakova sa smeđom bojom da bi izraz (4) vrijedio i za smeđe piksele. Na slici 37. znak nije pronađen jer se žuti znak s detektiranim crvenim okvirom preklapa manje od 60%. Uzrok tome su pozadinski crveni pikseli koji su prepoznati kao dio znaka pa je time povećana segmentirana regija na slici 38. što je konačno rezultiralo

većim pravokutnim okvirom. Neuspješna detekcija u slici 39. je posljedica loše detekcije rubova gdje zbog drveća nije prepoznat gornji lijevi kut znaka.

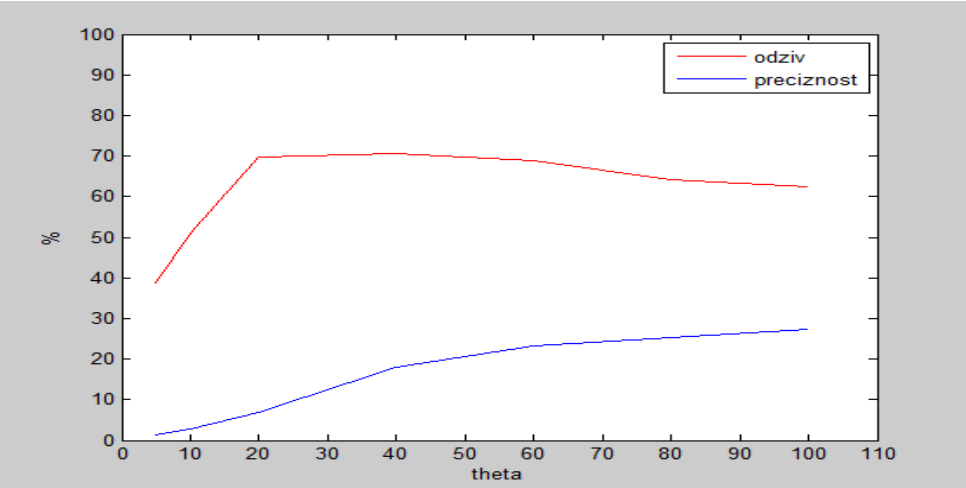
## 4.2. Detekcija plavih prometnih znakova

Cjeloviti skup slika za testiranje plavih znakova sadrži 232 slike na kojima se nalazi 300 traženih plavih znakova. Parametri koji ovise o trenutnom testnom skupu imali su sljedeće postavke za skup od 232 slike s plavim znakovima:

- `maxGroupSize`=103*128
- `minGroupSize`=30

Tablica 4. Najbolji ostvareni rezultati na skupu od 232 slike

| **detekcija na temelju** | **theta** | **odziv** | **preciznost** |
|---|---|---|---|
| boje | 40 | 70.7% | 17.9% |



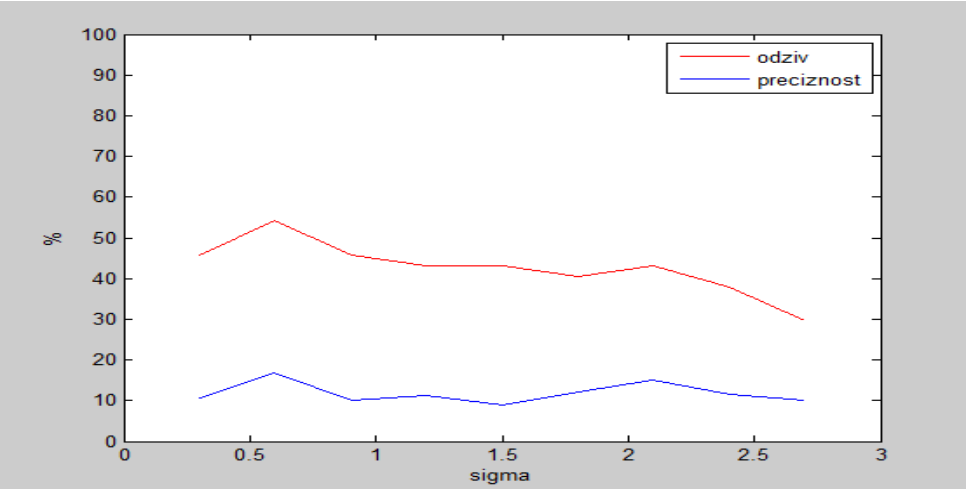

Slika 41. Rezultati detekcije na temelju boje na skupu od 232 slike

Iz skupa od 232 slike s plavim znakovima izdvojen je manji skup od 37 slika s 37 traženih pravokutnih plavih znakova kako bi se na plavim znakovima mogla primijeniti detekcija na temelju ruba. Izdvojene su samo slike gdje se nalaze bliži i veći pravokutni znakovi. Parametri koji ovise o trenutnom testnom skupu imali su sljedeće postavke za skup od 37 slika s plavim znakovima:

