Prepoznavanje objekata i njihovih dijelova na složenim scenama

Varga, Hrvoje

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Automobilsko računarstvo i komunikacije

PREPOZNAVANJE OBJEKATA I NJIHOVIH DIJELOVA NA SLOŽENIM SCENAMA

Diplomski rad

Hrvoje Varga

Osijek, 2021

SADRŽAJ

1. UVOD

Jedan od osnovnih problema autonomne vožnje, robota namijenjenih za rad u kućanstvu i drugih je detekcija objekta u stvarnom vremenu s visokom točnošću i preciznošću. Detekcija objekta u stvarnom vremenu u takvim situacijama je nužna, radi donošenja prave odluke na vrijeme, radi sprječavanja prometnih nesreća kod autonomne vožnje, itd. Postoji velik broj radova na tu temu i problematika je s kojom se bavi velik broj ljudi. Razne metode i rješenja su razvijena, ali niti jedno u potpunosti nije primjenjivo. Neka od tih rješenja su R-CNN, fast R-CNN, faster R-CNN, YOLO i razna druga rješenja. Većina tih rješenja koristi neuronske mreže koje imitiraju rad ljudskog mozga, ali u računalnom obliku. Takvi programi treniraju, uče i donose odluke kako bi bili što precizniji i točniji, i što je najbitnije donose odluke u stvarnom vremenu. Sustavi s najboljim performansama su složena cjelina te kombiniraju više značajki slika niske razine s kontekstom visoke razine detektora objekata i klasifikatora scene. U ovom diplomskom radu govorit će se o trenutnim trendovima neuronskih mreža, također će se eksperimentalno testirati rad i funkcionalnost metode identifikacije objekata YOLO. Metoda identifikacije objekata YOLO će služiti kako bi vratila granični okvir pojedine slike koja se koristila u eksperimentalnom dijelu diplomskog rada. Nakon toga korištenjem koordinata graničnih okvira određena je klasa, te prikladna pozicija kojom bi robotska ruka mogla primiti određeni objekt. Dakle, osim same detekcije objekata, u ovom se radu razmatra i problem detekcije dijelova objekata relevantnih za neku radnu operaciju. Pod dijelom objekta relevantnim za operaciju hvatanja, u ovom se radu podrazumijeva određivanje mjesta na površini objekta prikladnog za hvatanje objekta. U prvom dijelu rada biti će objašnjene metodologije neuronskih mreža, bit će pojašnjena usporedba pojedinih metoda neuronskih mreža. Nakon toga slijedi eksperimentalni dio u kojem je detaljno opisana metoda i način koji se koristio u ovom diplomskom radu. Posljednje poglavlje je programska evaluacija u kojoj su prikazani rezultati te pojašnjene i pokazane usporedbe korištenih metodologija.

1.1. Zadatak diplomskog rada

Izraditi program koji prepoznaje objekte zadane klase na složenim scenama primjenom neuronske mreže. Program također treba prepoznati određeni dio objekta relevantan za obavljanje nekog zadatka primjenom poliedarskog modela klasa oblika. Razvijeni program eksperimentalno ispitati na RGB-D slikama složenih scena.

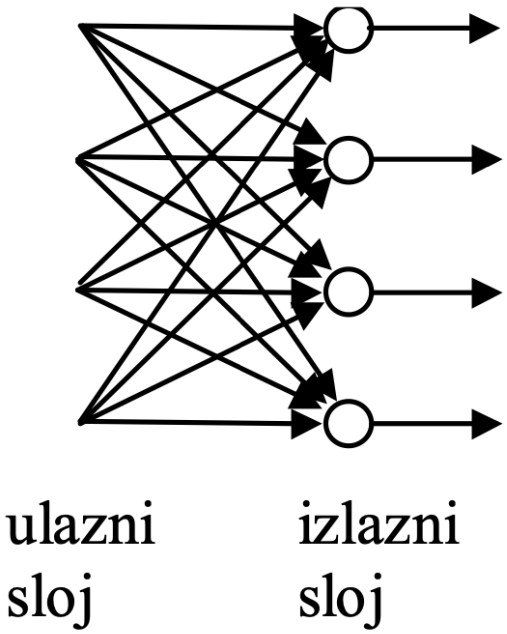
2. NEURONSKE MREŽE

Neuronske mreže su Alexander i Morton 1990. godine definirali kao masivno paralelni distribuirani procesor koji je dobar za pamćenje iskustvenog znanja. Sličnost između neuronskih mreža i mozga je u tome što se znanje stječe konstantnim učenjem i veze između neurona se koriste za pohranu znanja. Najvažnija svojstva neuronskih mreža su nelinearnost, adaptivnost, tolerancije na greške, itd. Najmanji dio neuronske mreže je neuron, te mu arhitektura mreže određuje način povezivanja. Iz tog razloga postoje četiri osnovne vrste neuronskih mreža, a to su jednoslojne mreže bez povratnih veza, višeslojne mreže bez povratnih veza, mreže s povratnim vezama i ljestvičaste mreže [1].

2.1. Vrste neuronskih mreža

2.1.1. Jednoslojne mreže bez povratnih veza

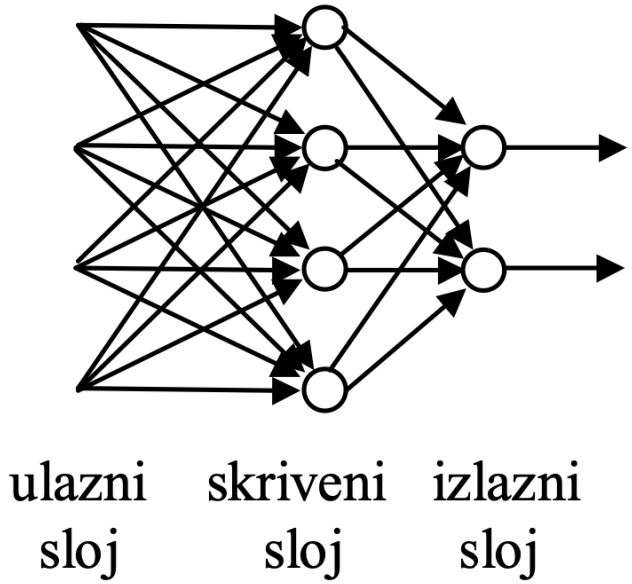
Jednoslojne mreže bez povratnih veza se sastoje od samo jednog sloja neurona. Ulazni sloj neurona se ne ubraja u slojeve jer se u njemu ne odvijaju nikakvi procesi obrade podataka. Jednoslojne mreže nemaju povratnu vezu. Prikaz takve mreže se može vidjeti na slici 2.1.



Slika 2.1 *Jednoslojna mreža bez povratne veze.*

2.1.2. Višeslojne mreže bez povrtnih veza

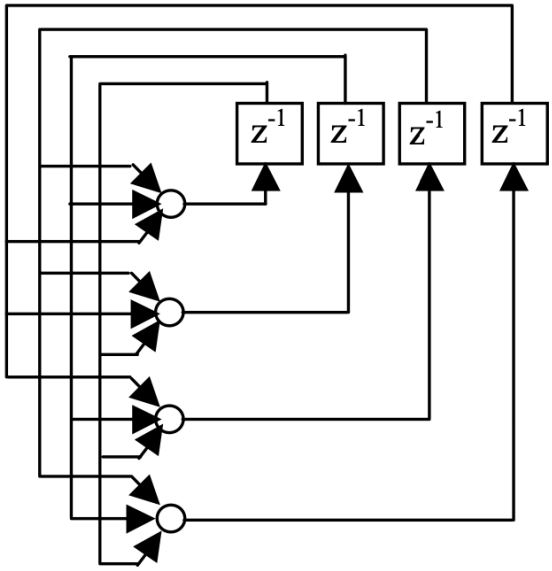
Višeslojne mreže se sastoje od ulaznog i izlaznog sloja, ali osim ta dva postoje i skriveni slojevi neurona. Kod višeslojnih mreža postoji pojam povezanosti mreže, te se definira tako da je mreža potpuno povezana ako je svaki neuron u svakom sloju povezan sa svim neuronima sljedećeg sloja. U slučaju da bilo koja veza nedostaje tada je mreža djelomično povezana. Na slici 2.2 se može vidjeti primjer višeslojne mreže koja ima jedan skriveni sloj i on se sastoji od četiri neurona. Broj ulaznih neurona u primjeru je isto četiri, ali oni ne moraju biti isti, dakle ulazni sloj i skriveni sloj mogu imati različit broj neurona, te izlazni sloj koji se sastoji od dva neurona.



Slika 2.2 *Višeslojna mreža s jednim skrivenim slojem.*

2.1.3. Mreže s povratnim vezama

Mreže s povratnom vezom moraju sadržavati barem jednu povratnu vezu, mogu sadržavati i skrivene slojeve, odnosno skrivene neurone. Ovakve mreže imaju dodatnu kvalitetu zato što sadrže povratnu vezu, ali time su i složenije pri analizi rada takvih mreža. Na slici 2.3 nalazi se mreža s povratnom vezom i bez skrivenih neurona. Svaki neuron dobiva za ulaz izlaz svih ostalih neurona osim samog sebe.



