Detekcija prepreka na cesti korištenjem kamere iz automobila

Lukačević, Filip

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni preddiplomski studij Računarstva

Detekcija prepreka na cesti korištenjem kamere iz automobila

Završni rad

Filip Lukačević

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 18.09.2020.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime studenta:	Filip Lukačević
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R4010, 24.09.2019.
OIB studenta:	46865461632
Mentor:	Izv. prof. dr. sc. Irena Galić
Sumentor:	Dr. sc. Krešimir Romić
Sumentor iz tvrtke:	
Naslov završnog rada:	Detekcija prepreka na cesti korištenjem kamere iz automobila
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	18.09.2020.
Datum potvrde ocjene Odbora:	23.09.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 13.10.2020.

Ime i prezime studenta:	Filip Lukačević
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R4010, 24.09.2019.
Turnitin podudaranje [%]:	7

Ovom izjavom izjavljujem da je rad pod nazivom**: Detekcija prepreka na cesti korištenjem kamere iz automobila**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Irena Galić

i sumentora Dr. sc. Krešimir Romić

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Prepoznavanje objekata i svrstavanje istih u kategorije je jedan od najosnovnijih mehanizama koji čovjeku omogućavaju snalaženje u svijetu. To je sposobnost koja je gotovo unikatna ljudima, te je razvijena do te mjere da nam omogućuje prepoznavanje objekata u varijabilnim uvjetima – različita orijentacija objekta, različito osvjetljenje, i slično. Prema trenutnim neurofiziološkim saznanjima, prihvaćeni model prepoznavanja objekta možemo podijeliti u četiri faze [1]:

- Faza 1 – obrada osnovnih karakteristika objekta – oblik, boja, dubina, …
- Faza 2 – grupiranje sličnih karakteristika koja omogućuje formiranje oblika objekta detekcijom rubova
- Faza 3 – dobivena vizualna reprezentacija objekta se uspoređuje sa dosad čovjeku poznatim strukturalnim opisima
- Faza 4 – vizualnoj reprezentaciji objekta pridodajemo određena semantička obilježja na temelju dosadašnjih iskustava čime se objekt svrstava u kategoriju poznatu i smislenu ljudima

Proces prepoznavanja i klasifikacije objekata je čovjeku intuitivan, no njegova replikacija u kontekstu računalnih programa je izuzetno kompleksan problem. Generalno prepoznavanje objekata u neposrednoj blizini je uglavnom riješeno korištenjem senzora, no oni ne mogu praviti distinkciju između očekivanog ponašanja i položaja objekta, nego samo detektirati udaljenost. Grana znanosti koja se bavi tim problem je računalni vid. Ona pokušava na temelju digitalne reprezentacije fotografija i video zapisa pridodati istima semantičko značenje koje drugi sistemi mogu razumjeti i na temelju rezultata prikladno reagirati.

U današnje vrijeme sustavi za autonomnu vožnju vozila postaju sve učestaliji. Možemo ih klasificirati po stupnjevima automatizacije – od najnižeg stupnja koji samo izdaje upozorenja ili eventualno kratkoročno preuzima kontrolu u kritičnim situacijama, preko srednjeg stupnja koji omogućuje automatizirane procese ubrzavanja, kočenja te upravljanja smjerom vozila, pa sve do vozila koja ne zahtijevaju nikakvu intervenciju čovjeka.

Oni najučestaliji u današnjim vozilima su sustavi koji omogućuju detekciju prometnih traka, znakova, pješaka i drugih automobila u prometu. Zbog kompleksnosti detekcije i klasifikacije takvih objekata u vožnji, područje strojnog učenja je najprikladnije za suočavanje s tim izazovima.

Strojno učenje je podskup šire grane znanosti – umjetne inteligencije. Algoritmi strojnog učenja koncipirani su na način da izvode zadatke na temelju prepoznavanja obrazaca i sposobnosti dedukcije, a ne na izvršavanju strogo zadanih slijednih programskih naredbi. Stvaraju matematički model na temelju ulaznih podataka i unaprijed označenih ili klasificiranih izlaznih podataka (engl. *training data*) koji kasnije koriste za klasifikaciju ili predviđanje.

1.1 Pregled područja

Iako danas algoritmi utemeljeni na dubokom učenju, poput R-CNN [2], Mask R-CNN, YOLO, predstavljaju standard u detekciji i klasifikaciji objekata [3], u ovom radu ograničit ćemo se na tehnike strojnog učenja.

Kao osnovni zadatak potrebno je implementirati klasifikator koji će kao ulaz primiti fotografiju digitalnog formata, a kao izlaz dati informaciju nalaze li se na fotografiji neke od prepreka koje klasifikator može prepoznati. U ovom radu ćemo se ograničiti na klasifikaciju dvaju potencijalnih prepreka, tj. subjekata u prometu – druge automobile te pješake. Istraživanja su pokazala kako su ta dva subjekta najčešći učesnici u prometnim nesrećama te stoga čine dobre kandidate za ovaj problem. Takav pristup je moguće proširiti na proizvoljan broj klasa, čime dobivamo kompetentniji i kompletniji sustav detekcije objekata i dolazimo korak bliže ostvarivanju autonomije vozila

Tehnika koju ćemo koristit za klasifikaciju objekata je stroj potpornih vektora (engl. *Support Vector Machines*, dalje u tekstu SVM). To je tehnika strojnog učenja čiji su standardni oblik predložili Vladimir Vapnik i Corinna Cortes 1995.godine [4], te su nam za svaku instancu SVM modela potrebni trening podatci podijeljeni u dvije označene kategorije – one koji pripadaju traženoj kategoriji te oni koji ne pripadaju.

Osim podjele podataka u dvije klase, potrebno je izdvojiti određena svojstva na temelju kojih ćemo obaviti klasifikaciju. U ovom radu fokusirat ćemo se na metodi histograma usmjerenih gradijenata

(engl. *Histogram of Oriented Gradients*, dalje u tekstu HOG) koji se pokazao kao dobar izbor za detekciju subjekata u prometu.

Kako bismo u potpunosti riješili problem detekcije objekta, osim klasifikacije objekta moramo riješiti još jedan problem – lokalizaciju objekta. Klasifikacija objekta nam daje samo informaciju nalazi li se na fotografiji objekt, no za prosuđivanje potencijalne opasnosti prepreka potrebno je znati i lokaciju danog objekta. Za ovaj korak upotrijebiti ćemo tehniku klizećeg prozora (engl. *sliding window technique*). Kada znamo lokaciju objekta unutar fotografije možemo primijeniti tehnike mijenjanja percepcije kako bismo procijenili relativnu udaljenost od subjekta.

Cilj ovog rada je dati pregled metode detekcije objekata strojnim učenjem, implementirati odabranu metodu programskim rješenjem, te analizirati dobivene rezultate. Kao glavni programski jezik odabran je Python, pošto sadrži potrebne metode i proširenja u vidu biblioteka otvorenog koda. Rješenje će biti implementirano u radnom okruženju Jupyter Notebook, koje je prikladno za pregled međurezultata metode.

U drugom poglavlju rada dan je teorijski pregled metoda koje će biti korištene, a treće poglavlje sadrži programsku implementaciju danog rješenja. Četvrto poglavlje sadrži pregled i analizu rezultata implementiranog rješenja.

2. DETEKCIJA OBJEKATA TEHNIKAMA STROJNOG UČENJA

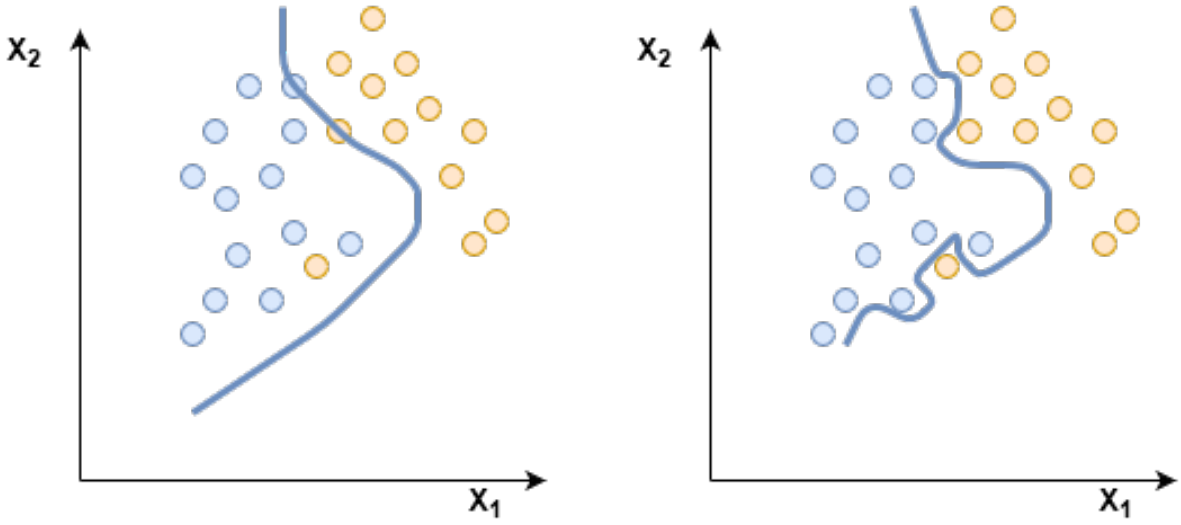
U ovom poglavlju sagledat ćemo matematičku i teorijsku podlogu predloženog rješenja.
Za računalni program se kaže da uči iz iskustva *E*, u odnosu na dani zadatak *T* i mjeru performansa *P*, ako se mjera performansa *P*, na zadatku *T*, povećava s iskustvom *E*, prema [5].