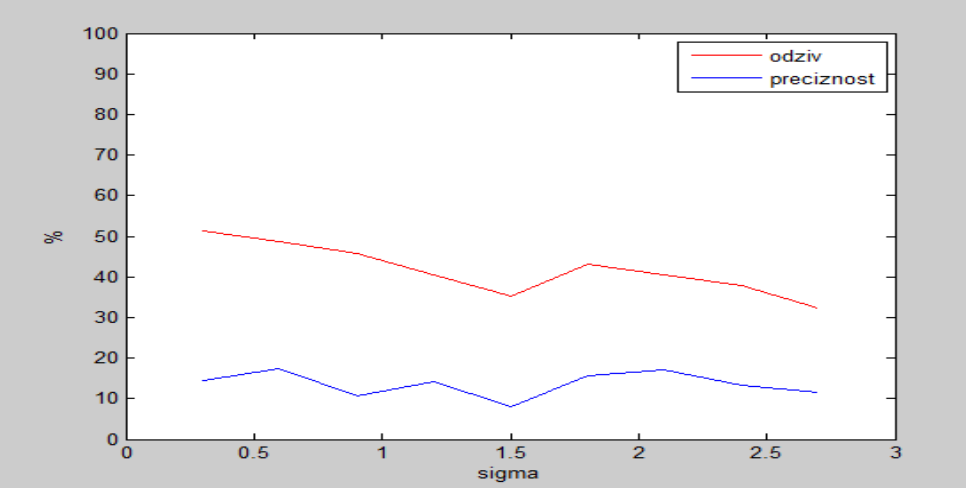
- `maxGroupSize`=103*128
- `minGroupSize`=200

Tablica 5. Najbolji ostvareni rezultati na skupu od 37 slika

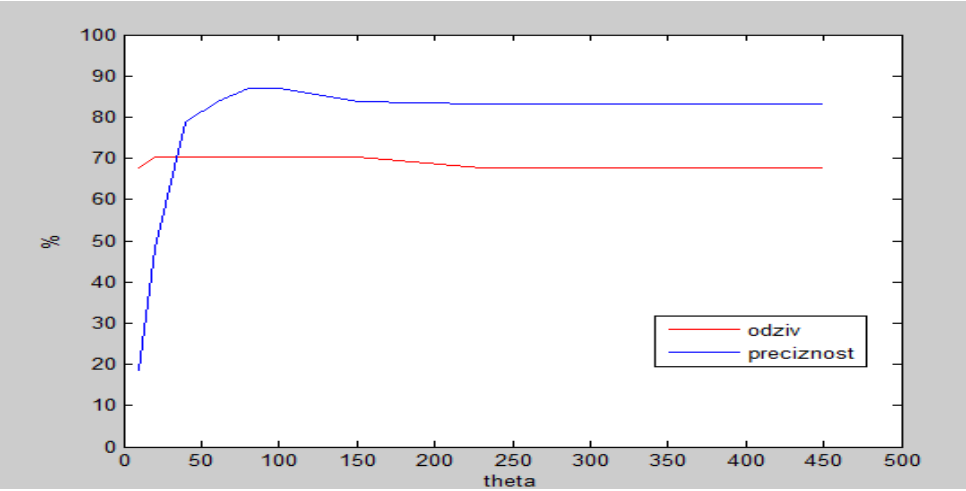
| **detekcija na temelju** | **sigma** | **theta** | **odziv** | **preciznost** |
|---|---|---|---|---|
| ruba dobivenog iz slike u boji | 0.6 | ne koristi se | 54.1% | 16.9% |
| ruba dobivenog iz sive slike | 0.3 | ne koristi se | 51.4% | 14.6% |
| boje | ne koristi se | 100 | 70.3% | 87% |
| boje + ruba dobivenog iz slike u boji | 0.6 | 100 | 78.4% | 22.1% |
| boje + ruba dobivenog iz sive slike | 0.3 | 100 | 78.4% | 20.1% |




Slika 42. Rezultati detekcije na temelju ruba dobivenog iz slike u boji na skupu od 37 slika