Slika 2.3 *Mreža s povratnom vezom.*

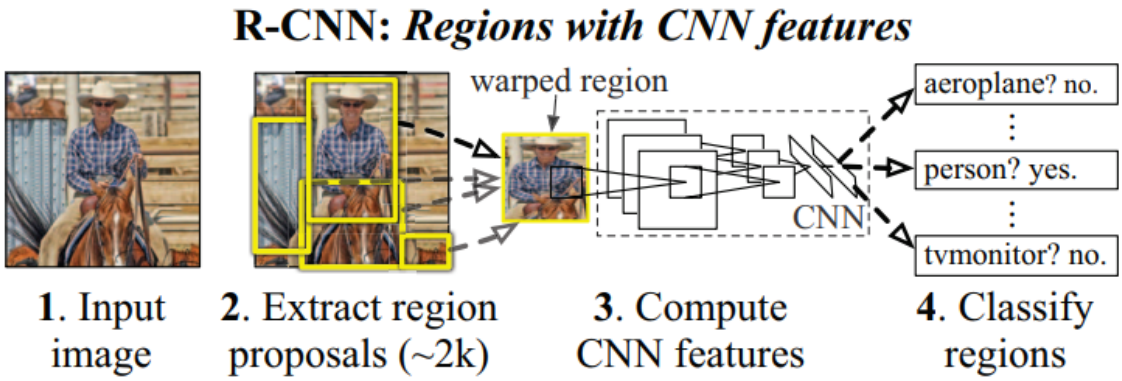
2.2.R-CNN

R-CNN u prijevodu konvolucijska neuronska mreža zasnovana na području nastala je zbog potrebe za jednostavnijim algoritmima neuronskih mreža, za prvobitnu primjenu u izdvajanju područja interesa. Prijašnja rješenja i način na koji se slika obrađivala nisu koristila nikakve algoritme koji bi prošli kroz sliku, te izdvojili područja na kojima postoji mogućnost da se nalazi objekt. Prijašnje mreže su odmah primjenjivale potpuno povezanu neuronsku mrežu na sliku te su kao izlaz dobili samo jednu riječ koja opisuje objekt na toj istoj slici. R-CNN je prva mreža koja, kao što je ranije spomenuto, koristi metodu odvajanja područja na kojima postoji mogućnost pojavljivanja objekta, te kao izlaz vraća granične okvire više objekata na slici, te daje vjerojatnost o kojem objektu se radi u određenom graničnom okviru [2].

2.2.1. Detekcija objekata korištenjem R-CNN

Sustav detekcije objekata korištenjem R-CNN-a sastoji se od tri modula. Prvi modul stvara prijedloge područja neovisno o kategoriji. Ovi prijedlozi definiraju skup kandidata koji su dostupni detektoru. Metoda koja se koristi kako bi predložila područja interesa je selektivna potraga (*engl. selective search*). Drugi modul je mreža koja se sastoji od 5 konvolucijskih slojeva, i 2 potpuno

povezana sloja. Rezultat prolaska slike kroz 5 konvolucijskih slojeva i 2 potpuno povezana sloja je vektor značajki fiksne veličine 4096 elemenata za svako predloženo područje dobiveno selektivnom pretragom. Da bi se navedena mreža mogla primijeniti na područja različitih veličina, područja se moraju skalirati na sliku veličine 227 x 227. Treći modul je skup linearnih SVM-ova specifičnih za pojedinu klasu objekta, koji na kraju određuju koji je objekt detektiran na pojedinom području interesa [2].



Slika 2.4 *Način rada R-CNN mreže*

Slika 2.4 slikovito prikazu način rada R-CNN mreže. Ukratko, na početnu sliku (*engl. Input image*) se primjenjuje algoritam selektivne potrage koji kao rezultat daje područja koje ne moraju biti pravokutnog oblika, zatim se prave granični okviri oko svakog takvog pronađenog područja kojih na kraju bude oko 2000. Svakom području pridružuje se vrijednost koja odražava procijenjenu pouzdanost da to područje predstavlja objekt. Nakon toga se primjenjuje operacija potiskivanja ne-maksimuma tako što se izračunava IoU (*engl. intersection-over-union*) između svaka dva područja te se izbacuju granični okviri koji zadovoljavaju uvjet da je njihov IoU s nekim područjem veće pouzdanosti ≥ 0.5. Tako se smanji broj područja interesa, zatim se svako područje prilagođava formatu slike 227 x 227. Nakon toga se primjenjuje 5 konvolucijskih slojeva i 2 potpuno povezana sloja koji na kraju daju vektor značajki za to područje. Na posljetku se primjenom SVM (*engl. Support-vector machine*) određuje kojoj klasi pripada detektirani objekt na tom interesnom području [2].

2.2.2. Trening

Prilikom treniranja mreže koristi se takozvani nadgledajući pred trening. Za pred trening konvolucijske neuronske mreže koristila se ILSVRC 2012 baza podataka, točnije baza slika. Ta baza slika sadrži par milijuna slika, koje je čovjek ručno detektirao, ali na slikama nije označen granični okvir već je jedna riječ, odnosno klasa, opis pojedine slike. Kako bi naučili R-CNN konvolucijsku neuronsku mrežu da detektira objekt na pojedinoj predloženoj regiji i napravi granični okvir svake regije, mrežu je bilo potrebno dodatno učiti. Za to dodatno učenje mreže korištena je PASCAL VOC baza podataka. PASCAL VOC baza podataka nije jednake veličine kao i ILSVRC 2012, već je puno manja i sastoji se od par tisuća slika koje je također čovjek označavao. Glavna razlika tih dviju baza podataka je ta što PASCAL VOC baza podataka sadrži granične okvire koje je označio čovjek te se u treningu koriste kao referentni podaci, odnosno podatak po kojem program uči i podatak ka kojem program teži da postigle što sličnije rezultate tim graničnim okvirima [2].

Tablica 2.1 *Prosječan prosjek detekcije pomoću R-CNN mreže u usporedbi s ostalim mrežama*

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [18]†	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [34]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [36]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [16]†	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	**71.8**	**65.8**	**53.0**	**36.8**	**35.9**	**59.7**	**60.0**	**69.9**	**27.9**	**50.6**	**41.4**	**70.0**	**62.0**	**69.0**	**58.1**	**29.5**	**59.4**	**39.3**	**61.2**	**52.4**	**53.7**

U tablici 2.1 se može vidjeti usporedba prosječnog prosjeka detekcije R-CNN mreže s ostalim mrežama kao što su DPM v5, UVA, Regionlets, SegDPM. Sve te mreže su testirane na VOC 2010 bazi podataka, koja se sastoji od slika na kojima je čovjek obilježavao granične okvire. Na slici 2.5 se može vidjeti primjer slika koje se nalaze u VOC 2010 bazi podataka [2].

Slika 2.5 *VOC 2010 baza podataka*

2.3. Fast R-CNN

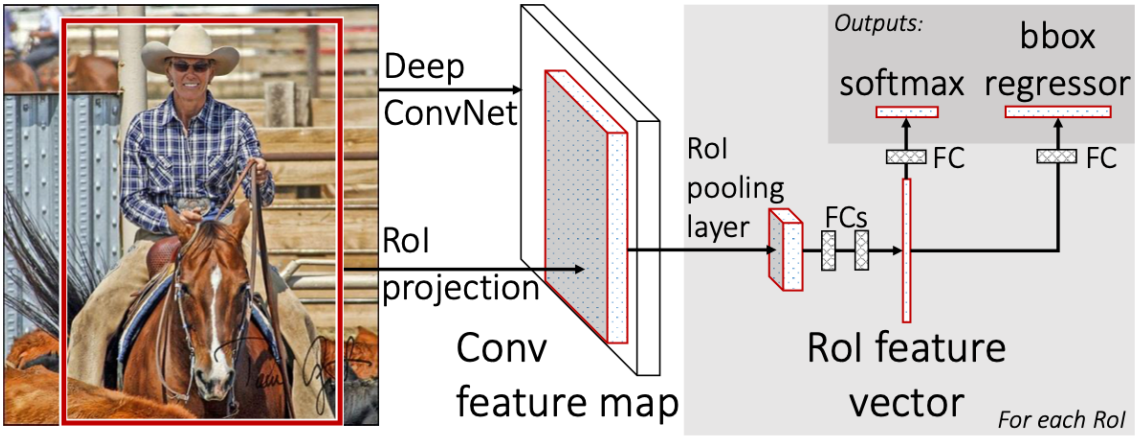
Fast R-CNN odnosno (*engl. Fast Region-based Convolutional Network*) je mreža za detekciju objekata. Fast R-CNN je baziran na prijašnjim radovima, ponajviše na R-CNN-u, koristeći duboke neuronske mreže. Mreža je 9 puta brža i učinkovitija od R-CNN-a. Fast R-CNN pokazuje novi način učenja koji nadopunjuje nedostatke R-CNN-a i SPPnet-a, povećavajući njihovu brzinu rada i preciznost detekcije **Error! Reference source not found.**.

2.3.1. Prednosti Fast R-CNN-a

Neke od prednosti Fast R-CNN-a su bolja kvaliteta i učinkovitost naspram R-CNN-a i SPPnet-a. Zatim, učenje u jednoj fazi. Također, učenje ažurira sve slojeve mreže. Posljednja veća prednost Fast R-CNN-a je ta da nije potrebna memorija na disku kako bi se kreirali vektori značajki. Fast R-CNN je napisan u Pythonu i C++u **Error! Reference source not found.**.