Potkategorija strojnog učenja koju ćemo koristiti je nadzirano učenje (engl. *supervised learning*). To je proces induktivnog učenja pri kojem, na temelju predefiniranih parova ulaz-izlaz, dolazimo do funkcije koja u skladu s danim parovima dodjeljuje izlaz odgovarajućem ulazu. Zadatak nadziranog učenja definiran je na sljedeći način, prema [6]:

Neka je dan skup za učenje N (engl. training set) dan kao niz uređenih parova ulaz – izlaz:
$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, *gdje je svaki* y_1 *generiran nepoznatom funkcijom* $y = f(x)$.
Otkrij funkciju h *koja aproksimira stvarnu funkciju* f.

Učenje je u ovom slučaju proces pronalaska i odabira odgovarajuće funkcije *h* koja ima zadovoljavajuće rezultate. Rezultate testiramo posebnim skupom - skupom za testiranje (engl. *test set*), koji je različit od skupa za učenje. Kada y_i može poprimiti vrijednosti samo iz konačnog skupa vrijednosti *Y* (npr. automobil, kamion, bicikl, pješak) tada je riječ o klasifikaciji. Ukoliko skup *Y* sadrži samo dvije moguće opcije riječ je o binarnoj klasifikaciji.

Može se dogoditi da primjeri za učenje nisu dovoljni da bi se na temelju njih došlo do jednoznačnog rješenja. Situacija u kojoj više potencijalnih funkcija zadovoljavajuće opisuje parove ulaz – izlaz dan je na slici 2.1. Iz navedenoga proizlazi da nije moguće donijeti konačan zaključak bez uvođenja određenih pretpostavki. Princip koji se vrlo često koristi je princip Occamove britve – težnja odabiru najjednostavnijeg rješenja [7]. U primjeru danom na slici 2.1. u tom slučaju odabrali bi funkciju pod a) za naš model.



Slika 2.1. Funkcije koje razdvajaju parove ulaz – izlaz, iz [8]

2.1. Histogram usmjerenih gradijenata

2.1.1. Izdvajanje opisnih značajki

Fotografije u svom sirovom, digitalnom formatu nisu idealan izvor za konstrukciju modela. Naime, fotografije u tom formatu sadrže vrlo velike količine informacija, što rezultira dugim i zahtjevnim treniranjem modela, kompleksnim modelima koji su teško razumljivi ljudima, te fenomenom poznatim kao prokletstvo dimenzionalnosti u kojem, zbog velike i izrazito specifične raspodjele karakteristika, postaje teško klasificirati ulazne podatke i dobiti statistički značajan i pouzdan rezultat.

Kako bismo izbjegli taj problem potrebno je ulazne podatke obraditi, te iz njih izvući podskup značajki koje karakteriziraju dani ulazni podatak. Taj proces nazivamo selekcija i izdvajanje opisnih značajki. Proces je utemeljen na pretpostavci da postoje značajke ulaznog podatka koje su ili irelevantne ili redundantne, te ih time možemo ukloniti. Kao rezultat dobivamo takav podskup značajki koji dovoljno precizno opisuje originalni podatak bez velikog gubitka informacija.
Sam proces je kompleksan, te je potrebno specifično znanje problematike kojom se bavimo kako bismo uspješno mogli odrediti koji podskup značajki je optimalno izdvojiti.

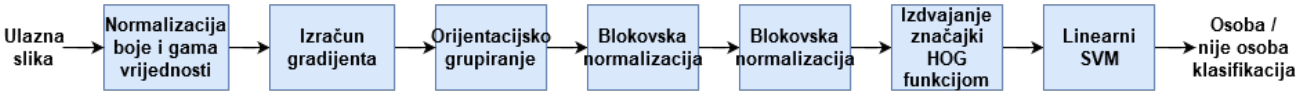
Reprezentacija originalnog podatka skupom značajki zove se opisnik značajki (engl. *feature descriptor*). Opisnik sadrži informacije, najčešće u obliku vektora, o određenim značajkama svakog

pojedinog piksela – poput boje, teksture, obrisa, itd. Za potrebe ovog rada odabrat ćemo opisnik histogram usmjerenih gradijenata.

2.1.2. Algoritam metode histogram usmjerenih gradijenata

Koncept iza histogram usmjerenih gradijenata predstavljen je u svom osnovnom obliku 1986. godine, u radu McConnella [9], no u području računalnog vida postaje popularan 2005. godine, nakon što su Dalal i Triggs objavili svoj rad Histogram of Oriented Gradients for Human Detection [10].

Osnovna ideja na kojoj počiva ova metoda je činjenica da se obris lokaliziranog objekta može opisati pomoću distribucije intenziteta gradijenta ili usmjerenja ruba. Histogram usmjerenih gradijenata ima nekoliko ključnih prednosti u odnosu na slične metode – pošto se algoritam primjenjuje na lokalizirane dijelove slike, na njega ne djeluju fotometrijska i geometrijska izobličenja. Također, iz istog razloga, algoritam je posebno dobar za detekciju pješaka, pošto nije bitno u kojem je točno položaju ljudsko tijelo.



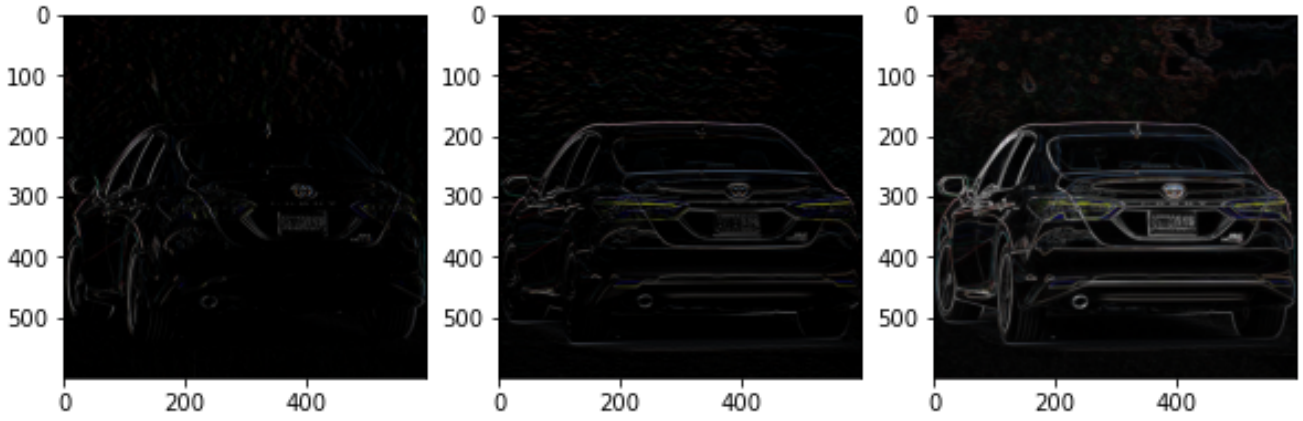
Slika 2.2. Dijagram toka algoritma metode histogram usmjerenih gradijenata

Kao i u sličnim algoritmima, prvi korak je normalizacija boje te gama vrijednosti fotografije. Prema nalascima Dalala i Higgsa, ovaj korak nije nužan u algoritmu jer nema veliki utjecaj na performans.

2.1.2. Izračun gradijenta

Stoga, prvo što moramo napraviti je izračunati horizontalni i vertikalni gradijent svakog pojedinog piksela. Najlakša metoda je primijeniti filter na vrijednosti boje fotografije koristeći sljedeće vektore:

$$[-1, 0, 1], [-1, 0, 1]^T$$



a) b) c)

Slika 2.3. Apsolutne vrijednosti gradijenta: a) x gradijent, b) y gradijent, c) x i y gradijent

Na slici 2.2 možemo vidjeti apsolutne vrijednosti x i y gradijenata. Iznosi apsolutnih vrijednosti su visoke tamo gdje postoji nagla promjena intenziteta smjera. Kao što možemo vidjeti, rezultat izostavlja mnoge redundantne informacije, kao boja pozadine i ostavlja nam samo obris objekta. Shodno gore navedenim vektorima, slika a naglašava vertikalne linije, dok slika b naglašava horizontalne linije.