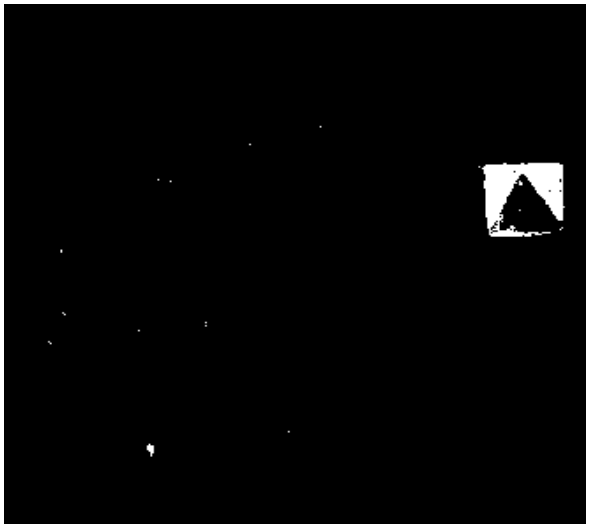

Slika 43. Rezultati detekcije na temelju ruba dobivenog iz sive slike na skupu od 37 slika




Slika 44. Rezultati detekcije na temelju boje na skupu od 37 slika

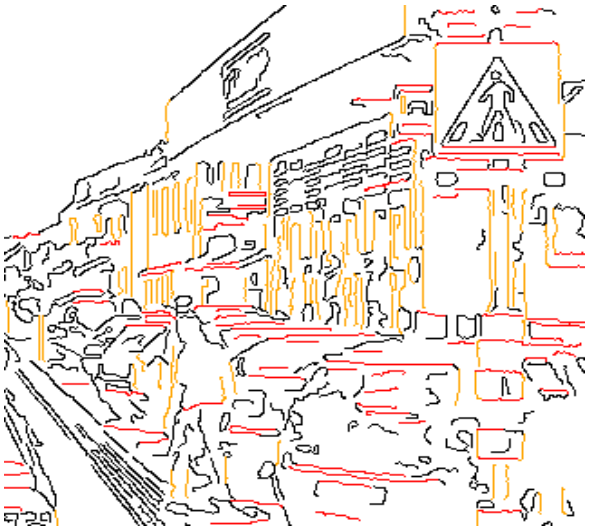
### 4.2.1. Pozitivni rezultati

Slika 45. Pozitivna detekcija na temelju boje



Slika 46. Binarna slika nastala detekcijom znakova u slici 45.

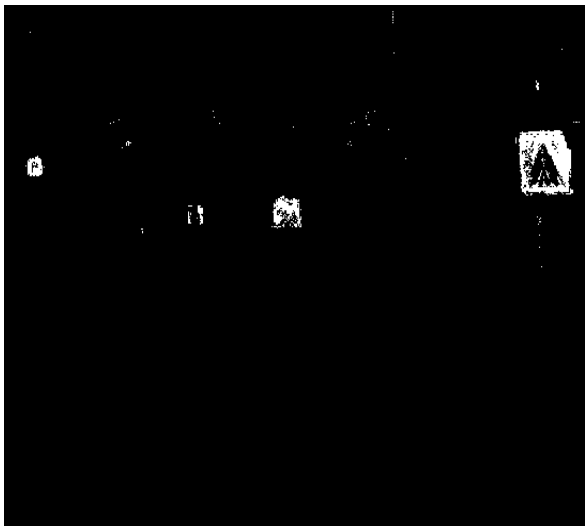
Slika 47. Pozitivna detekcija na temelju ruba



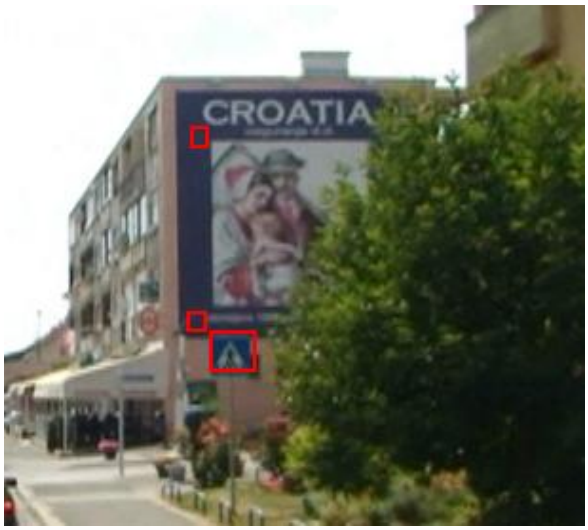
Slika 48. Slika segmenata nastala detekcijom znakova u slici 47.

### 4.2.2. Lažno pozitivni rezultati

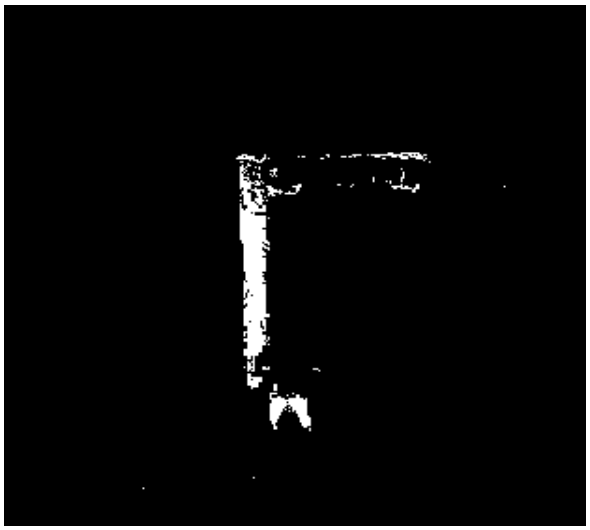
Slika 49. Lažno pozitivna detekcija na temelju boje



Slika 50. Binarna slika nastala detekcijom znakova u slici 49.