2.3.2. Arhitektura Fast R-CNN-a

Na slici 2.6 se može vidjeti ilustrirana Fast R-CNN arhitektura. Fast R-CNN mreža uzima kao ulaz cijelu sliku i niz prijedloga područja interesa. Područja interesa nastaju na jednak način kao i u R-CNN mreži, odnosno korištenjem selektivne pretrage. Mreža najprije obrađuje cijelu sliku pomoću potpuno povezane konvolucijske mreže. Rezultat te obrade je mapa vektora značajki. Dio ove mape obuhvaćen nekim interesnim područjem se skalira na mapu, fiksne veličine, nakon čega se kreira vektor značajki pomoću potpuno povezanih slojeva. Iz ovog se vektora izračunavaju dva izlazna vektora. Prvi izlazni vektor se izračunava korištenjem softmax funkcija, a govori kojoj klasi pripada objekt. Dok drugi izlazni vektor predstavlja granični okvir koji se sastoji od četiri broja. Granični okvir koji izlazi kao drugi vektor predstavlja korekciju početnog graničnog okvira interesnog područja. Drugim riječima konvolucijska mreža je smanjila širinu i visinu slike, ali je povećala njezinu dubinu, te detektirani objekt na određenom interesnom području i njegov granični okvir nisu jednakih veličina ni položaja zbog te prethodno navedene korekcije.



Slika 2.6 *Fast R-CNN arhitektura.*

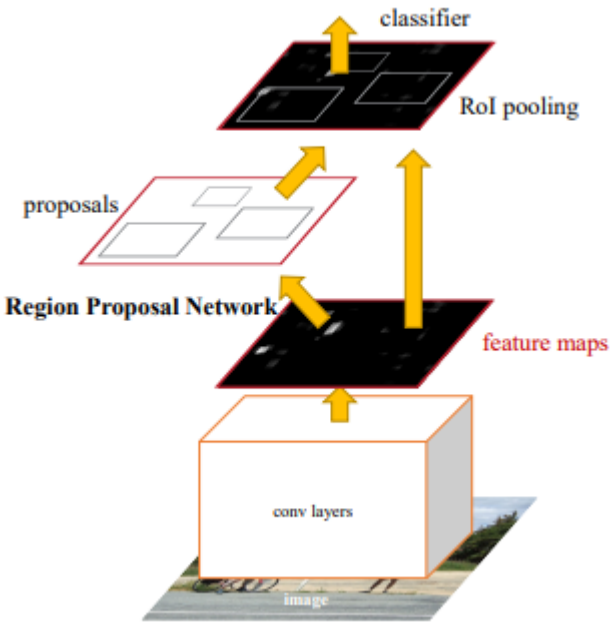
2.4. Faster R-CNN

Napredak u detekciji objekata potaknut je uspjehom metode prijedloga područja interesa i konvolucijskih neuronskih mreža temeljnih na području interesa [2]. Iako su CNN-ovi s prijedlogom područja interesa računalno skupi, kao što je prvotno spomenuto u [2], njihovi troškovi drastično su smanjeni zahvaljujući novim arhitekturama. Novije rješenje, Fast R-CNN, postiže vremena da skoro pripada u mreže koje detektiraju u stvarnom vremenu koristeći vrlo duboke neuronske mreže, zanemarujući vrijeme provedeno za generiranje prijedloga područja

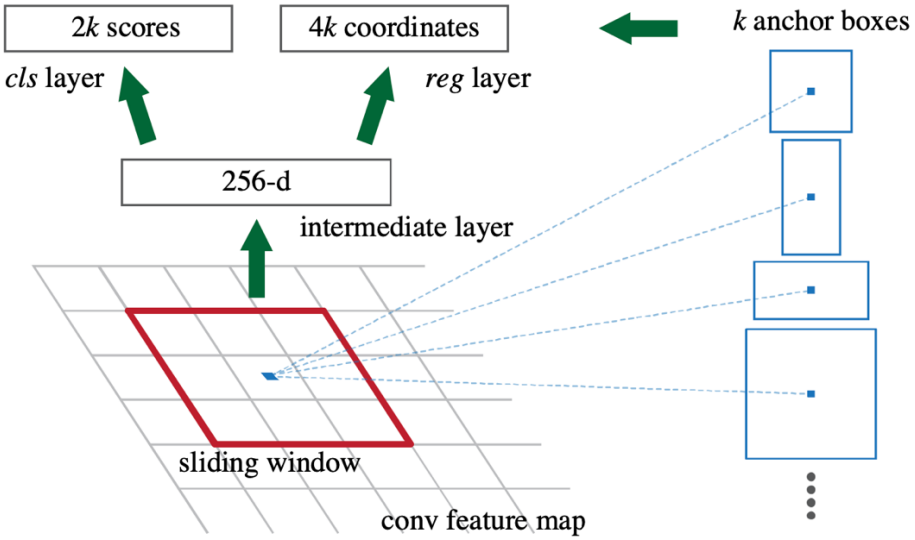
interesa. Metode prijedloga područja interesa obično se oslanjaju na pronalazak značajki i ekonomične sheme zaključivanja. Selektivna pretraga, jedan od najpopularnijih metoda, spaja superpiksele temeljene na značajkama niske razine. Faster R-CNN koristi jednu novu metodu koja nije selektivna pretraga, već metodu koja se zasniva na sidrima (*engl. anchor*) **Error! Reference source not found.**.

2.4.1. Arhitektura Faster R-CNN-a

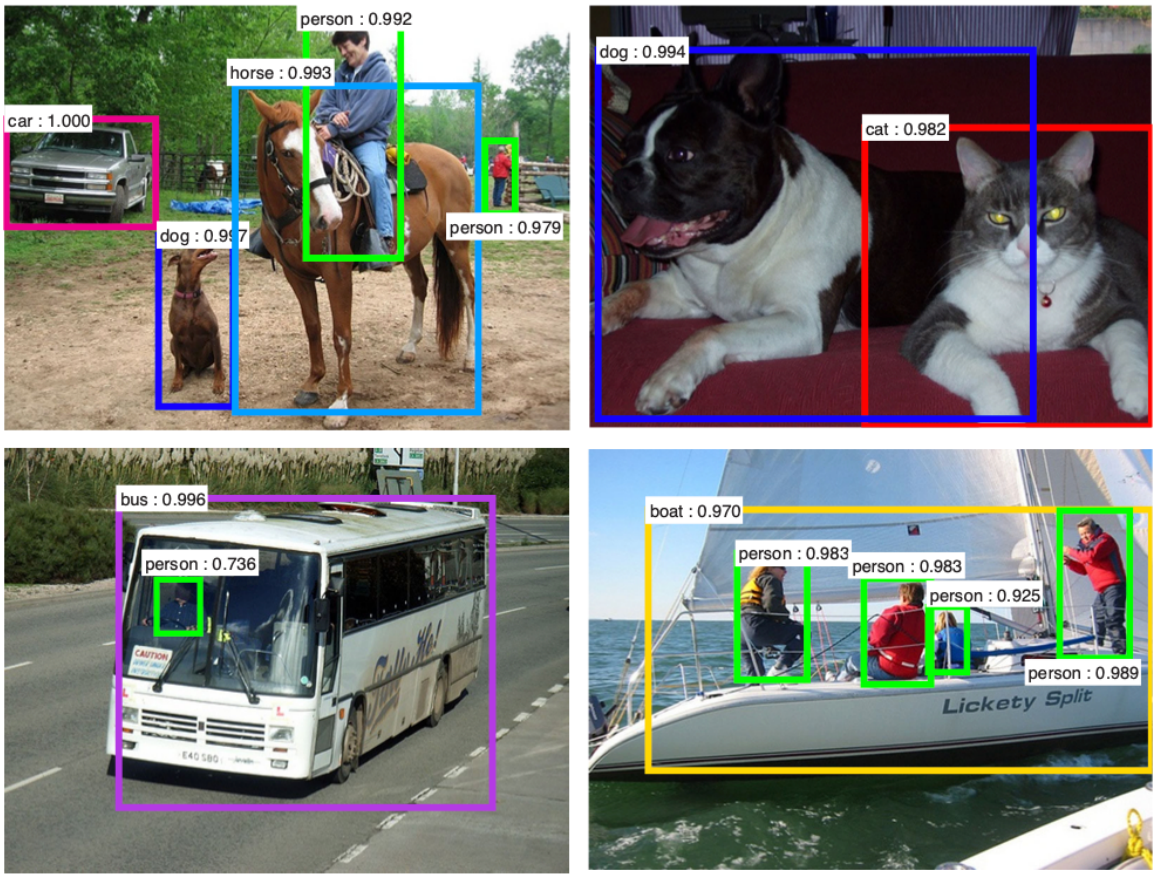
Faster R-CNN arhitektura slika 2.7, kao i do sada opisane mreže, kao ulaz ima sliku. Nakon toga se primjenjuje određeni broj potpuno povezanih konvolucijskih slojeva, pomoću kojih se dobiva mapa vektora značajki. Na svojstvenu mapu se primjenjuje takozvani klizni prozorčić veličina 3 x 3 pixela. Taj prozorčić klizi po svojstvenoj mapi kako bi prepoznao interesno područje. U svakom tom prozorčiću se provjeravaju 3 oblika u 3 veličine. Oblici su kvadratnog oblika, zatim pravokutnog koji ima veću visinu nego širinu, te pravokutnog koji ima veću širinu u odnosu na visinu. Na slici 2.8 se može vidjeti taj klizni prozorčić, te okviri koji se isprobavaju svakim pomakom kliznog prozorčića. U ovoj mreži takvih okvira ima 9. Primjena takvog načina se može vidjeti na slici 2.9 koja prikazuje primjere detekcije korištenjem načina objašnjenog prethodno na testnom skupu PASCAL VOC 2007. Također je važno naglasiti kako zbog takvog pristupa ova mreža ima širok spektar prepoznavanja na različitim skalama i raznim dimenzijama slika. Daljnji postupak svrstavanja u klasu je na sličan način kao u prethodnim mrežama **Error! Reference source not found.**.