2.1.3. Orijentacijsko grupiranje

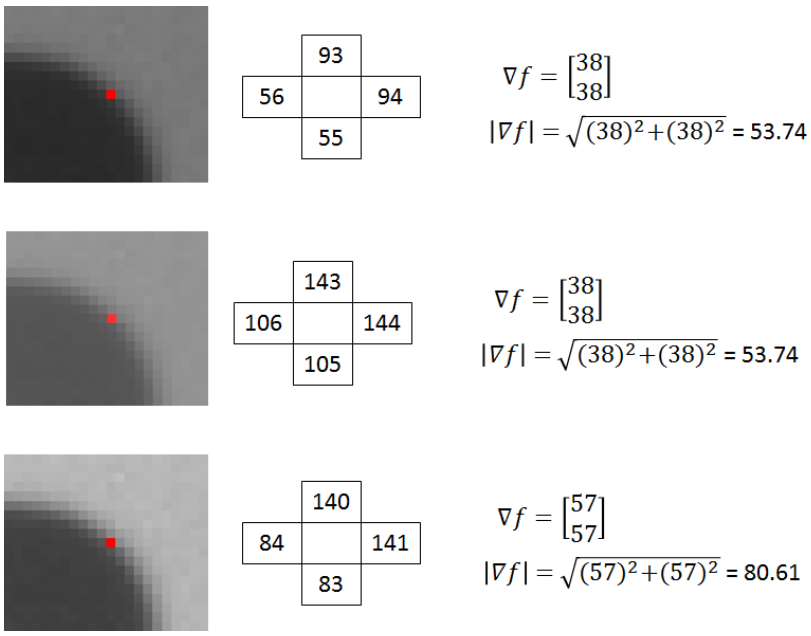
Kako bi se izbjegla velika odstupanja među pojedinim susjednim pikselima, smanjio utjecaj šuma na rezultat, te kako bi kompaktnije prikazali podatke i time povećali performans, grupiramo piksele u ćelije proizvoljne veličine. Svaki piksel predstavlja vektor određen jačinom te smjerom gradijenta.

Sljedeći korak je stvoriti histogram gradijenata za svaku ćeliju. Broj orijentacija dijagrama određujemo eksperimentalnim putem, a zatim svakoj orijentaciji proporcionalno dodjeljujemo pripadajući dio vrijednosti intenziteta gradijenta.

2.1.4. Blokovska normalizacija

Glavni problem koji blokovska normalizacija pokušava riješiti je normalizacija odstupanja nastala zbog faktora izvan naše kontrole, kao npr. različito osvjetljenje, odsjaj, itd. Ukoliko učinimo fotografiju dvostruko svjetlijom (množeći vrijednosti svakog piksela sa 2), intenzitet gradijenta će se

udvostručiti, što će rezultirati i dvostruko većim vrijednostima u histogramu. Ovu pojavu možemo vidjeti na slici 2.3.



Slika 2.4. Primjer blokovske normalizacije

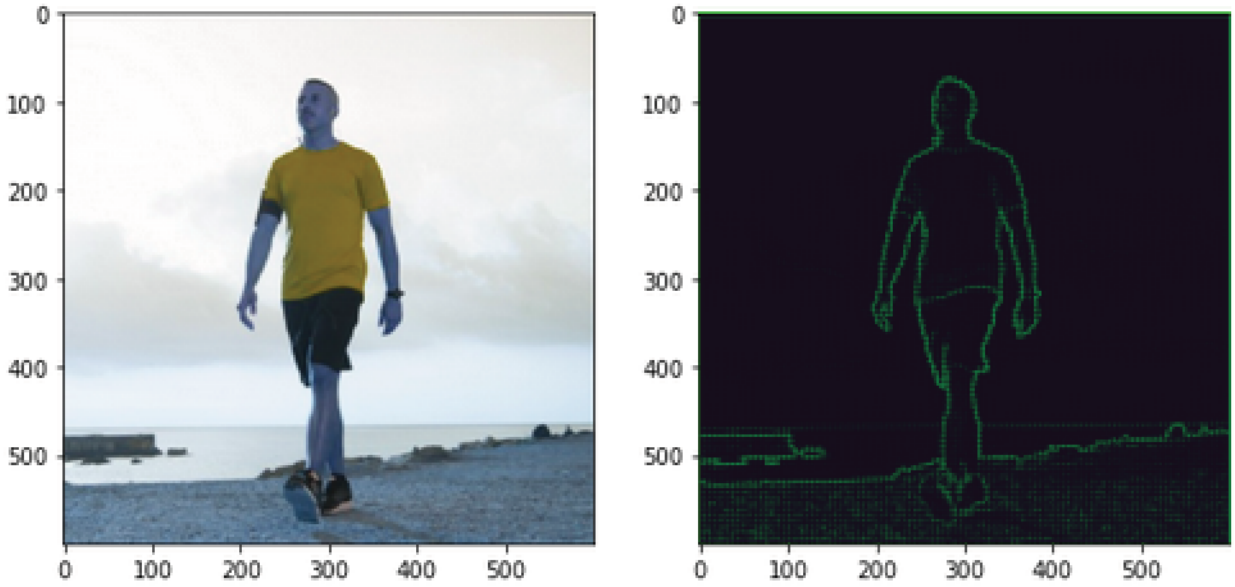
(Izvor: chrisjmccormick.files.wordpress.com/2013/05/magnitudes.png)

Kako bismo ovo izbjegli, grupiramo skupine susjednih ćelija, uzimamo srednju vrijednost iznosa i smjera vektora, te normaliziramo dobiveni vektor dijeleći ga njegovom apsolutnom vrijednošću, tj. iznosom intenziteta.

$$v = \frac{v1 + v2 + v3 + v4}{|v1 + v2 + v3 + v4|} \qquad (2-1)$$

2.1.5 Konačni vektor opisnih značajki

Kako bismo dobili konačni vektor moramo zbrojiti sve vektore dobivene blokovskom normalizacijom. Vizualizacija vektora vidljiva je na slici 2.4.

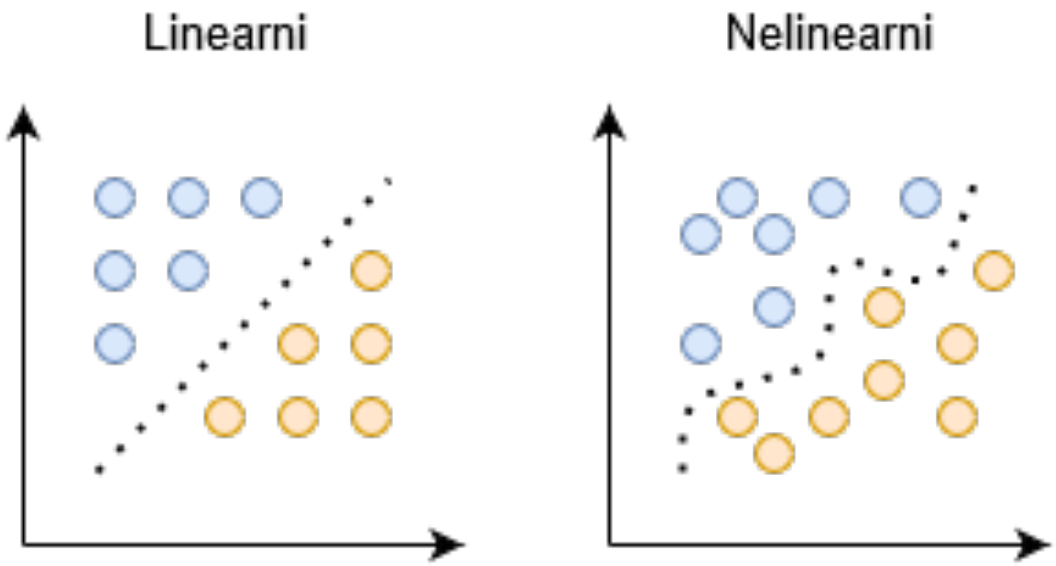


Slika 2.5. Originalna slika i vizualizacija vektora opisnih značajki

2.2. Klasifikacija

Klasifikacija u kontekstu strojnog učenja je proces dodjeljivanja klasne oznake određenim značajkama. Konkretan algoritam koji obavlja taj proces nazivamo klasifikator, a zadatak mu je inducirati hipotezu (funkciju) na temelju skupa podataka za učenje.

Ukoliko vizualiziramo opisane parove ulaz – izlaz u kartezijevom koordinatnom sustavu možemo si jasnije predočiti problem klasifikacije. Cilj klasifikatora je pronaći funkciju koja predstavlja granicu među klasama. Analogno odabranoj funkciji koja zadovoljava problem, klasifikatore možemo podijeliti u linearne i nelinearne



Slika 2.6. Linearni (lijevo) i nelinearni klasifikator (desno)

Osim po obliku grafa funkcije, klasifikatore možemo podijeliti na binarne i višeklasne.

Klasifikator koji klasificira podatak u jednu od dvaju klasa nazivamo binarni klasifikator. Učenje binarnog klasifikatora analogno je učenje booleaove funkcije:

Neka je hipoteza $h: X \rightarrow [0, 1]$

$$h(x) = 1, x \in C$$

$$h(x) = 0, x \notin C \qquad (2-2)$$

Kažemo da primjer zadovoljava hipotezu ukoliko vrijedi:

$$x \in X: h(x) = 1 \qquad (2-3)$$

Kažemo da je hipoteza konzistentna s obzirom na podatak za učenje *(x, y)* ukoliko vrijedi *h(x) = y*.