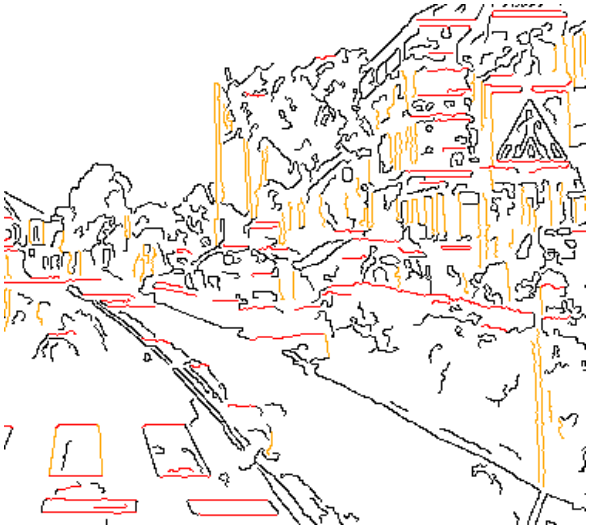



Slika 51. Lažno pozitivna detekcija na temelju boje



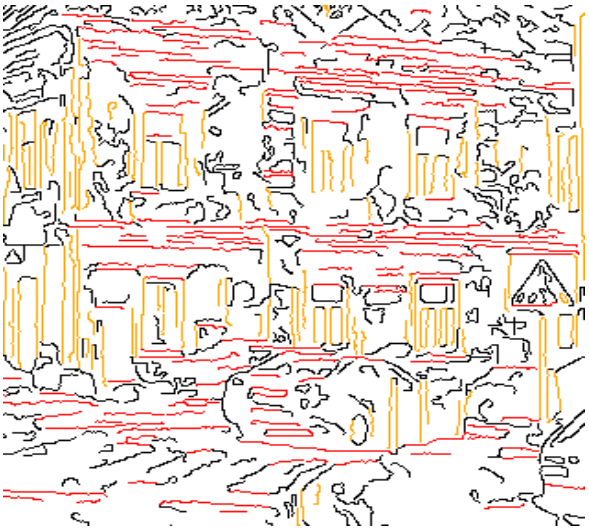
Slika 52. Binarna slika nastala detekcijom znakova u slici 51.

Slika 53. Lažno pozitivna detekcija na temelju ruba



Slika 54. Slika segmenata nastala detekcijom znakova u slici 53.

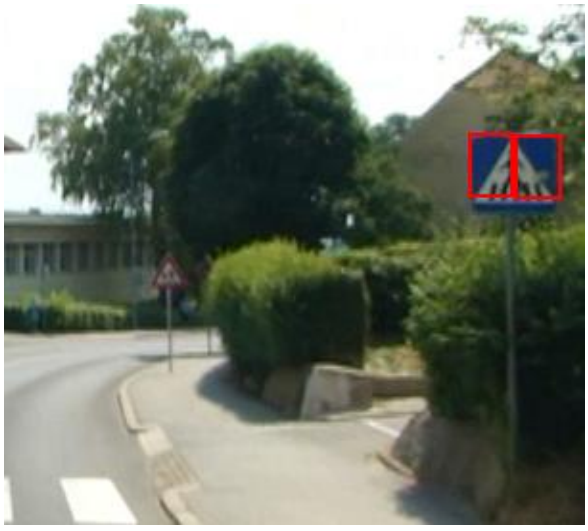
Slika 55. Lažno pozitivna detekcija na temelju ruba



Slika 56. Slika segmenata nastala detekcijom znakova u slici 55.

U potrazi za plavim znakovima na slikama 49. i 51. osim ispravno pronađenih plavih znakova neispravno su detektirani plavi plakati. Na slici 53. neispravno je kao znak prepoznat dio pješačkog prijelaza zbog dovoljno jakog kontrasta u odnosu na sivu cestu. Na slici 55. detektiran je tamni prozor zbog jakog kontrasta u odnosu na bijeli zid zgrade.