Slika 2.7 *Faster R-CNN arhitektura*



Slika 2.8 *Mreža prijedloga regije.*



Slika 2.9 *Primjer detekcije pomoću prijedloga regija na PASCAL VOC 2007 testu.*

Tablica 2.2 *Rezultati detekcije Faster R-CNN-a na ispitnom skupu PASCAL VOC 2007 primjenom različitih postavki anchor-a.*

settings	anchor scales	aspect ratios	mAP (%)
1 scale, 1 ratio	128^2	1:1	65.8
	256^2	1:1	66.7
1 scale, 3 ratios	128^2	{2:1, 1:1, 1:2}	68.8
	256^2	{2:1, 1:1, 1:2}	67.9
3 scales, 1 ratio	$\{128^2, 256^2, 512^2\}$	1:1	**69.8**
3 scales, 3 ratios	$\{128^2, 256^2, 512^2\}$	{2:1, 1:1, 1:2}	**69.9**

U tablici 2.2 se mogu vidjeti rezultati Faster R-CNN-a na testnom skupi PASCAL VOC 2007 koji pokazuje kako izbor raznih omjera okvira utječe na detekciju objekta. Pokazalo se da primjenom 3 okvira i 3 različite veličine tih oblika okvira ostvaruje najbolji rezultat u detekciji objekata **Error! Reference source not found.**.

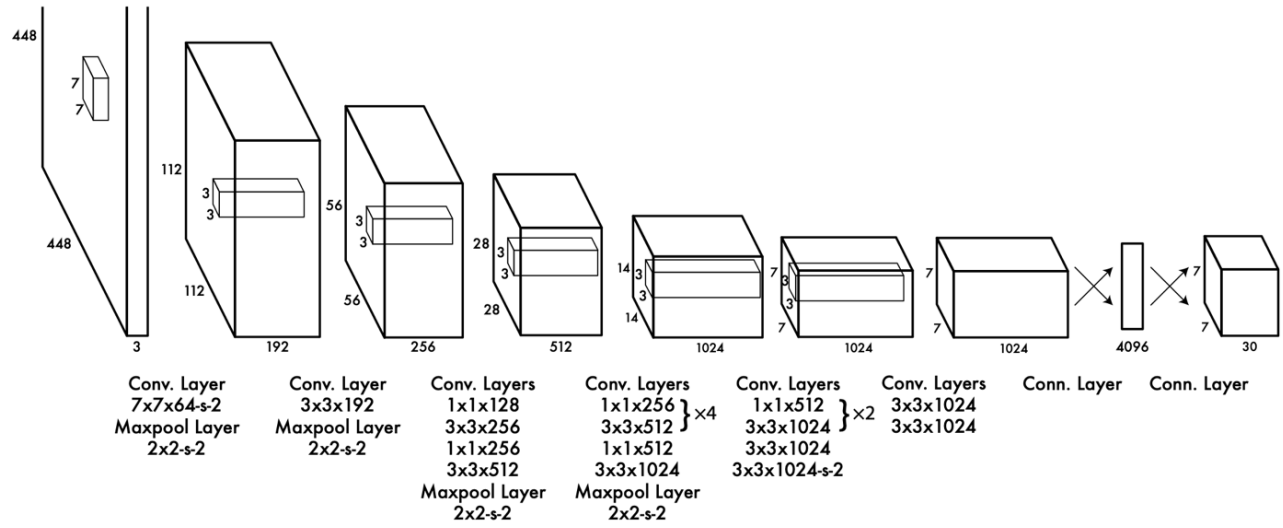
2.5. You Only Look Once

Ljudi kada pogledaju sliku odmah znaju koji su predmeti na slici, gdje se nalaze i kako djeluju. Ljudski vizualni sustav je brz i precizan što nam omogućava obavljanje složenih zadataka poput vožnje s manje razmišljanja, jer iskustvo i intuicija povećavaju vještinu. Brzi i precizni algoritmi za otkrivanje objekata omogućili bi računalima da voze automobile bez specijalnih senzora, omogućili bi pomoćnim uređajima da ljudskim korisnicima prenose podatke o okolini u stvarnom vremenu i prošire potencijal za opće namjene, u skladu s robotskim sustavima [7].

2.5.1. Arhitektura mreže

Arhitektura mreže je dizajnirana na kompliciraniji način u odnosu na prijašnje mreže. Na početku kao ulaz, ulazi slika veličine 448 x 448 x 3, odnosno slika veličine 448 x 448 u RGB formatu. Slika se podijeli na ćelije na način da se odabere broj podjela u stupce i retke na slici. U ovom primjeru je korišten broj 7, odnosno ulazna slika je podijeljena na 7 stupaca i 7 redaka što kreira mrežu od 49 ćelija. Nakon toga se primjenjuju 2 konvolucijska sloja veličine 7 x 7 x 64 te 2 Maxpool sloja veličine 2 x 2. Primjenom dva konvolucijska sloja i 2 Maxpool sloja izvorna slika se smanji u visini i širini, ali poveća u dubinu, te bude formata 112 x 112 x 192. Postupak se ponavlja s različitim brojem ponavljanja konvolucijskog sloja, kao i Maxpool sloja koji na kraju dovedu sliku u format 7 x 7 x 1024 nakon kojeg se primjenjuje potpuno povezani konvolucijski

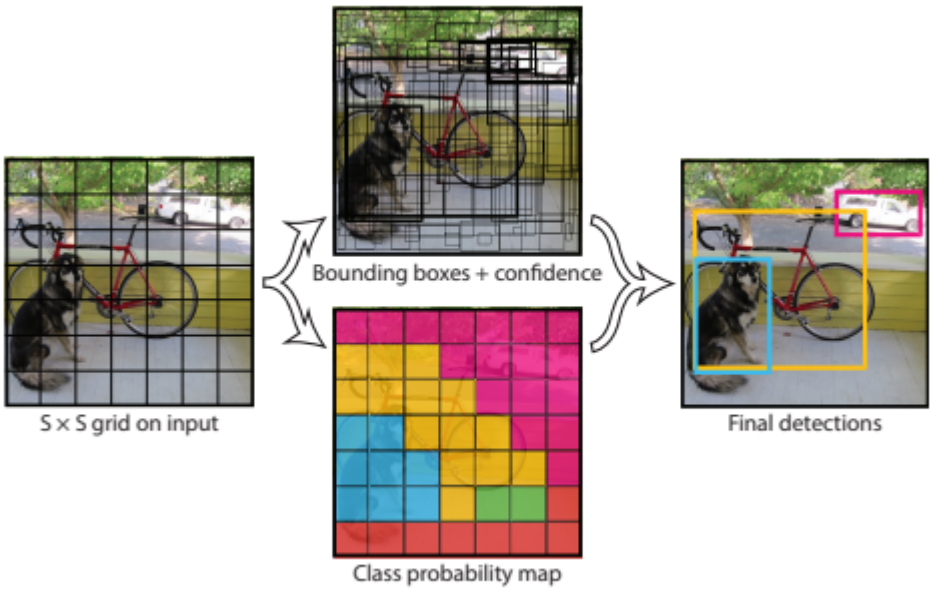
sloj veličine 4096 koji kao izlaz vrati sliku veličine 7 x 7 x 30. Izraz (2-1) definira dimenzije izlazne slike. U izrazu (2-1) *S* predstavlja veličinu ćelije, dok B predstavlja broj koliko graničnih okvira želimo da mreža kreira za svaku ćeliju. Broj 5 u izrazu (2-1) se koristi zato što granični okvir ima 4 vrijednosti koje ga opisuju što zajedno s vrijednošću pouzdanosti pridruženom graničnom okviru čini ukupno 5 vrijednosti za svaki granični okvir. Te vrijednosti su X i Y koordinata gornjeg lijevog kuta graničnog okvira, zatim širina i visina graničnog okvira, te razina sigurnosti. Oznaka C u formuli predstavlja broj klasa na koje je mreža naučena [7].



Slika 2.10 *Arhitektura.*

$$S\ x\ S\ x\ (B * 5 + C) \qquad (2\text{-}1)$$

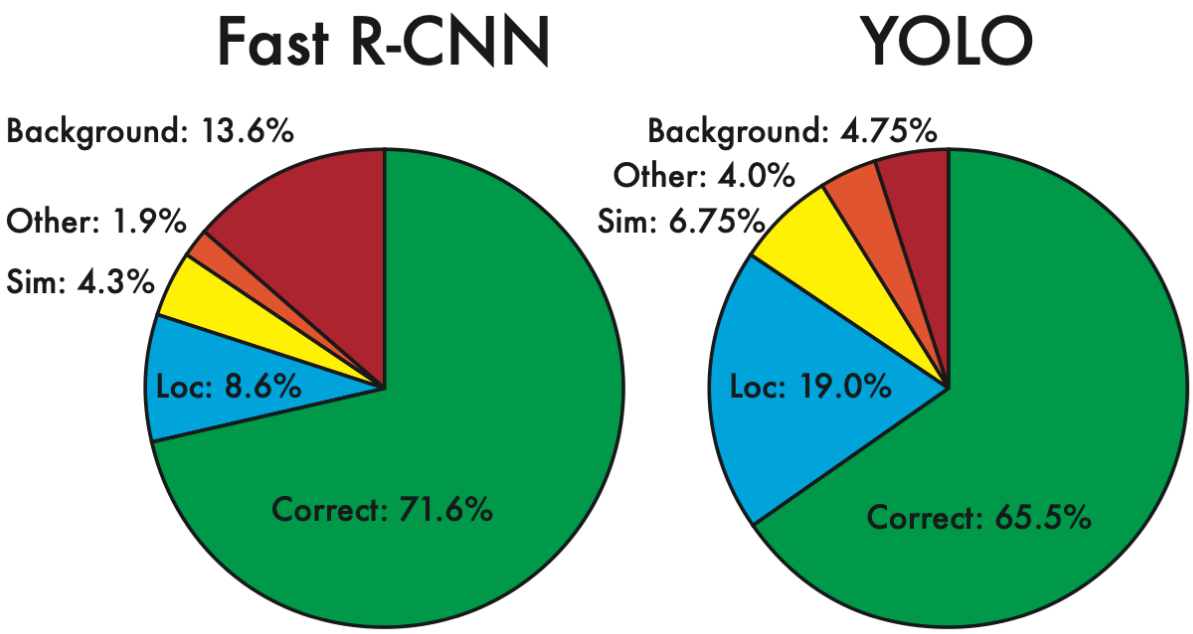
Na slici 2.11 se može vidjeti na koji način se izvorna slika podijeli u mrežu ćelija, te kako svaka ćelija kao izlaz daje mapu vjerojatnosti pripadnosti pojedinoj klasi za svaku ćeliju, te granične okvire i određeni broj koji predstavlja koliko je mreža sigurna za pojedini granični okvir. Na slici 2.11 na posljednjem pravokutniku se može vidjeti kako je mreža detektirala više objekata na jednoj slici, te svaki od njih uokvirila graničnim okvirom [7].