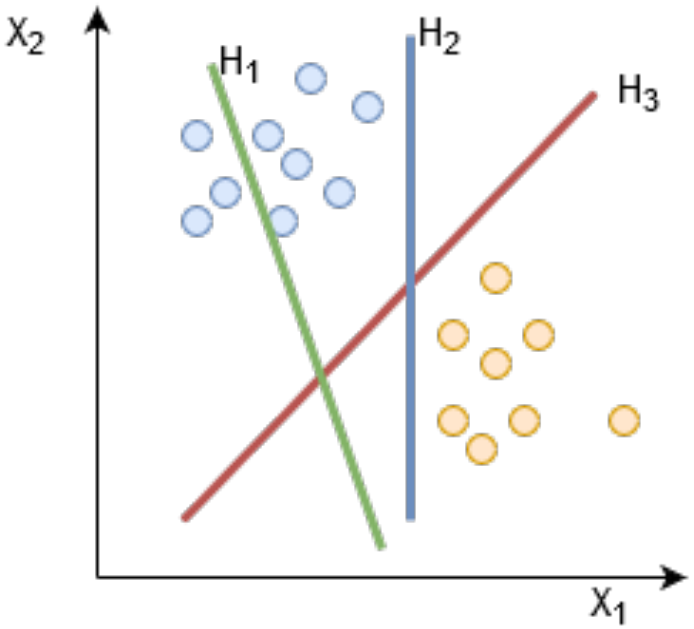
Analogno tome, višeklasni klasifikatori dodjeljuje podatku neku od klasa iz skupa potencijalnih klasa *Cj*, gdje *j = 1, …, k*

2.2.1. Metoda potpornih vektora

Iskustvo je pokazalo kako stroj potpornih vektora dobro funkcionira u kombinaciji sa metodom histograma usmjerenih gradijenata, te ćemo njega kao klasifikator.

Stroj potpornih vektora pripada skupini diskriminativnih modela koji, kao što je definirano u [6], izravno modeliraju granicu između klasa na kartezijevoj ravnini, za razliku od generativnih modela koji istu granicu modeliraju preko zajedničke gustoće vjerojatnosti. Korištenjem diskriminativnih modela dolazimo do problema separabilnosti – između dobivenih uređenih parova u ravnini možemo povući beskonačan broj linija ili ploha koje adekvatno razdvajaju dane parove u odgovarajuću klasu.

Stroj potpornih vektora rješava taj problem korištenjem teorema, tj. kriterija maksimalne margine hiperravnine. Laički gledano, prema kriteriju maksimalne margine odabire se ona hiperravnina koja je najviše moguće udaljena i od pozitivnih i od negativnih točaka koje predstavljaju uređene parove iz seta za učenje (slika 2.6).



Slika 2.7. Vizualizacija hiperravnina i točke klasa

Kod stroja potpornih vektora odabrana hiperravnina može se prikazati kao kombinacija vektora odabranih iz skupa za učenje, tzv. potpornih vektora.

Prema [6] linearni model definiramo na sljedeći način:

$$h(x) = w^T * \phi(x) + \omega_0 \quad (2-4)$$

gdje je w^Tvektor težina, $\omega_0$pomak, x je novi ulazni podatak, a $\phi(x)$ funkcija koja x preslikava u prostor značajki.

Pretpostavit ćemo da su oznake, tj. klase primjera za učenje $y \in [-1, +1]$. Klasa kojoj pripada x odgovara predznaku funkcije $h(x)$, tj. $x = sgn(h(x))$. Ako pretpostavimo da su svi primjeri linearno razdvojivi, tada vrijede sljedeće pretpostavke:

$$h(x) \geq 0, y_i = +1$$
$$h(x) < 0, y_i = -1 \quad (2-5)$$

U tom slučaju udaljenost između primjera x i hiperravnine $h(x) = 0$ jest:

$$d = \frac{h(x)}{\|w\|}, \quad (2-6)$$

Zanimaju nas jedino rješenja kod kojih su svi primjeri ispravno klasificirani, te stoga udaljenost možemo prikazati, prema [6] kao:

$$\frac{y^{(i)} h\left(x^{(i)}\right)}{\|w\|} = \frac{y^{(i)}(w^T \phi(x^{(i)} + \omega_0)}{\|w\|} \quad (2-7)$$

Iz čega je evidentno da će udaljenost biti nenegativna jer vrijedi

$$y^{(i)} h\left(x^{(i)}\right) \geq 0 \quad (2-8)$$

Marginu definiramo kao najkraću udaljenost od hipperavnine do najbližeg primjera iz seta učenja.

$$\frac{1}{\|w\|} \min_i \left\{ y^{(i)}(w^T \phi(x^{(i)} + \omega_0) \right\} \quad (2-9)$$

S obzirom da takvih margina postoji beskonačno, nas zanima samo slučaj kada će margina biti maksimalna, u odnosu na parametre *w* i w_0:

$$\max_{w,w_0}\left\{\frac{1}{\|w\|}\min_i\{y^{(i)}(w^T\phi(x^{(i)}+\ \omega_0)\}\right\} \qquad (2-10)$$

Budući da su svi primjeri linearno razdvojivi, ovaj izraz bi nam trebao dati samo 1 rješenje
Iz navedenog proizlazi kako za $x^{(i)}$ koji je najbliži margini vrijedi:

$$y^{(i)}(w^T\phi\left(x^{(i)}+\ \omega_0\right)=1 \qquad (2-11)$$

Prema [6], maksimalizirana margina imat će barem 2 aktivna ograničenja – jedno za najbliži primjer jedne klase, jedno za najbliži primjer druge klase. Širina maksimalne margine dana je, prema [6] kao:

$$\underset{w,w_0}{\text{argmax}}\frac{1}{\|w\|} \qquad (2-12)$$

tj.:

$$\underset{w,w_0}{\text{argmin}}\frac{1}{2}\|w\|^2 \qquad (2-13)$$

uz:

$$\min\|w\|=\ \min\|w\|^2 \qquad (2-14)$$

2.2.2. Višeklasna klasifikacija

Prema zadatku navedenom u ovom radu, potrebno je implementirati klasifikaciju za proizvoljan broj objekata na cesti. U svom osnovnom obliku stroj potpornih vektora vrši binarnu klasifikaciju – ulazni podatak je označen ili kao da pripada ili kao da ne pripada danoj klasi. Postoje dva ustaljena načina koja možemo iskoristiti kako bismo proširili stroj potpornih vektora na višeklasnu klasifikaciju:

1. Jedan naspram ostali (engl. *one-vs-rest*) – svedemo problem na K-1 binarnih klasifikatora tako da svaki binarni klasifikator razdvoji hiperravninom primjer tražene klase od primjera ostalih klasa. [4]
2. Jedan naspram jedan (engl. *one-vs-one*) – problem svedemo na K povrh 2 binarnih klsaifikatora, jedan za svaki par klasa [4]

U ovom radu ćemo koristit prvi pristup, pošto je jednostavan za jednostavan za kasnije proširivanje, te njime izbjegavamo sve probleme višeznačne klasifikacije, pošto svaki rezultat mora dati i stupanj pouzdanosti klasifikacije.

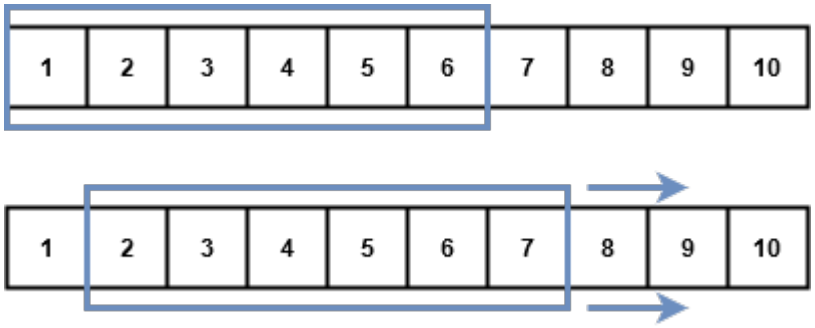
Stroj potpornih vektora u svom osnovnom obliku ne daje pouzdanost pojedine klasifikacije, nego samo udaljenost točke za ulazni primjer od hiperravnine. Kako bismo dobili pouzdanost, moramo primijeniti Plattovu kalibraciju, tehnika koju je predložio John Platt 1999. godine [11].

2.3. Lokalizacija

Proces detekcije objekata sastoji se od klasifikacije i lokalizacije objekta. Na danom originalnoj slici moramo moći izdvojiti objekt te izdvojeni objekt poslati modelu na klasifikaciju. U ovom radu implementirat ćemo tehniku klizećeg prozora.

2.3.1. Tehnika klizećeg prozora

Tehnika klizećeg prozora je, u općenitom smislu, tehnika u kojoj veći ulazni podatak dijelimo na manje dijelove određene veličine te ih sekvencijalno obrađujemo.



Slika 2.8. Tehnika klizećeg prozora

Ova tehnika ima široku primjenu na razne probleme koji zahtijevaju operacije nad nizovima, jer često omogućuju rješenje istog problema u O(n) umjesto u $O(n^3)$. U našem problemu je algoritam koristan, jer možemo izdvojiti određeni dio scene (dio koji pokriva cestu), te ga podijeliti na prozore određene veličine. Ovisno o percipiranoj udaljenosti od kamere, scena bi bila podijeljena na manje ili veće prozore. Dio slike koji pokriva prozor zatim šaljemo na izdvajanje opisnih značajki i, naposljetku, u klasifikator.