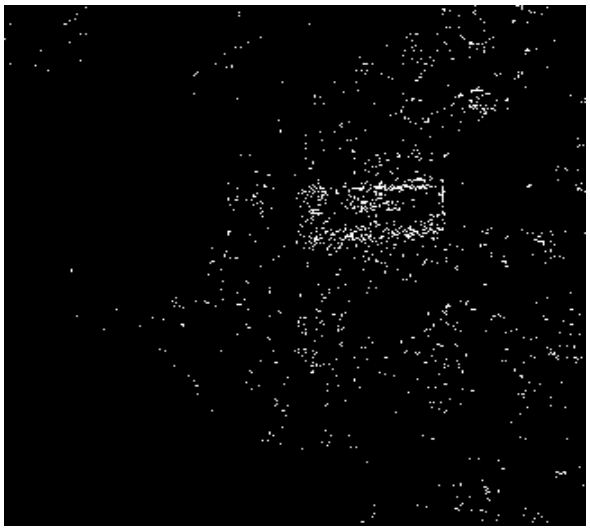
### 4.2.3. Lažno negativni rezultati



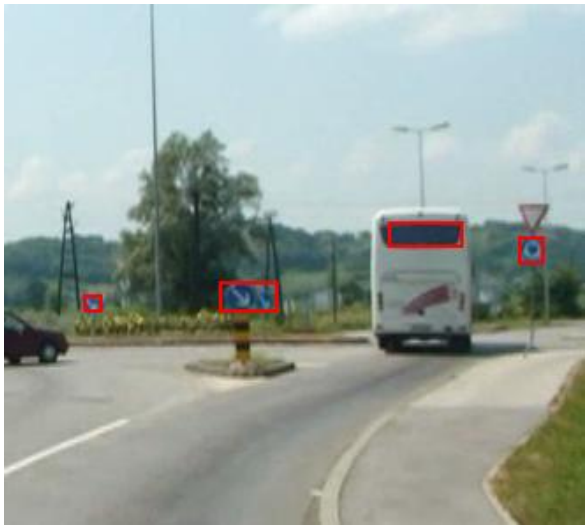
Slika 57. Lažno negativna detekcija na temelju boje

Slika 58. Binarna slika nastala detekcijom znakova u slici 57.

Slika 59. Lažno negativna detekcija na temelju boje



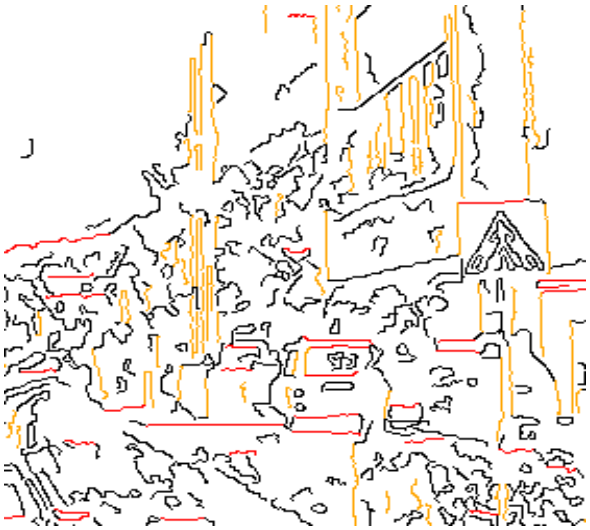
Slika 60. Binarna slika nastala detekcijom znakova u slici 59.



Slika 61. Lažno negativna detekcija na temelju boje

Slika 62. Binarna slika nastala detekcijom znakova u slici 61.

Slika 63. Lažno negativna detekcija na temelju ruba



Slika 64. Slika segmenata nastala detekcijom znakova u slici 63.

Iz sličnih razloga kao na slici 33. i na slici 57. nije pronađen plavi znak. Bijeli pikseli unutar plavog znaka nisu zadovoljili izraz (4) pa su dva plava segmenta od kojih nijedan ne zadovoljava uvjet preklapanja ostala razdvojena. Uzrok tome da bijeli pikseli ne zadovoljavaju (4) je činjenica da dosta bijelih piksela ima među pikselima za učenje vjerojatnosti boje pozadine. Na slici 59. kao i na slici 35. znak nije pronađen jer u slikama za učenje nije bilo dovoljno plavih znakova te nijanse plave boje. Na slici 61. dva plava znaka lijevo od autobusa prepoznata su kao jedan plavi znak zbog toga jer se plave regije dvaju znakova dodiruju. Znakovi nisu detektirani jer je ukupna pronađena regija omeđena crvenim pravokutnikom dovoljno velika da uvjet preklapanja ne bude zadovoljen. Na slici 63. znak nije detektiran jer detektor ruba nije uspio pronaći donji brid plavog znaka.