Slika 2.11 *Model YOLO mreže*

2.5.2. Usporedba sa sličnim mrežama

R-CNN i Fast R-CNN zamjenjuju selektivno pretraživanje statičkim prijedlozima za ograničavanje okvira. Iako je mnogo brži od R-CNN-a, on u stvarnom vremenu još uvijek zaostaje zbog nedostatka dobrih prijedloga. Fast R-CNN ubrzava fazu klasifikacije R-CNN-a, ali i dalje se oslanja na selektivno pretraživanje koje može trajati oko 2 sekunde po slici da se generiraju prijedlozi graničnih okvira. Stoga ima visoku mAP, ali pri 0,5 fps još je daleko od stvarnog vremena. Noviji Faster R-CNN zamjenjuje selektivno pretraživanje dubokom neuronskom mrežom kako bi predložio granične okvire. U testovima, njihov najprecizniji model postiže brzinu od 7 slika u sekundi, dok manji, manje točan model radi na 18 fps. Verzija Faster R-CNN-a VGG-16 daje za 10 veći mAP, ali je i 6 puta sporija od YOLO-a. Zeiler-Fergus brži R-CNN je samo 2,5 puta sporiji od YOLO, ali je i manje precizan. Na slici 2.12 se može vidjeti Fast R-CNN nasuprot YOLO. Ovi grafovi prikazuju postotak lokalizacije i pozadinskih pogrešaka u top N detekcijama za različite kategorije (N = # objekata u toj kategoriji) [7].



Slika 2.12 *Analiza grešaka.*

2.5.3. Ograničenja mreže

Mreža YOLO se ograničava u pojedinim segmentima kako bi mogla djelovati u stvarnom vremenu. Neka od tih ograničenja su da se po svakoj ćeliji u mreži može kreirati samo 2 granična okvira i može se odrediti samo jedna klasa za pojedinu ćeliju. Nadalje, YOLO mreža ima poteškoća kod prepoznavanja malih objekata kao što su jato ptica. Također, jedan od problema je taj što je YOLO mreža naučena na graničnim okvirima, te iz tog razloga ima problema u prepoznavanju ako su neki objekti neuobičajenih veličina ili proporcija [7].

2.6. VolumeNet

Prethodno objašnjeni modeli, algoritmi, sustavi i programska rješenja su se najviše bazirala na detekciji objekta u stvarnom vremenu, s velikom preciznošću, visokom točnošću i što manjim računalnim zahtjevima. Tako se pokazalo kako je Faster R-CNN bio najbolji kandidat jer je on postigao bolje rezultate s manjom računalnom zahtjevnošću. Najčešća primjena dosadašnjih modela je najviše u autonomnoj vožnji, jer je u tom području bitna detekcija objekta u stvarnom vremenu i potrošnja minimalne količine energije. Problemi se javljaju ako se takvi modeli pokušaju koristiti u nekakvom drugom području kao što je detekcija objekta za robotsku manipulaciju ili slično. Problem je taj što se ti modeli baziraju na brzom detektiranju i preciznom klasificiranju objekta na slici ili u prostoru, ali ne daju nikakve informacije o položaju objekta, niti

o dijelovima objekta relevantnim za neku radnu operaciju, npr. drška šalice i slično. Osim toga, važno je znati koji je najbolji položaj da robotska ruka prihvati neki objekt. VolumeNet [13] je jedan način na koji je taj problem riješen. Temeljen je na aproksimaciji oblika pomoću poliedra koji se mogu definirati deskriptorima čije komponente odgovaraju stranicama poliedra. VolumeNet je novi parametarski model 3D oblika, koji omogućava veliku unutarklasnu varijabilnost i vrlo učinkovito prilagođavanje modela na dubinsku sliku. Prilagodba modela dubinskoj slici pruža informacije o parametrima položaja i oblika objekta. VolumeNet se može koristiti za klasifikaciju jednostavnih kućanskih objekata, s rezultatima konkurentnim nekoliko najsuvremenijih pristupa [5].

2.6.1. Rezultati

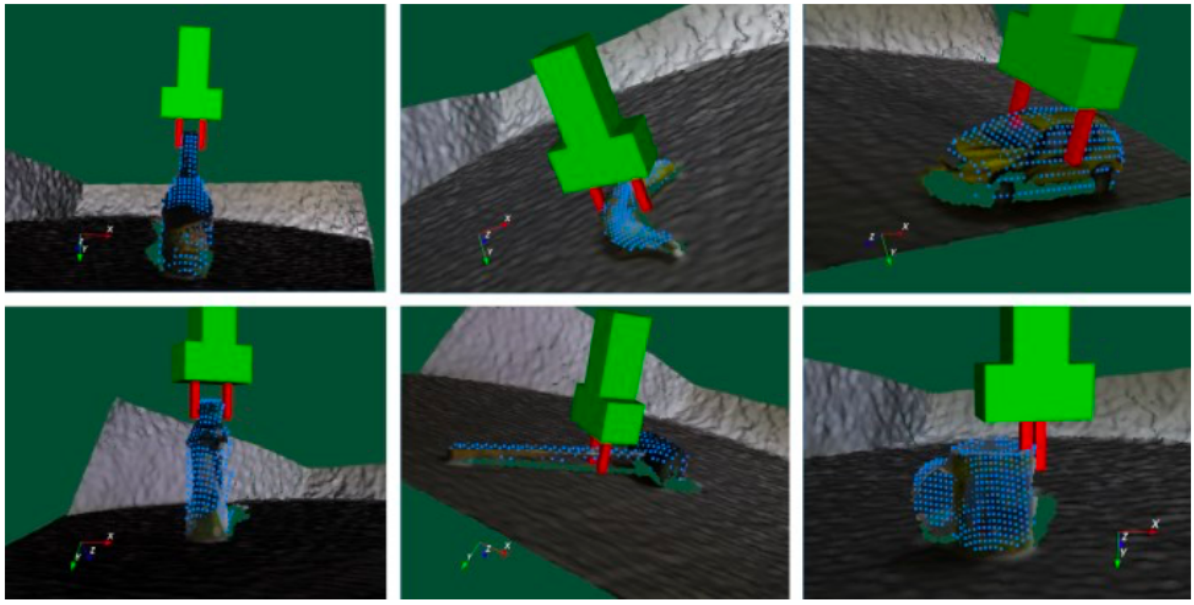
U tablici 2.3 se mogu vidjeti parametri VolumeNet modela nekoliko klasa objekata, najviše deskriptora ima klasa krafna (engl. *donut)* dok su najmanji deskriptori triju klasa: jabuka (engl. *apple*), automobil(engl.*car*) i tetrapak. U tablici 2.4 nalaze se rezultati klasifikacije raznih objekata pomoću nekoliko metoda detekcije i klasifikacije objekata. Iz tablice se može vidjeti kako je VolumeNet najtočniji u klasifikaciji navedenih klasa s čak 94% ukupne točnosti. Na Slici 2.3 je prikazana robotska hvataljka u položajima koji su definirani pomoću odabranih stranica VolumeNet modela koji su prilagođeni objektima na sceni. Plave točke predstavljaju instance VolumeNet-a projicirane na RGB-D sliku [5].