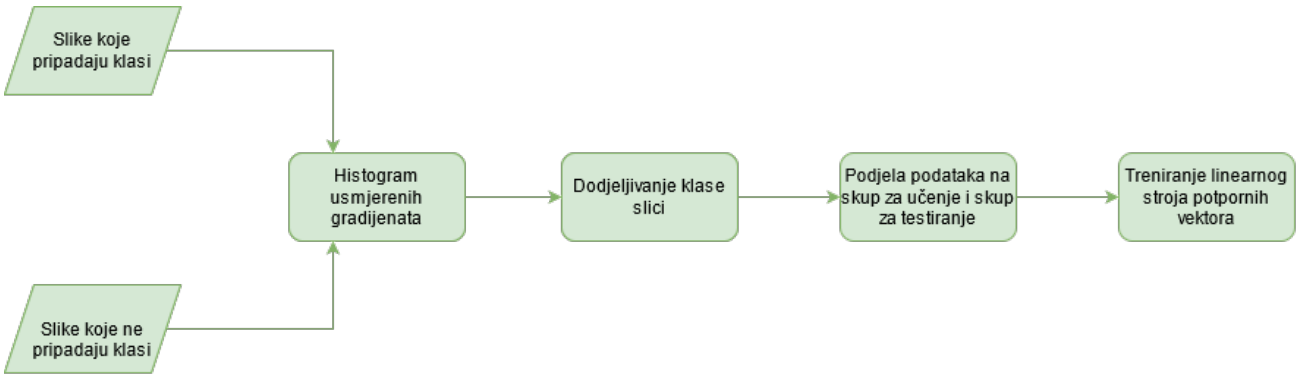
3. PREGLED PROGRAMSKE IZVEDBE RJEŠENJA

Na temelju teorijskih načela opisanih u prethodnom poglavlju, u ovom poglavlju ćemo implementirati programsko rješenje. Kao glavni programski jezik odabran je Python, zbog svoje široke primjene u polju računalnog vida i strojnog učenja, te velike zastupljenosti u akademskim krugovima. Programsko okruženje koje ćemo koristit je Jupyter Notebook – interaktivno web okruženje za programski jezik Python koje nam omogućava lakše eksperimentiranje i prezentaciju rezultata pojedinih odsječaka koda. Za implementaciju algoritama specifičnih za računalni vid, koristit ćemo biblioteku Open Source Computer Vision Library – OpenCV [12]. Biblioteka sadrži brojne pomoćne metode i implementacije koje pomažu pri manipulaciji digitalnih fotografija, izdvajanju opisnih značajki te neke gotove modele za detekciju objekata. Također ćemo koristit biblioteku scikit-learn [13] za Python, koja sadrži algoritme za strojno učenje, uključujući i implementaciju stroja potpornih vektora. Za matematičke operacije koristit ćemo biblioteku NumPy [14], koja dolazi s implementacijama za matematičke operacije višeg stupnja apstrakcije – operacije nad nizovima i matricama.

Model za detekciju pješaka koji će biti korišten u ovom radu je model dostupan u OpenCV biblioteci, te on implementira istu metodu opisanu u ovom radu. Model za detekciju automobila ćemo implementirati u potpunosti. Ista metoda se može koristiti za proizvoljan broj objekata koje želimo detektirati – autobus, bicikl, motocikl, itd.

3.1. Dijagram toka

Program možemo podijeliti u dva logička toka. Prvi je treniranje i testiranje modela na prikladnom setu fotografija (Slika 3.1). Drugi dio programskog rješenja obuhvaća učitavanje video zapisa i podjelu na pojedinačne okvire. Svaki okvir se zatim provodi kroz proces lokalizacije uz tehniku klizećeg prozora, a iz svakog pojedinog prozora izdvajamo svojstvene značajke i učitavamo ih u linearni klasifikator. Ukoliko linearni klasifikator odluči da odabrani prozor sadrži sliku klase koju tražimo, tako ga označava i dodaje u skup označenih prozora tog okvira. Primjenom NMS algoritma pronalazimo presjek tog skupa koji predstavlja konačan označeni objekt.



Slika 3.1. dijagram toka procesa učenja modela

3.2. Izvor podataka za učenje i testiranje

Podatci korišteni za učenje modela prepoznavanja automobila dolaze iz izvora: baza slika vozila sa sveučilišta u Madridu, Grupo de Tratamiento de Imagenes (GIT) [15]. Baza slika sadrži 3425 fotografija automobila uslikanih sa stražnje strane, iz različitih perspektiva, te 3900 fotografija koje ne sadrže automobile. Fotografije su uslikane i odrezane na takav način da pruže veći varijabilitet ulaznih podataka za učenje, a time i robustnije rezultate testiranja. Fotografije su rezolucije 64x64, izrezane iz videozapisa rezolucije 360x256 piksela snimljenih u Bruxellesu, Madridu i Torinu.

Video zapisi korišteni za testiranje dolaze iz skupa podataka sa Karlsruhe instituta za tehnologiju (KIT) [16]. Dostupni su video zapisi s različitih lokacije, različitih vremenskih uvjeta i prometnih uvjeta što nam daje dobre mogućnosti za daljnje testiranje.

3.3. Izdvajanje opisnih značajki

Opisne značajke biti će izvučene korištenjem metode histogram usmjerenih gradijenata, opisan u prethodnim poglavljima. Biblioteka skimage sadrži pomoćne funkcije za obradu fotografija u python programskom jeziku. Iskoristit ćemo implementaciju funkcije hog koja kao rezultat vraća vektor koji sadrži opisne značajke fotografije, te vraća vizualizaciju dobivenog histograma usmjerenih gradijenata.

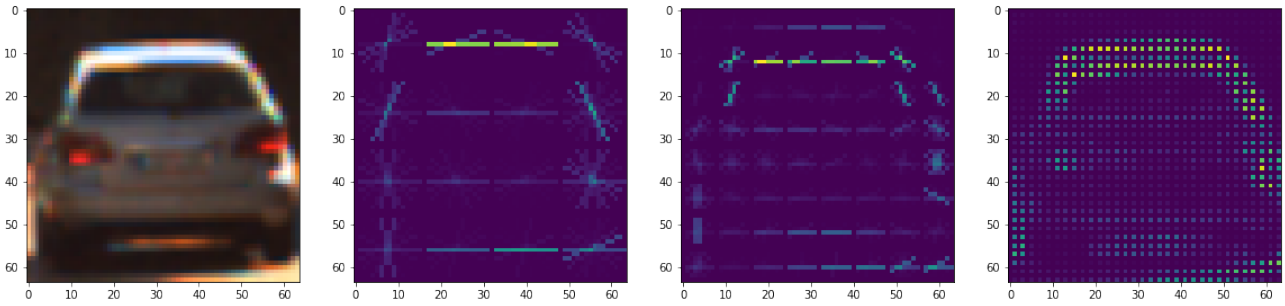
```
import matplotlib.pyplot as plt
from skimage.feature import hog


hog_features, hog_image = hog(image, orientations=9,
                              pixels_per_cell=(8, 8),
                              cells_per_block=(2, 2),
                              visualize=True, feature_vector=feature_vector_flag)

f, axes= plt.subplots(1,4,figsize=(20,10))
axes[0].imshow(vehicle_images_original[random_vehicle])
```

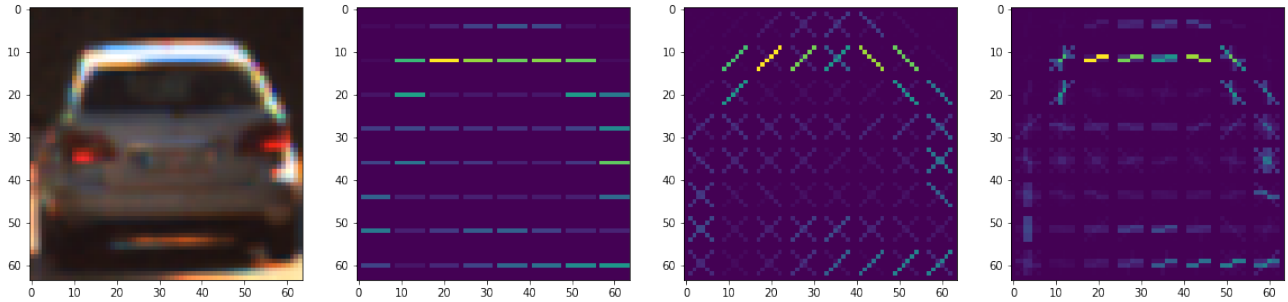
Slika 3.5. Programski kod za izdvajanje HOG opisnih značajki

Na slici 3.6 jasno možemo vidjeti kako sa smanjenjem broja piksela po ćeliji, histogram usmjerenih gradijenata poprima karakterističan oblik stražnjeg kraja automobila, dok pri većim vrijednostima tog parametra histogram izgleda apstraktnije i neodređenije. Analogno tome, ukoliko povećavamo broj orijentacija možemo vidjeti kako usmjereni gradijenti bolje prate rubove automobila (slika 3.7). U oba slučaja možemo vidjeti kako je intenzitet gradijenta jači kod jasnije određenih rubova, koji su u većem kontrastu u odnosu na pozadinu.



a) b) c) d)

Slika 3.6. Izvorna sliku automobila (a), te vizualizaciju HOG-a s 16 piksela po ćeliji (b), 8 piksela po ćeliji (c) i 2 piksela po ćeliji (d)



a) b) c) d)

Slika 3.7 Izvorna slika automobila (a), te vizualizacija HOG-a uz broj orijentacija = 1 (b), broj orijentacija = 2 (c), te broj orijentacija = 6 (d)

S obzirom na odabrane parametre funkcija hog nam vraća vektor opisnih značajki različitih dimenzija. Veličina vektora je obrnuto proporcionalna broju piksela po ćeliji i broju ćelija po bloku, a proporcionalna broju orijentacija. S obzirom na to da veličina vektora može drastično utjecati na performanse, potrebno je naći optimalnu kombinaciju veličine vektora i brzine izvođenja.