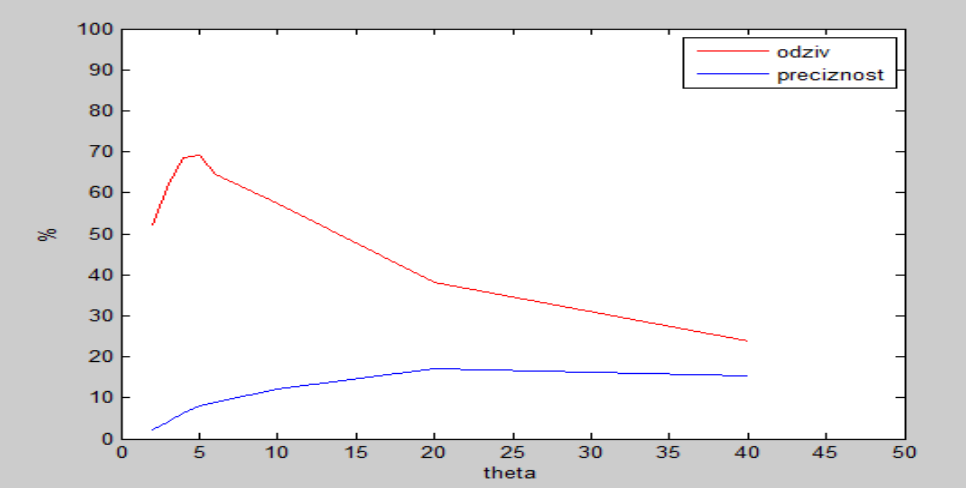
## 4.3. Detekcija crvenih prometnih znakova

Cjeloviti skup slika za testiranje crvenih znakova sadrži 191 sliku na kojima se nalazi 214 traženih crvenih znakova. Parametri koji ovise o trenutnom testnom skupu imali su sljedeće postavke za skup od 191 slike s crvenim znakovima:

- `maxGroupSize`=101*108
- `minGroupSize`=30

Tablica 6. Najbolji ostvareni rezultati na skupu od 191 slike

| **detekcija na temelju** | **theta** | **odziv** | **preciznost** |
|---|---|---|---|
| boje | 5 | 69.1% | 8% |




Slika 65. Rezultati detekcije na temelju boje na skupu od 191 slike

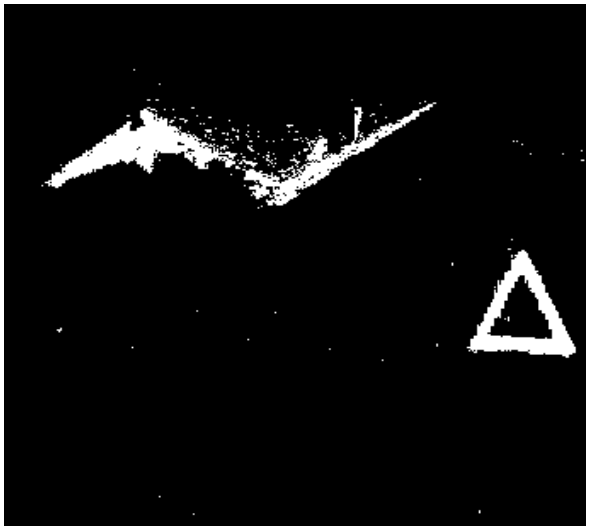
### 4.3.1. Pozitivni rezultati

Slika 66. Pozitivna detekcija na temelju boje

Slika 67. Binarna slika nastala detekcijom znakova u slici 66.

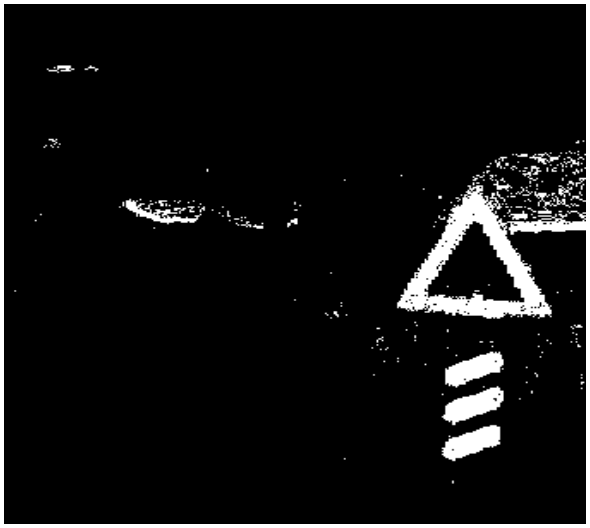
### 4.3.2. Lažno pozitivni rezultati

Slika 68. Lažno pozitivna detekcija na temelju boje



Slika 69. Binarna slika nastala detekcijom znakova u slici 68.