Tablica 2.3 *Parametri razreda klasa modela.*

Class	**Metamodel**	**Descriptor size**	**Latent space dimensions**
apple	convex	66	2
banana	convex + torus	79	4
bottle	2 × convex	102	6
bowl	convex + concave	104	6
car	convex	66	2
donut	convex + torus	173	2
hammer	2 × convex	107	7
mug	2 × convex + concave	149	6
tetrapak	convex	66	2
toilet paper	convex + concave	82	2

Tablica 2.4 *Rezultati klasifikacije objekata.*

Class	VFH	CVFH	ESF	SHOT	VN
apple	64	95	**99**	93	80
banana	97	99	89	73	**100**
bottle	84	87	94	**99**	**99**
bowl	97	97	88	**100**	**100**
car	87	91	**98**	95	72
donut	78	96	96	90	**100**
hammer	93	98	**100**	97	99
mug	98	**100**	**100**	95	94
tetrapak	5	5	**91**	74	**91**
toilet paper	20	11	61	58	**100**
OVERALL	72	78	92	87	**94**

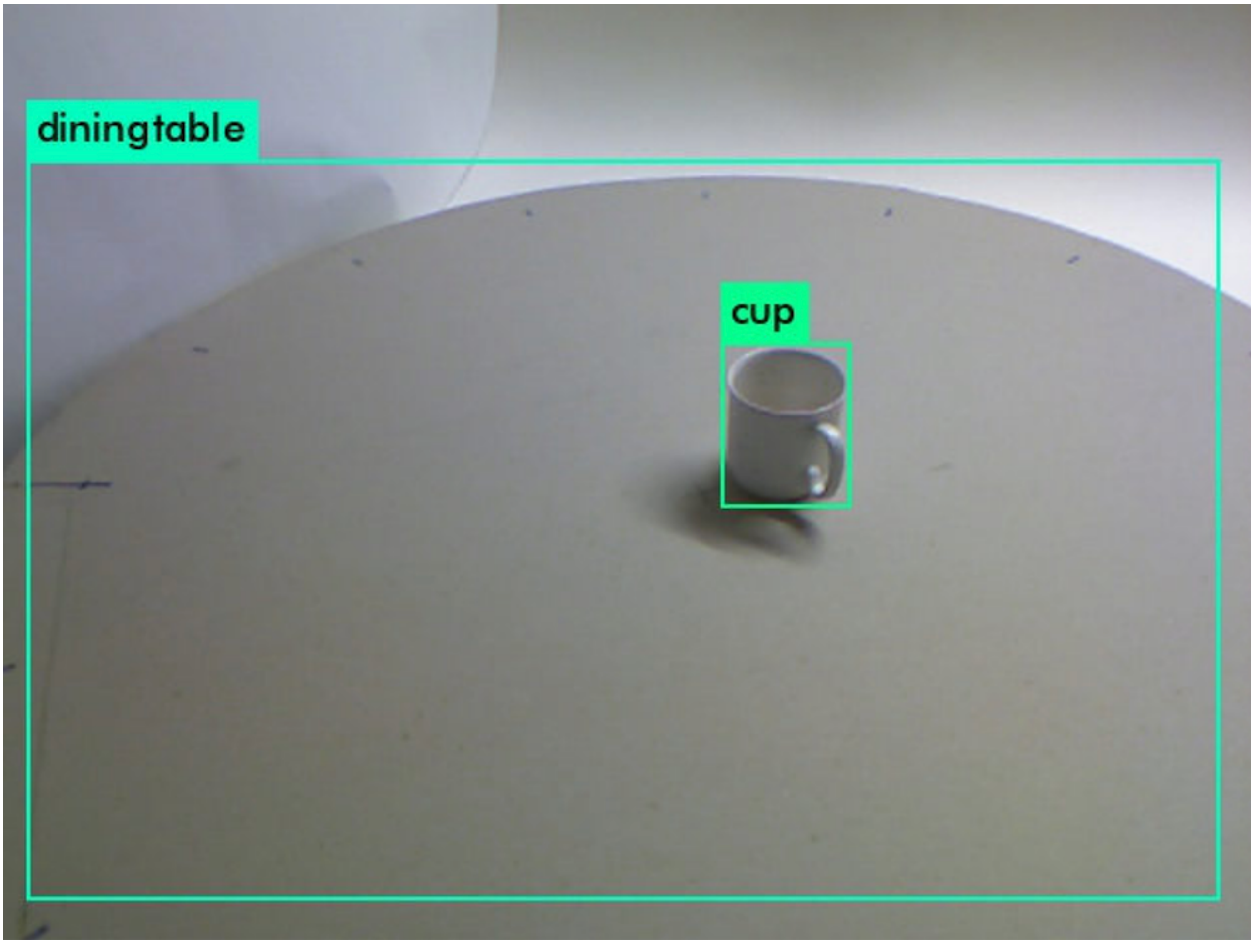


Slika 2.13 *Prikaz VolumeNet modela.*

3. PROGRAMSKA REALIZACIJA

U ovom je radu prepoznavanje objekata i njihovih dijelova ostvareno na sljedeći način. Dostupan je skup podataka, 3DNet dataset [6], koji se sastoji od slika u boji i pripadajućih dubinskih slika predstavljenih oblakom točaka. Na slikama se nalazi po jedan predmet iz jedne od 10 klasa navedenih u tablici 2.4. Predmeti se nalaze na stolu. Slika u boji prvo se obrađuje unaprijed naučenom YOLO [7] mrežom za prepoznavanje objekata. S obzirom da se među naučenim klasama unutar YOLO-a ne nalaze sve klase iz 3D Net skupa podataka, algoritmu se predaju samo slike iz 7 klasa navedenih u tablici 4.1. Ova mreža osim detektirane klase objekata vraća i granični okvir (engl. *bounding box)* na slici unutar kojeg se nalazi klasificirani predmet. Koordinate detektiranog graničnog okvira, kao i klasa detektiranog objekta spremaju se u datoteku za daljnju obradu. Podaci se spremaju u datoteku iz razloga što VolumeNet i YOLO nisu napisani u istom projektu, odnosno YOLO mreža je isprogranirana da vraća koordinate graničnog okvira, dok VolumeNet kao ulaz uzima .ply datoteku i koordinate graničnog okvira. Ove se koordinate, zajedno s dubinskim slikama predaju algoritmu VolumeNet. Detektirani granični okviri smanjuju područje pretraživanja na slici. Izlaz iz VolumeNet-a su klasa detektiranog objekta i njegov položaj. Osim toga, određuje se dio objekta prikladan za hvatanje i približan položaj hvataljke robotskog manipulatora. U nastavku je detaljnije opisana programska realizacija predloženog rješenja.

Nakon primjene algoritma YOLO, dobiveni su granični okviri i nazivi klasa detektiranih objekata, kao što je prikazano na Slici 3.1.

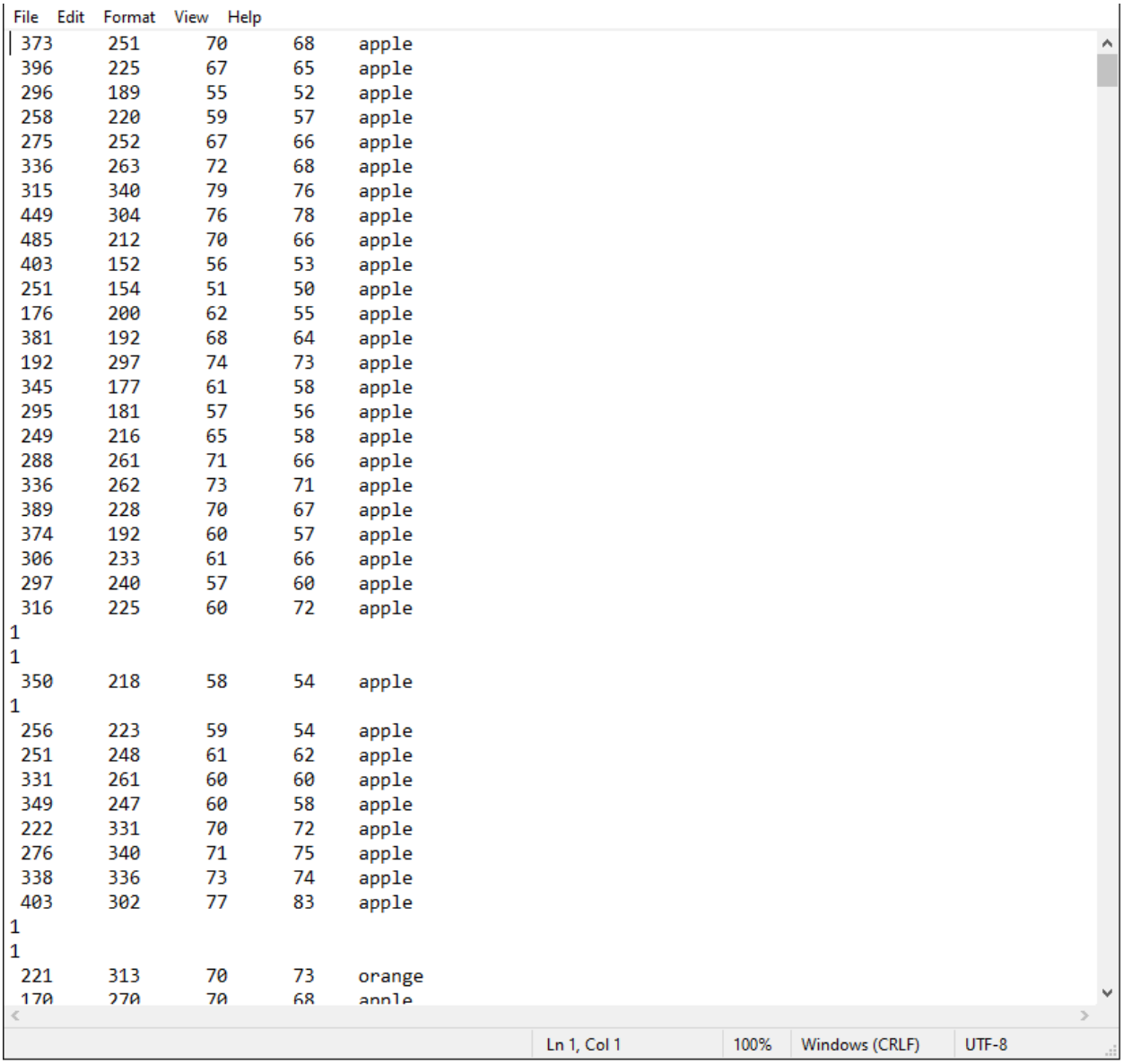


Slika 3.1 *Prikaz graničnih okvira dobivenih YOLO mrežom.*

U datoteci image.c u *darknet solution*-u unutar YOLO-a nalazi se dio koda koji je zaslužan za izračunavanje graničnih okvira te dodjeljivanje oznaka detektiranim objektima. U kodu 3.1 se može vidjeti kako se otvara datoteka „bbox.txt“ koja služi za zapisivanje koordinata graničnih okvira. Izgled te datoteke se može vidjeti u slici 3.2. Datoteka u jednom retku ima izlaz YOLO mreže za jednu sliku iz skupa slika za testiranje. Prvi broj je X koordinata gornjeg lijevog kuta graničnog okvira, drugi broj je Y koordinata gornjeg lijevog kuta graničnog okvira, treći broj je širina graničnog okvira, dok je četvrti broj visina graničnog okvira. Na kraju retka je tekst koji predstavlja detektiranu klasu u određenom graničnom okviru.

```
fptr = fopen("D:/diplomski/yolofile/yolo output/bbox.txt", "a");
```