3.4. Učenje modela

Model koji se pokazao dobar u kombinaciji sa HOG metodom je linearni stroj potpornih vektora. Trenirati ćemo ga na GIT skupu podataka, koristeći 80% fotografija za treniranje, a 20% za inicijalno testiranje preciznosti modela.

```
# podjela podataka na skup za ucenje i skup za testiranje

from sklearn.model_selection import train_test_split

X_ucenje,  X_testiranje, Y_ucenje, Y_testiranje = train_test_split(
    [svojstva_automobila, svojstva_ostala],
    lista_oznaka,
    posto_test=0.2,
    shuffle=True)
```

Slika 3.7. Podjela podataka na skup za učenje i skup za testiranje modela

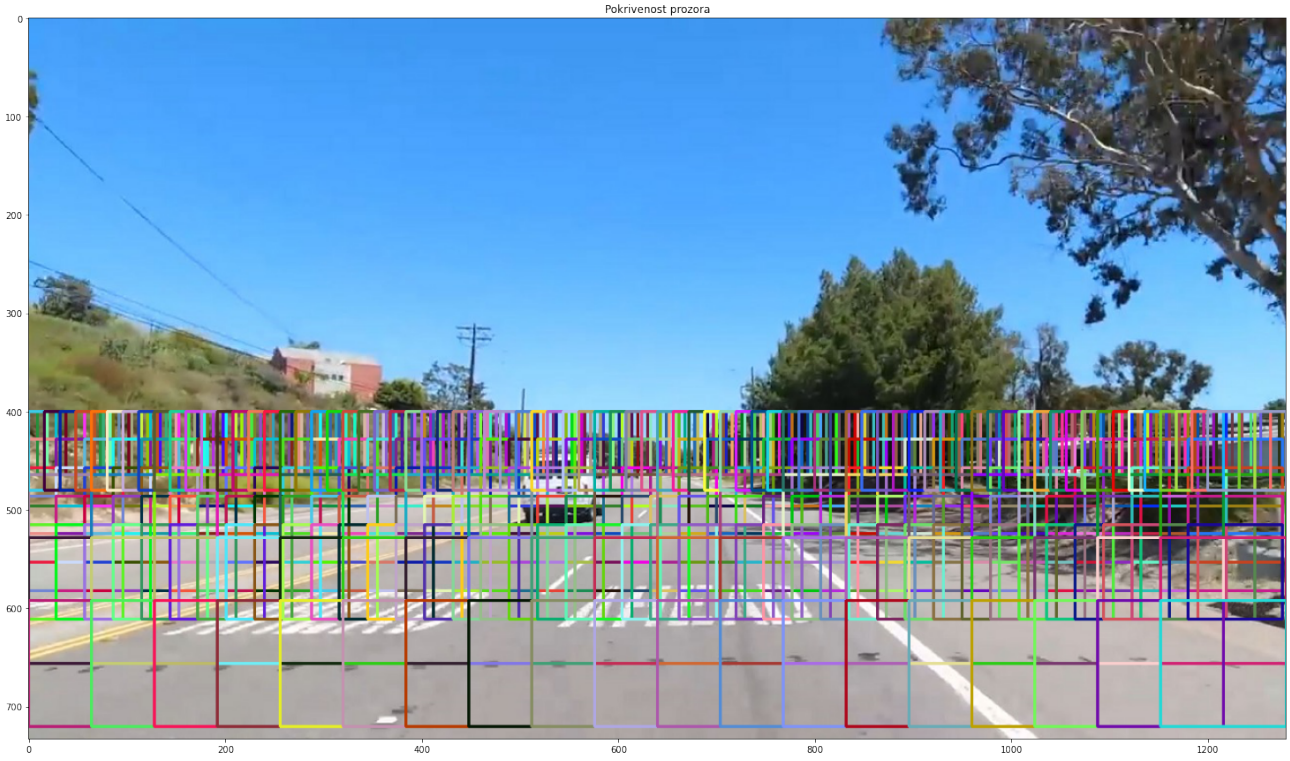
Konačno, instanciramo linearni SVM klasifikator, te mu predajemo podatke za učenje i testiranje (Sl. 3.8.).

```
from sklearn.svm import LinearSVC
klasifikator= LinearSVC()
klasifikator.fit(X_ucenje,Y_ucenje)
klasifikator.score(X_testiranje,Y_testiranje)
```

Slika 3.8. Kod za učenje i testiranje

3.5. Lokalizacija objekta

Kako bismo uspješno klasificirali objekt prvo ga moramo lokalizirati. Za ovaj korak koristit ćemo već opisanu tehniku klizećeg prozora. Scenu dobivenu iz kamere automobila podijelit ćemo u četiri sektora, od kojih svaki sadrži prozore različitih dimenzija i različite stope međusobnog preklapanja. Sektori bliže kameri će imati veće prozore s višom stopom preklapanja, a sektori dalji od kamere će imati manje prozore s manjom stopom preklapanja – analogno očekivanoj percepciji prepreka na cesti.



Slika 3.9. Tehnika klizećeg prozora primijenjena na područje ceste ispred automobila

Za svaki okvir videozapisa uzet ćemo sadržaj fotografije u svakom pojedinom prozoru te ga poslati višeklasnom multiklasifikatoru kako bi odredio pripada li sadržaj prozora nekoj od klasa. Prozori koji prođu kroz klasifikator i budu klasificirani kao automobil zatim bivaju tako i označeni na izlaznom videu.

3.6. Implementacija višeklasne klasifikacije

Linearan SVM je po prirodi binarni klasifikator te stoga moramo proširi njegovu funkcionalnost na klasifikacije više klasa. To ćemo učiniti koristeći metodu „Jedan naprama ostali“ opisanu u prethodnom poglavlju.

Višeklasni klasifikator će biti izveden kao zasebna klasa koja u sebi sadrži reference na sve instance klasifikatora koje naš program ima na raspolaganju (u ovom slučaju automobil i pješak), izvršiti klasifikaciju za ulazni lokalizirani objekt, provesti Plattovu kalibraciju kako bismo dobili pouzdanost klasifikacije, te zatim donijeti odluku pripada li lokalizirani objekt ijednoj od klasa.

3.7. Implementacija metode potiskivanja ne-maksimuma

Ukoliko naš algoritam za lokalizaciju i klasifikaciju radi očekivano u kombinaciji sa tehnikom klizećeg prozora, doći će do određnih preklapanja prozora označenih da pripadaju nekoj od klasa. Kako bismo preciznije odredili stvarni oblik objekta, te kako bi izlaz bio pregledniji, primijenit ćemo metodu potiskivanja ne-maksimuma (engl. *non-maximum suppression algorithm*) (Sl. 3.10.).

a) b)

Slika 3.10. NMS algoritam: a) prije primjene, b) nakon primjene

4. ANALIZA REZULTATA PROGRAMSKOG RJEŠENJA

U ovom poglavlju ćemo testirati implementaciju programskog rješenja. Za odabrani videozapis ručno brojimo tražene objekte, a zatim isti video zapis predajemo programu te uspoređujemo rješenja. Treba obratiti pažnju na lažno pozitivne i lažno negativne rezultate, te na postotak uspješnog prepoznavanja objekata u ovisnosti o ulaznim parametrima programa. Testiranje programa se provodi na videozapisima, ili pojedinačnim okvirima izvučenih iz istih, rezolucije 1280x720 piksela. Svi pokusi se izvode na računalu sljedećih specifikacija: Procesor Intel i5-4460, 8GB RAM DDR3, Windows 10 operacijski sustav.

4.1. Postavljanje parametara

Prvi parametri koje je potrebno definirati su parametri HOG funkcije. Kako bismo to odabrali provodimo niz testova, rezultati kojih su dostupni u tablici 4.1. Na temelju tih rezultata odabiremo sljedeće parametre: broj orijentacija = 9, broj ćelija po bloku = 2, broj piksela po bloku = 16. metoda potiskivanja ne-maksimuma

Tablica 4.1. Rezultati iterativnog testa treniranja linearnog SVM-a

Broj izvođenja	Broj orijentacija	Broj ćelija po bloku	Broj piksela po bloku	Veličina vektora	Vrijeme treniranja (sekunde)	Preciznost modela (%)
1	2	2	16	72	33.39	86.29
2	2	1	8	128	113.45	81.95
3	9	1	8	576	124.01	97.86
4	9	2	16	324	41.39	98.98
5	9	2	8	1764	111.94	98.85
6	12	2	8	2352	125.81	98.9
7	12	2	4	10800	N/A	N/A

Sljedeći parametar koji moramo odrediti je pokrivenost slike prozorima za detekciju objekata. Prozori će pokrivati oko 60% visine slike i 100% širine slike, te će biti podijeljeni u 4 različite kategorije. Točne specifikacije prozora dane su u tablici 4.2.