Slika 70. Lažno pozitivna detekcija na temelju boje



Slika 71. Binarna slika nastala detekcijom znakova u slici 70.

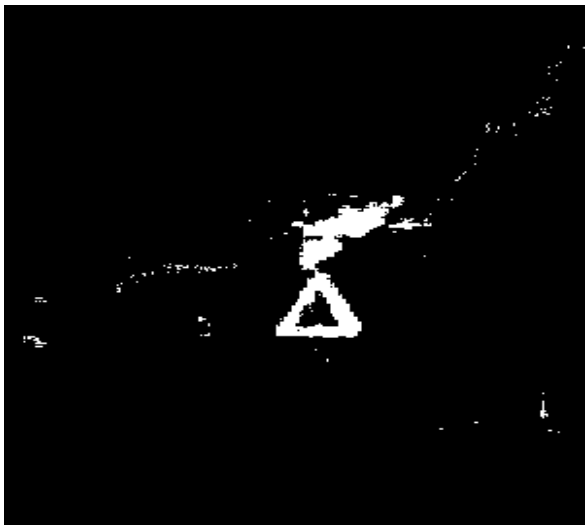
Na slici 68., odnosno 70. uz ispravno pronađene crvene trokutaste znakove neispravno su detektirani crveni krov kuće, odnosno crveni dijelovi stupa znaka.

### 4.3.3. Lažno negativni rezultati

Slika 72. Lažno negativna detekcija na temelju boje

Slika 73. Binarna slika nastala detekcijom znakova u slici 72.

Slika 74. Lažno negativna detekcija na temelju boje



Slika 75. Binarna slika nastala detekcijom znakova u slici 74.

Na slikama 72. i 74. crveni trokutasti znakovi nisu pronađeni iz istih razloga kao i na slici 61. zbog toga jer crvena regija znaka dodiruje crvenu regiju objekta koji nije znak pa algoritam temeljen na boji pronalazi ukupnu spojenu regiju. Dobivena regija najčešće ne zadovoljava uvjet preklapanja površina. Na slici 72. traženi trokutasti crveni znak dodiruje crveni okrugli znak koji nije u skupu traženih znakova, dok na slici 74. regija znaka dodiruje crveni zid kuće.

# Zaključak

Razvijeni pristup detekcije na temelju ruba ima smisla primijeniti jedino kod pravokutnih znakova, dok je pristup temeljen na boji neovisan o obliku traženog znaka. Sukladno očekivanjima da je boja u slici postojana unatoč raznim tipovima osvjetljenja i vremenskih neprilika dobiven je jako dobar odziv od 95% na skupu od 145 slika sa žutim znakovima detekcijom temeljenom na boji. U svim testnim skupovima slika veći je odziv ostvaren detekcijom temeljenom na boji, nego detekcijom temeljenom na rubu zbog osjetljivosti detektora ruba na smanjenje kontrasta traženog znaka i pozadine čime se gubi dio ruba znaka potrebnog u daljnjoj obradi naspram spomenutoj postojanosti boje. Detekcijom temeljenom na boji na skupu slika sa žutim znakovima osim visokog odziva postignuta je i puno veća preciznost u odnosu na slike s plavim i crvenim znakovima zbog toga jer žuta boja nije toliko prisutna u pozadini koliko su u pozadini prisutne crvena, plava i bijela boja kao na krovovima kuća ili cestovnim oznakama. Velika zastupljenost navedenih boja u pozadini utječe na sadržaj histograma vjerojatnosti boje pozadine, tj. na (4) pa je stoga potrebno dosta sniziti vrijednost *thete* da bi se detektirali crveni i plavi znakovi. Snižavanjem *thete* nužno se detektiraju dijelovi slike koji nisu dio prometnih znakova što smanjuje preciznost. Nedostatak postupka temeljenog na boji je potreba za velikim skupom slika za učenje histograma vjerojatnosti boje te osjetljivost na izbor piksela za učenje istih. Za svaku boju traženog znaka potrebno je imati dovoljno slika za učenje na kojima se nalaze znakovi s traženim bojama pomoću kojih će se osvježavati histogrami vjerojatnosti boje znaka. Idealno bi bilo kada se boje traženog znaka ne bi nalazile među bojama piksela za učenje histograma vjerojatnosti boje pozadine. Uočeno je da na metodu temeljenu na boji negativno utječu istobojni znakovi koji se dodiruju, dodirivanje znaka i objekata koji nisu znak iste boje te prisutnost regija boje traženog znaka koje nisu dio prometnog znaka. Metoda temeljena na rubu ovisi o rezultatu detektora ruba pa na nju negativno utječe gubitak kontrasta između znaka i pozadine te zaklanjanje znaka objektima poput drveća. U daljnjem radu mogla bi se s ciljem povećanja učinkovitosti obraditi regija binarne slike pronađena metodom temeljenom na boji u potrazi za oblikom traženog znaka čime bi se možda mogao riješiti problem dodirivanja istobojnih objekata. Također bi se učinkovitost metoda mogla povećati ako bi se detekcijom prometnog traka eliminirao dio slike koji pripada cesti gdje nema prometnih znakova.