Kod 3.1 *Otvaranje file-a.*



Slika 3.2 *bbox.txt datoteka.*

Kod određivanja koordinata graničnih okvira važna su nam četiri broja.

$$X = (x - \frac{w_f}{2}) * W \tag{3-1}$$

$$Y = (y - \frac{h_f}{2}) * H \tag{3-2}$$

$$w_d = w_f * W \tag{3-3}$$

$$h_d = h_f * H \tag{3-4}$$

U formuli (3-1) se može vidjeti postupak računanja X koordinate gornjeg lijevog kuta graničnog okvira. Oznaka x predstavlja vrijednost x koordinate graničnog okvira, w_f predstavlja vrijednost širine graničnog okvira, dok W predstavlja vrijednost širine slike. Formula (3-2) predstavlja izračun Y koordinate gornjeg lijevog kuta graničnog okvira, y predstavlja *float* vrijednost y koordinate graničnog okvira, h_f predstavlja *float* vrijednost visine graničnog okvira, dok H predstavlja *int* vrijednost visine slike. Treća formula (3-3) predstavlja izračun širine graničnog okvira. Posljednja formula (3-4) predstavlja izračun visine graničnog okvira. Nakon toga se pomoću naredbe fprintf ta četiri broja upisuju u datoteku, kao što je prikazano u kodu 3.2.

```
fprintf(fptr, "%4.0f\t%4.0f\t%4.0f\t%4.0f\t%s\n",(selected_detections[i].det.bbox.x -
selected_detections[i].det.bbox.w / 2) * im.w, (selected_detections[i].det.bbox.y -
selected_detections[i].det.bbox.h / 2) * im.h,selected_detections[i].det.bbox.w *
im.w, selected_detections[i].det.bbox.h * im.h, names[best_class]);
```

Kod 3.2 *Proračun koordinata graničnih okvira.*

Funkcija get_actual_detection, unutar YOLO-a, prepoznaje objekte na slici. Jedan od izlaza ove funkcije je selected_detections_num, odnosno broj detektiranih objekta. Poziv ove funkcije je prikazan u kodu 3.3.

```
detection_with_class* selected_detections = get_actual_detections(dets, num, thresh,
&selected_detections_num);
```

Kod 3.3 *get_actual_detections funkcija.*

Vrijednost selected_detections_num je potrebna kako bismo mogli proći kroz sve objekte detektirane na određenoj slici. Ove se detekcije sortiraju te se objektu na slici dodjeljuje klasa s najvećom vjerojatnosti detekcije, kao što je prikazano u kodu 3.4.

```
float* detected_objects = (float*)calloc(10, sizeof(float));

for (j = 0; j < selected_detections_num; j++) {
	const int best_class = selected_detections[j].best_class;
	detected_objects[j] = selected_detections[j].det.prob[best_class] * 100;
}

float higher_percent = detected_objects[0];

for (b = 0; b < selected_detections_num; b++) {
	if (detected_objects[b] > higher_percent)
		higher_percent = detected_objects[b];
	}
```

Kod 3.4 *Sortiranje prema vjerojatnosti detekcije objekta.*

Nakon toga se ulazi u petlju koja se ponavlja onoliko koliko je objekata detektirano na slici te se u datoteku upisuju parametri graničnog okvira samo za jedan detektirani objekt koji zadovoljava određene uvjete. S obzirom da se na testnom skupu podataka na svakoj slici nalazi samo jedan objekt, prvi uvjet osigurava da se za pojedinu sliku sprema samo jedan granični okvir. U slučaju detekcije više objekata na slici, provjerava se je li veličina objekta veća od ¼ slike. Ukoliko više graničnih okvira zadovoljava ovaj uvjet, tada se uzima onaj granični okvir čije je središte bliže središtu slike. Naime, u testnom skupu objekti su smješteni blizu središta slike te ne zauzimaju više od četvrtine slike. U slučaju da je za isti granični okvir unutar YOLO-a predloženo više klasa, pohranjuje se taj granični okvir, a kao ispravna klasa uzima se ona s najvećom vjerojatnošću. Kako bi se kasnije napravila matrica konfuzije, također je napravljena struktura koja služi za brojanje detektiranih objekata i čuvanje broja određenih klasa. Nakon ispisa graničnog okvira u datoteku, provjerava se koja je klasa detektirana te se vrijednost elementa koji označava detektiranu klasu povećava. Result_apple i ostale vrijednosti su postavljene kao int sa vrijednošću 1. Nakon detekcije se provjerava koji je objekt detektiran pa se vrijednost promjeni na 0 za objekt koji je detektiran. U strukturi su postavljeni class_apple i ostali elementi koji odgovaraju klasama koje se detektiraju i njihova vrijednost je postavljena na 0, ukoliko je slučaj da je određen neki objekt odgovarajući element se poveća za 1. Određivanje detektirane klase je prikazano u kodu 3.5.

```
if (result_apple == 0)
       class_1.class_apple += 1;

if (result_banana == 0)
       class_1.class_banana += 1;

if (result_bottle == 0)
       class_1.class_bottle += 1;

if (result_bowl == 0)
       class_1.class_bowl += 1;

if (result_car == 0)
       class_1.class_car += 1;

if (result_cup == 0)
       class_1.class_cup += 1;

if (result_donut == 0)
       class_1.class_donut += 1;

if (result_apple != 0 && result_banana != 0 && result_bottle != 0 && result_bowl != 0
&& result_car != 0 && result_cup != 0 && result_donut != 0)
       class_1.class_other += 1;
```

Kod 3.5 *Određivanje detektirane klase.*

Ako je detektiran jedan objekt na slici samo se ispisuju koordinate graničnog okvira tog objekta u bbox.txt datoteku te se također provjerava koja je to klasa te se ona zapisuje u strukturu class1 . Ako na slici nije detektiran niti jedan objekt, u tom slučaju se u datoteku unosi vrijednost 1 kako bi kasnije takvu sliku bilo lakše izbaciti iz skupa za daljnju analizu. U kodu 3.6 i 3.7 se mogu vidjeti i ovi uvjeti.

```
if (selected_detections_num == 1) {
       fprintf(fptr, "%4.0f\t%4.0f\t%4.0f\t%4.0f\t%s\n",
       (selected_detections[i].det.bbox.x - selected_detections[i].det.bbox.w / 2) *
       im.w, (selected_detections[i].det.bbox.y - selected_detections[i].det.bbox.h /
       2) * im.h,selected_detections[i].det.bbox.w * im.w,
       selected_detections[i].det.bbox.h * im.h, names[best_class]);

if (result_apple == 0)
       class_1.class_apple += 1;

if (result_banana == 0)
       class_1.class_banana += 1;

if (result_bottle == 0)
       class_1.class_bottle += 1;

if (result_bowl == 0)
       class_1.class_bowl += 1;

if (result_car == 0)
       class_1.class_car += 1;

if (result_cup == 0)
       class_1.class_cup += 1;

if (result_donut == 0)
       class_1.class_donut += 1;

if (result_apple != 0 && result_banana != 0 && result_bottle != 0 && result_bowl != 0
&& result_car != 0 && result_cup != 0 && result_donut != 0)
       class_1.class_other += 1;
}
```

Kod 3.6 *Uvjet ukoliko je detektiran jedan objekt.*

```
if (selected_detections_num == 0){
       fprintf(fptr, "1\n");
       class_1.class_undetected += 1;
}
```

Kod 3.7 *Uvjet ukoliko nije detektiran niti jedan objekt.*

Tako se analiziraju sve slike te se stvara jedna datoteka koja sadrži sve granične okvire i detektirane klase. Nakon toga se podaci koji se nalaze u bbox.txt datoteci razdvajaju, tako da se svaki granični okvir sprema u zasebnu datoteku naziva jednakog slici na kojoj je detektiran, s ekstenzijom .bbx. Za razdvajanje je korišten python. Prvi python module kreira listu, zatim se pomiče na direktorij u kojem se nalazi datoteka u kojoj su nazivi svih slika bez ekstenzija. Nakon toga se ulazi u petlju, čita se linija po linija i svakom nazivu se dodaje .bbx, dodaje se u listu, premješta se u drugi direktorij u kojem se kreira datoteka za svaku sliku. Nakon toga se iščitavaju koordinate graničnog

okvira iz bbox.txt datoteke i upisuju u pojedinu .bbx datoteku. Python module je prikazan u kodu 3.8.

```
import os
bbx_list = []

x = 0
i = 0

os.chdir("D:\diplomski\yolofile\yolo read")
file1 = open("YoloInputPics_bak.txt", 'r')

for x in range(1651):
   lines = file1.readline()
   lines = lines[:-1]
   lines = lines + ".bbx"
   bbx_list.append(lines)
   os.chdir("D:\diplomski\yolofile\yolo output\data")
   file2 = open(lines, "w")
   x=x+1

file1.close()
file2.close()

os.chdir("D:\diplomski\yolofile\yolo output")
file3 = open("bbox.txt", 'r')

for i in range(1651):
   line = file3.readline()
   line = line[:19]
   os.chdir("D:\diplomski\yolofile\yolo output\data")
   file4 = open(bbx_list[i], "w")
   print(line)
   print(line,file=file4)
   i=i+1

file3.close()
file4.close()
```