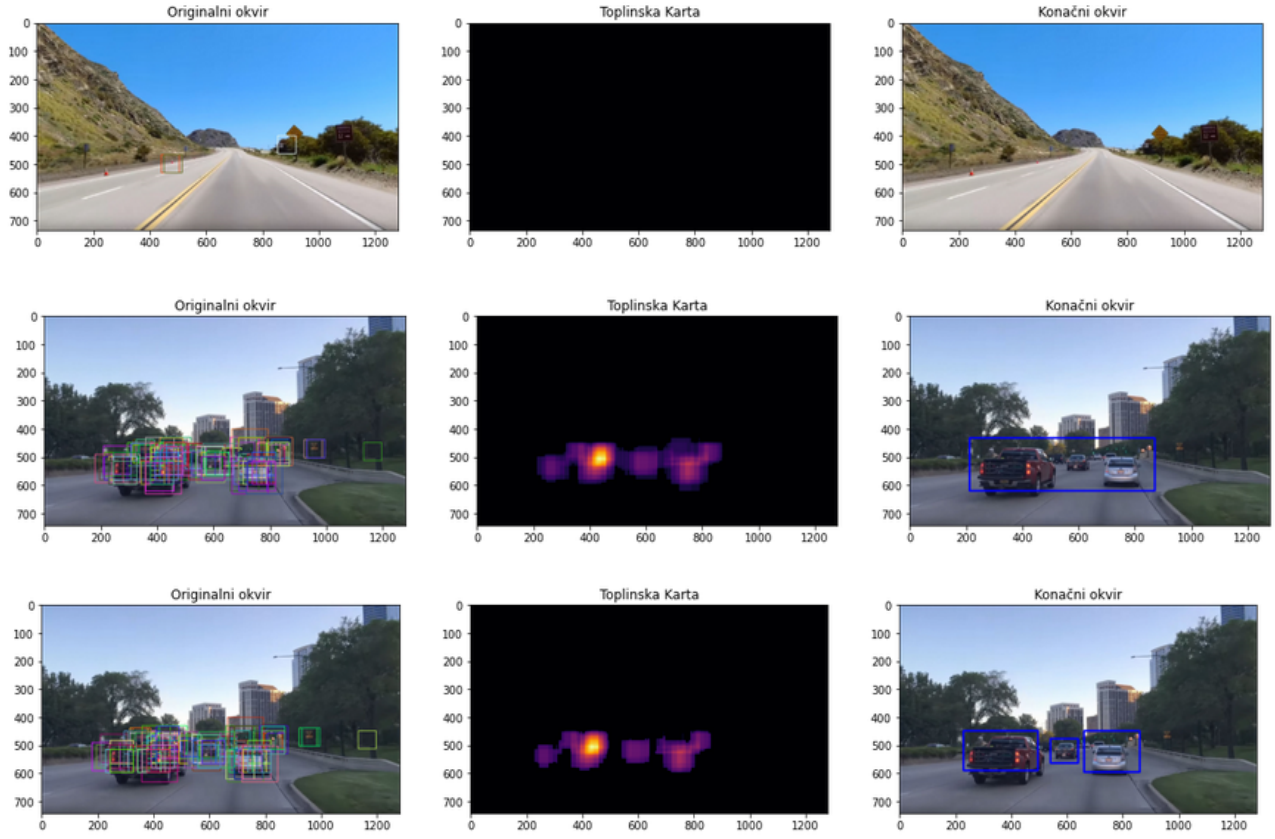
Tablica 4.2. Specifikacije prozora za detekciju

Veličina prozora	Stopa preklapanja [%]	Raspon Y koordinate [px]
64x64	84	[410, 550]
82x82	80	[410, 600]
100x100	70	[410, 690]
128x128	60	[410, 710]
	Ukupan broj prozora:	**2107**

Prepoznavanje automobila testiramo na slikama izdvojenima iz videozapisa kamere automobila. Slike su dimenzija 1280x720 piksela. Prva slika je kontrolna, bez automobila na cesti, a na drugoj slici se nalaze 3 automobila neposredno ispred kamere. Na Sl. 4.1. možemo primijetiti 3 lažno pozitivna prozora, a na drugoj slici 430 prozora s točno detektiranim automobilom i 3 prozora s lažno detektiranim automobilom. To nam daje stopu lažno pozitivnih rezultata od 0.69%.

Slika 4.1 Lokalizirani prozori s detektiranim automobilom.

Kako bismo smanjili stopu lažno pozitivnih rezultata i preciznije odredili oblik objekata pred automobilom primjenjujemo metoda potiskivanja ne-maksimuma. Uz primjenu stope preklapanja 3, na slici 4.2 gore i u sredini, vidimo kako smo uspješno riješili problem lažno pozitivnih rezultata, no sva 3 automobila spojena su u jedan detektirani objekt. Na slici 4.2. dolje vidimo da smo uz primjenu stope preklapanja 5 uspjeli detektirati 3 različita automobila.



Slika 4.2. Metoda potiskivanja ne-maksimuma

Zadnji parametar koji moramo odrediti je dužina povijesti prozora koji sadrže detektirane objekte. Ovo radimo kako bismo stabilizirali okvir detektiranog objekta kroz vrijeme. Ukoliko izostavimo ovaj korak, kratkoročno može postojati velika razlika u tehničkim parametrima fotografije koji utječu na detekciju objekta, te možemo propustiti već prije detektirani objekt. Praćenjem detektiranih objekata možemo izgladiti te razlike i osigurati veću razinu konzistentnosti među pojedinim okvirima videozapisa.

Testiranje provodimo na video zapisu od 269 okvira, u rezoluciji 1280x720 pixela. Odraditi ćemo nekoliko iterativnih testova, s varijabilnom dužinom povijesti prozora s detektiranim objektima te usporediti rezultate. U prvom testu ne pratimo povijest prozora, u drugom testu pratimo prethodnih 10 prozora, a u trećem prethodnih 20. Pokazalo se da preciznost detekcije blago opada s povećanjem broja prozora koje pamtimo, no broj lažno pozitivnih rezultata se drastično smanjuje (tablica 4.3.).

Tablica 4.3. Usporedba učinkovitosti pamćenja prijašnjih detektiranih prozora

Broj zapamćenih prozora	**Točne detekcije**			**Lažno pozitivne detekcije**		
	Veličina uzorka	Detektirano	%	Veličina uzorka	Detektirano	%
20	250	236	94.4	250	11	4.4
10	250	241	96.4	250	34	13.6
0	250	245	98.0	250	67	26.8

Brzina obrade pojedinog okvira sa zadanim parametrima je 1.11 okvira po sekundi. Budući da je prosječna stopa videozapisa 25 okvira po sekundi, vidimo da je algoritam za jedan red veličine prespor kako bi mogao biti primijenjen u stvarnom svijetu.

4.2. Rezultati višeklasnog algoritma

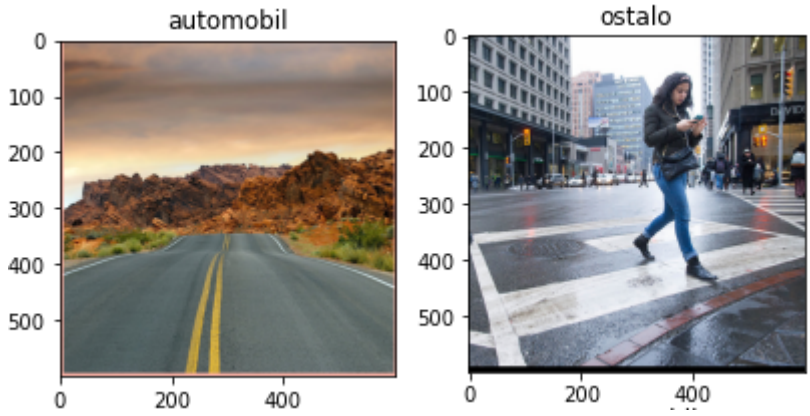
Testiranje kvalitete donošenja odluke višeklasnog klasifikatora provest ćemo na setu od 100 slika. Set se sastoji od 45% fotografija automobila, 45% fotografija pješaka, te 10% fotografija ostalih objekata. Višeklasni klasifikator učitava pojedinu sliku, obavlja detekciju pješaka i automobila, te donosi odluku o klasifikaciji na temelju dobivenih iznosa pouzdanosti.

Ukoliko je detektiran pješak i pouzdanost iznosi barem 0.5 slika je klasificirana kao pješak, a analogno vrijedi i za automobil. Ukoliko klasifikator detektira oba objekta, slika se klasificira kao onaj objekt koji ima veću rezinu pouzadnosti. Na uzorku od 100 slika prosječno vrijeme obrade slike je 0.1853 sekundi, tj. 5.4 slike po sekundi. Vidimo da je algoritam značajno brži nego kod analize videozapisa, gdje pamtimo prijašnje okvire, te ima potencijal za uporabu u stvarnom svijetu.