# Literatura


[1] Babić, T., Primjena histograma boje u računalnom vidu, Diplomski seminar, Fakultet elektrotehnike i računarstva, 2010.

[2] Babić, T., Lukinić, T., Kovač, D., Popović, K., Rojković, D., Šverko, M., Prepoznavanje znakova, Projekt iz programske potpore, Fakultet elektrotehnike i računarstva, 2008.

[3] Hsien, J. C., Liou, Y. S., Chen, S. Y., Road Sign Detection and Recognition Using Hidden Markov Model, Asian Journal of Health and Information Sciences, Vol. 1, No. 1 (2006), str. 85-100

[4] Ishizuka, Y., Hirai, Y., Segmentation of Road Sign Symbols using Opponent-Color Filters, ITSWC, Nagoya, (2004)

[5] Fang, C. Y., Fuh, C. S., Chen, S. W., Road Sign Detection from Complex Backgrounds, 28.12.2009., http://www.csie.ntu.edu.tw/~fuh/personal/RoadSignDetectionfromComplexBackgrounds.pdf, 16.4.2011.

[6] Šegvić, S., Brkić, K., Kalafatić, Z., Stanislavljević, V., Ševrović, M., Budimir, D., Dadić, I., Mapping and Assessing the State of Traffic Infrastructure: A computer vision assisted geoinformation inventory for traffic infrastructure, ITSC, Madeira, (2010)

[7] Bašić, Š., Čepo, P. Š., Dodolović, I., Dostal, D., Grbić, S., Gulić, M., Horvatin, I., Louč, S., Sučić, I., Cannyjev detektor rubova, Projekt iz programske potpore, Fakultet elektrotehnike i računarstva, 2008.

[8] Starbird, K., Owens, J., Analysis and Characterization of a Color Edge Detection Algorithm, http://graphics.stanford.edu/~jowens/223b/index.html, 14.4.2011.

[9] Louč, S., Pronalaženje granica objekata ulančavanjem rubnih elemenata, Završni rad, Fakultet elektrotehnike i računarstva, 2009.

[10] Keller, C. G., Sprunk, C., Bahlmann, C., Giebel, J., Baratoff, G., Real-time Recognition of U.S. Speed Signs, IEEE Intelligent Vehicles Symposium, Eindhoven, (2008), str. 518-523

# Detekcija obojenih pravokutnih prometnih znakova


## Sažetak

Razmatra se detekcija obojenih znakova metodama temeljenima na boji i rubu. Metoda temeljena na boji pomoću Bayesovog teorema i uvjetnih vjerojatnosti svaki piksel na temelju njegove boje klasificira kao dio znaka ili pozadine. Zatim se susjedni pikseli klasificirani kao dio znaka postupkom poplavljivanja grupiraju u segmente. Uvjetne vjerojatnosti dobivaju se na temelju piksela slika za učenje gdje se u dva histograma vjerojatnosti boje sprema informacija o zastupljenosti određene boje u pikselima znaka, odnosno pozadine. Metoda temeljena na rubu koristi sliku rubova dobivenu Cannyjevim detektorom. Iz slike rubova izdvajaju se lanci rubnih piksela iz kojih se zatim dobivaju pravocrtni segmenti. Postupkom ulančavanja pravocrtni segmenti se nadovezuju jedan na drugoga s ciljem stvaranja pravokutnih likova sličnih pravokutnim prometnim znakovima.



**Ključne riječi:** detekcija znakova, detekcija boje, detekcija ruba, ulančavanje pravocrtnih segmenata, klasifikacija na temelju vjerojatnosti boje, pravokutni oblici


# Detection of coloured rectangular traffic signs


## Abstract

We consider detection of coloured traffic signs using methods based on colour and edge. Method based on colour using Bayes theorem and conditional probabilities classifies every pixel considering its colour as part of traffic sign or background. Then neighbourly pixels classified as part of traffic signs are grouped in segments using flooding method. Conditional probabilities are acquired using pixels from learning images and collecting information in two colour probability histograms about specific colour presence in traffic sign or background pixels. Method based on edge uses image of edges extracted by Canny detector. Image of edges is used for forming chains of edge pixels from which straight segments are then extracted. In chaining method straight segments are being added one after another with goal of making rectangular shapes similar to rectangular traffic signs.



**Key words:** sign detection, color detection, edge detection, chaining straight segments, color probability classification, rectangular shapes