Tablica 4.4 Rezultati testiranja višeklasnog klasifikatora

	Veličina uzorka	Točno detektirani		Lažno pozitivni	
		broj - postotak		broj - postotak	
Automobili	45	44	97.78	3	6.82
Pješaci	45	42	93.3	0	0
Ostalo	10	7	70	4	40

U tablici 4.4 dani su rezultati testiranja. Svi osim jednog automobila su točno detektirani, no detektirana su i 3 lažno pozitivna automobila (Sl. 4.3. a), što ukupno daje pouzdanost u iznosu od 90.96%. Pješaci nemaju lažno pozitivnih detekcija, no broj točno detektiranih je nešto niži (Sl. 4.3. b). Slike klasificirane kao 'ostalo' imaju najnižu razinu uspješne klasifikacije, što potencijalno ukazuje na potrebu podizanja praga pouzdanosti klasifikatora za automobile i pješake. Također, uzorak ostalih slika je 9 puta manji od kombiniranog uzorka automobila i pješaka, što rezultira visokim postotkom lažno pozitivnih detekcija.



a) b)

Slika 4.3. Primjeri pogrešno klasificiranih slika

5. ZAKLJUČAK

Razvoj autonomnih sustava vožnje sve je aktualnije područje interesa kako akademskih istraživanja, tako i industrije. Detekcija i klasifikacija prepreka na cesti jedno je od najbitnijih aspekata koji omogućuju implementaciju takvih sustava. U današnje vrijeme vozila su opremljena mnogim alatima koje možemo iskoristiti u tu svrhu, ponajprije kamerama na automobilu. U ovom radu pokazali smo kako možemo učitati videozapise s automobilske kamere, obraditi ih okvir po okvir, te na temelju dobivenih podataka pomoći vozilu da napravi odgovarajuću radnju.

Predložena metoda za detekciju i klasifikaciju prepreka je metoda potpornih vektora, koja spada u skupinu metoda nadziranog strojnog učenja. Kako bismo implementirali ovu metodu, potrebno je sliku dobivenu iz video kamere podijeliti na sektore u kojima očekujemo potencijalne prepreke, izdvojiti vektore opisnih značajki iz svakog sektora, a zatim te vektore provesti kroz klasifikator kako bismo mogli donijeti zaključak postoji li prepreka ispred automobila i, ako da, pod koju klasu spada.

Za izdvajanje značajki korištena je metoda histograma orijentiranih gradijenata. Prednost ove metode je što se ne obrađuje slika u cjelini, nego se zasebno obrađuju lokalizirani dijelovi slike, što znači da određena geometrijska i fotometrijska izobličenja imaju manji utjecaj na konačni rezultat.

Za implementaciju klasifikacije koristili smo linearni potporni vektor, koji je po prirodni binarni klasifikator, što znači da smo njegovu funkcionalnost morali proširiti kako bismo implementirali višeklasnu klasifikaciju. U ovom radu pokazali smo potpunu implementaciju linearnog potpornog vektora na primjeru automobila, a za detekciju pješaka iskoristili smo model implementiran u OpenCV biblioteci. Ovaj postupak može se proširiti na n klasa, implementiranih na opisan način. Višeklasni klasifikator zatim uspoređuje s kojom razinom pouzdanosti je određena prepreka klasificirana te donosi konačnu odluku.

Rezultati testiranja ove metode pokazali su da je uspješnost detekcije automobila i pješaka 95.55%, dok su slike koje ne sadrže prepreke klasificirane s nešto nižim postotkom točnosti. Za potrebe primjene ovakvog sustava taj omjer je adekvatan, pošto je bolje lažno pozitivno detektirati prepreku nego propustiti prepreku. Performanse dobivene provođenjem testova ovise o nekoliko faktora, ponajprije o broju sektora na koje dijelimo svaku pojedinu sliku, te koliko slika unazad pamtimo

rezultate. Testiranja provedena na navedenom računalu i s navedenim parametrima su ispod optimalne brzine obrade, no s napretkom tehnologije ili korištenjem računala s boljim komponentama moguće je ostvariti performanse adekvatne za stvarnu primjenu u automobilu.

Glavni problem ovakve implementacije višeklasne klasifikacije je činjenica da moramo zasebno implementirati model potpornog vektora za svaku traženu klasu, te što umanjujemo performanse sa svakom novom implementiranom klasom. Za *n* klasa moramo analizirati sliku *n* puta, te provesti *n* usporedbi razina pouzdanosti klasifikacije. Iz tog razloga nameće se zaključak da su za ovaj model adekvatnije neke od metoda dubokog učenja.

LITERATURA

[1] G. W. Humphreys, C. J. Price, and M. J. Riddoch, "From objects to names: A cognitive neuroscience approach," *Psychol. Res.*, vol. 62, no. 2, pp. 118–130, Jul. 1999, doi: 10.1007/s004260050046.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[3] T. M. Press, "Introduction to Machine Learning, Second Edition | The MIT Press." https://mitpress.mit.edu/books/introduction-machine-learning-second-edition (accessed Sep. 12, 2020).

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[5] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[6] J. Š. Bojana Dalbelo Bašić, *Strojno učenje*. Zagreb, 2014.

[7] I. J. Myung and M. A. Pitt, "Applying Occam's razor in modeling cognition: A Bayesian approach," *Psychon. Bull. Rev.*, vol. 4, no. 1, pp. 79–95, Mar. 1997, doi: 10.3758/BF03210778.

[8] "Occam's Razor." https://www.geeksforgeeks.org/occams-razor/.

[9] R. K. McConnell, "Method of and apparatus for pattern recognition," US4567610A, Jan. 28, 1986.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Jun. 2005, vol. 1, pp. 886–893 vol. 1, doi: 10.1109/CVPR.2005.177.

[11] J. C. Platt, "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," in *Advances in Large Margin Classifiers*, 1999, pp. 61–74.

[12] "OpenCV." https://opencv.org/.

[13] "scikit-learn: machine learning in Python — scikit-learn 0.23.2 documentation." https://scikit-learn.org/stable/ (accessed Sep. 13, 2020).

[14] "NumPy." https://numpy.org/ (accessed Sep. 13, 2020).

[15] Grupo de Tratamiento de Imágenes (GTI), “Vehicle Image Database.” https://www.gti.ssr.upm.es/data/Vehicle_database.html.

[16] J. Williams, “Vision meets Robotics: The {KITTI} Dataset | Autonomous Vision,” *Max Planck Institute for Intelligent Systems*. https://is.mpg.de (accessed Sep. 12, 2020).

SAŽETAK

Detekcija prepreka na cesti korištenjem kamere iz automobila

S obzirom na nagli rast područja razvoja programske potpore za autonomna vozila sposobnost detekcije i klasifikacije objekata u neposrednom okruženju automobila relevantnija je nego ikad. U ovom radu predstavljena je jedna od mogućih metoda koja rješava takav problem – metoda linearnih potpornih vektora. Kao prvi korak implementacije ove metode izdvojene su svojstvene značajke promatranih objekata pomoću histograma usmjerenih gradijenata. Na temelju dobivenih vektora istreniran je model za prepoznavanje automobila. Za potrebe implementacije višeklasne klasifikacije iskorišten je model za prepoznavanje pješaka iz biblioteke OpenCV, te su dobivene vrijednosti pouzdanosti detekcije uspoređene u svrhu donošenja konačne odluke. Potrebni parametri dobiveni su provođenjem iterativnih testova prilagođeni konkretnim primjerima. Konačni rezultat je uspješna detekcija i klasifikacija automobila i pješaka u većini slučajeva.

Ključne riječi: histogram usmjerenih gradijenata, metoda potpornih vektora, klasifikacija, lokalizacija, detekcija objekata

ABSTRACT

Obstacle Detecetion on the Road by Utilizing a Dashboard Camera

Due to the rapid growth of the software development area for autonomous vehicles, the ability to detect and classify objects in the immediate vicinity of a car is more relevant than ever. This paper presents one of the possible methods that solves such a problem - the method of linear support vectors. As the first step in the implementation of this method, the features of the observed objects were extracted using histograms of oriented gradients. Based on the obtained vectors, a car recognition model was trained. For the purposes of implementing the multi-class classification, a model for pedestrian recognition from the OpenCV library was used, and the final decision was made by comparing the obtained detection reliability values. The required parameters were obtained by conducting iterative tests adapted to specific examples. The end result is successful detection and classification of cars and pedestrians in most cases.

Keywords: histogram of oriented gradients, support vector machine, classification, localization, object detection

ŽIVOTOPIS

Filip Lukačević rođen je 21. srpnja 1993. godine u Osijeku. Osnovnu školu pohađao je u Višnjevcu, nakon koje upisuje 3. matematičku gimnaziju Osijek. Tijekom osnovnoškolskog obrazovanja sudjelovao je na natjecanjima iz matematike i geografije, a tijekom srednjoškolskog na natjecanjima iz engleskog jezika. Završetkom srednje škole upisuje Fakultet elektrotehnike i računarstva u Zagrebu, a 2016. godine prelazi na Fakultet elektrotehnike, računarstva i informacijskih tehnologija. 2018. godine odrađuje dvomjesečnu praksu u firmi UHP Digital u Osijeku, a nakon nje i sedmomjesečnu praksu u norveškoj firmi Rubrikk Group AS u Oslu, gdje i nastavlja raditi.