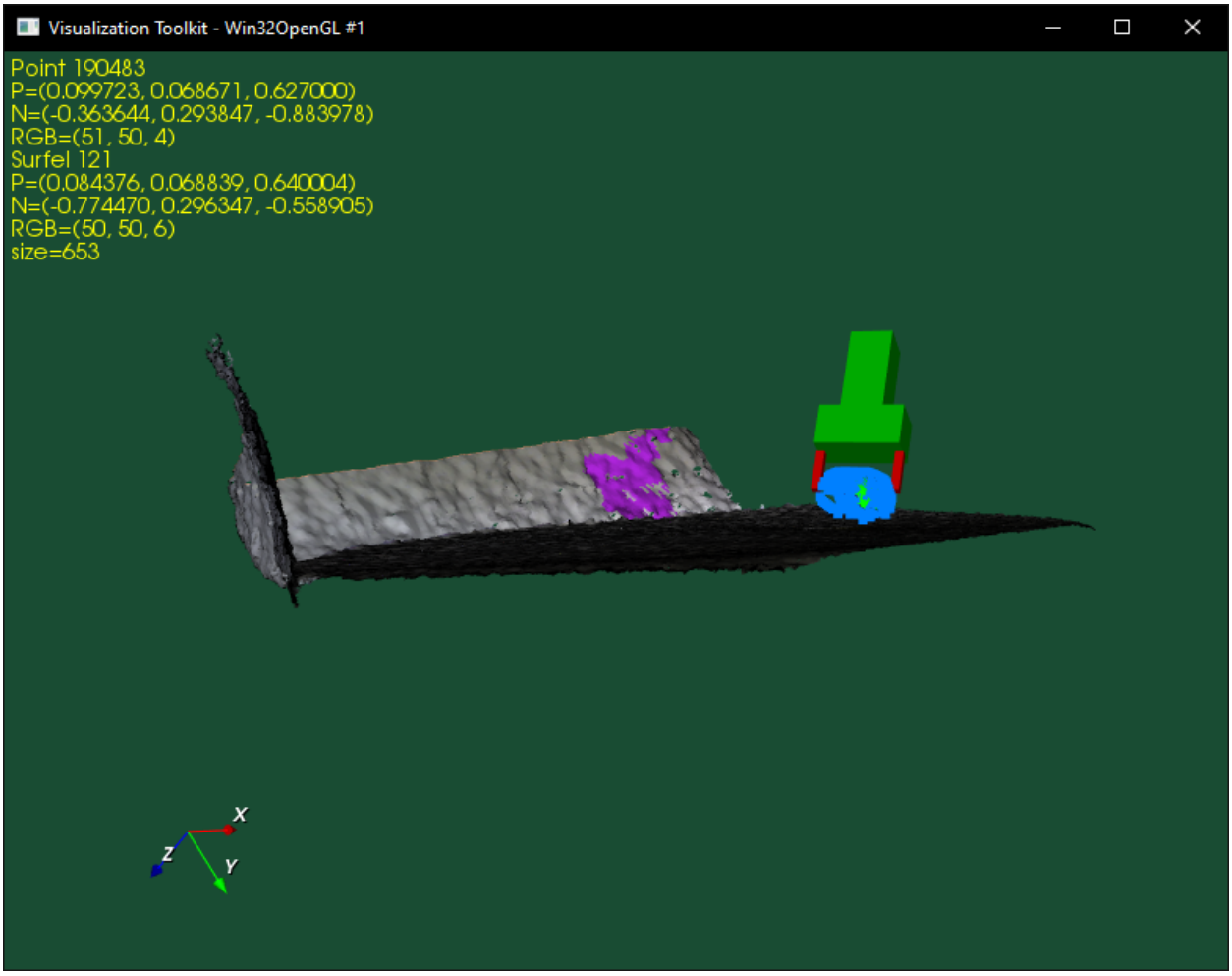
Kod 3.8 *Python module za kreiranje i unos graničnih okvira.*

Nakon toga slijedi modul koji će izbrisati sve .bbx datoteke koje ne sadrže granične okvire, već sadrže broj 1, kao oznaku da objekt nije detektiran YOLO mrežom. Python module se može vidjeti u kodu 3.9.

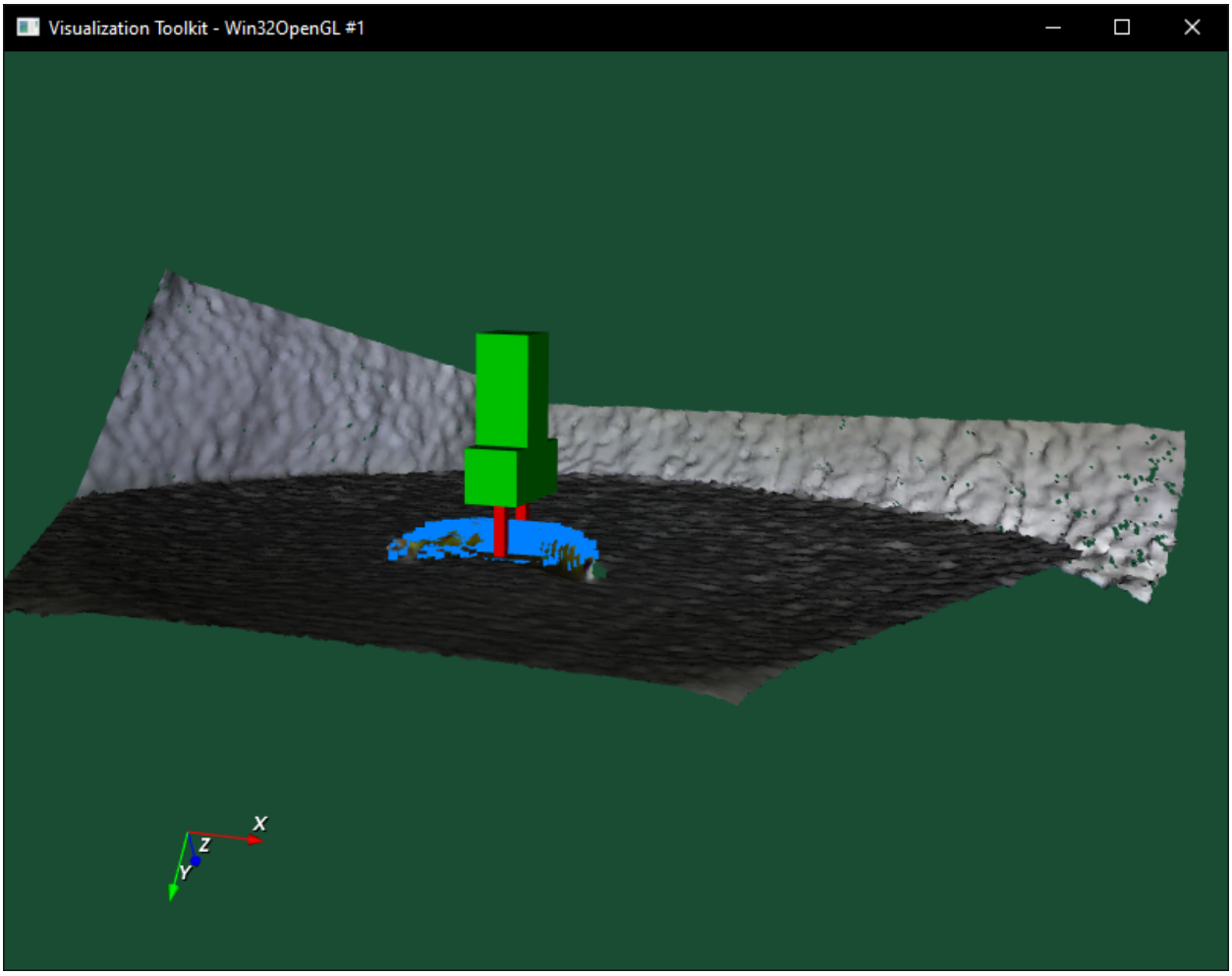
```
import os

bbx_list = []
x = 0
i = 0

os.chdir("D:\diplomski\yolofile\yolo read")
file1 = open("YoloInputPics_bak.txt", 'r')

for x in range(1651):
   lines = file1.readline()
   lines = lines[:-1]
   lines = lines + ".bbx"
   bbx_list.append(lines)
   x=x+1

file1.close()

os.chdir("D:\diplomski\yolofile\yolo output")
file2 = open("bbox.txt", 'r')

for i in range(1651):
   line = file2.readline()
   if line == "1\n":
      os.chdir("D:\diplomski\yolofile\yolo output\data")
      os.remove(bbx_list[i])
   i=i+1

file2.close()
```

Kod 3.9 *Python modul koji služi za brisanje datoteka koje ne sadrže koordinate graničnog okvira.*

Nakon kreiranja i unošenja koordinata graničnih okvira u .bbx datoteke, te se datoteke i njihove odgovarajuće dubinske slike s ekstenzijom .ply predaju VolumeNet-u. VolumeNet na temelju dohvaćenih datoteka .ply i .bbx od određene slike detektira i klasificira objekt. Ovaj algoritam određuje položaj objekta na slici te osmišljava položaj hvatanja tog objekta. Na slikama 3.3 i 3.4 su prikazani rezultati dobiveni VoluNet algoritmom.



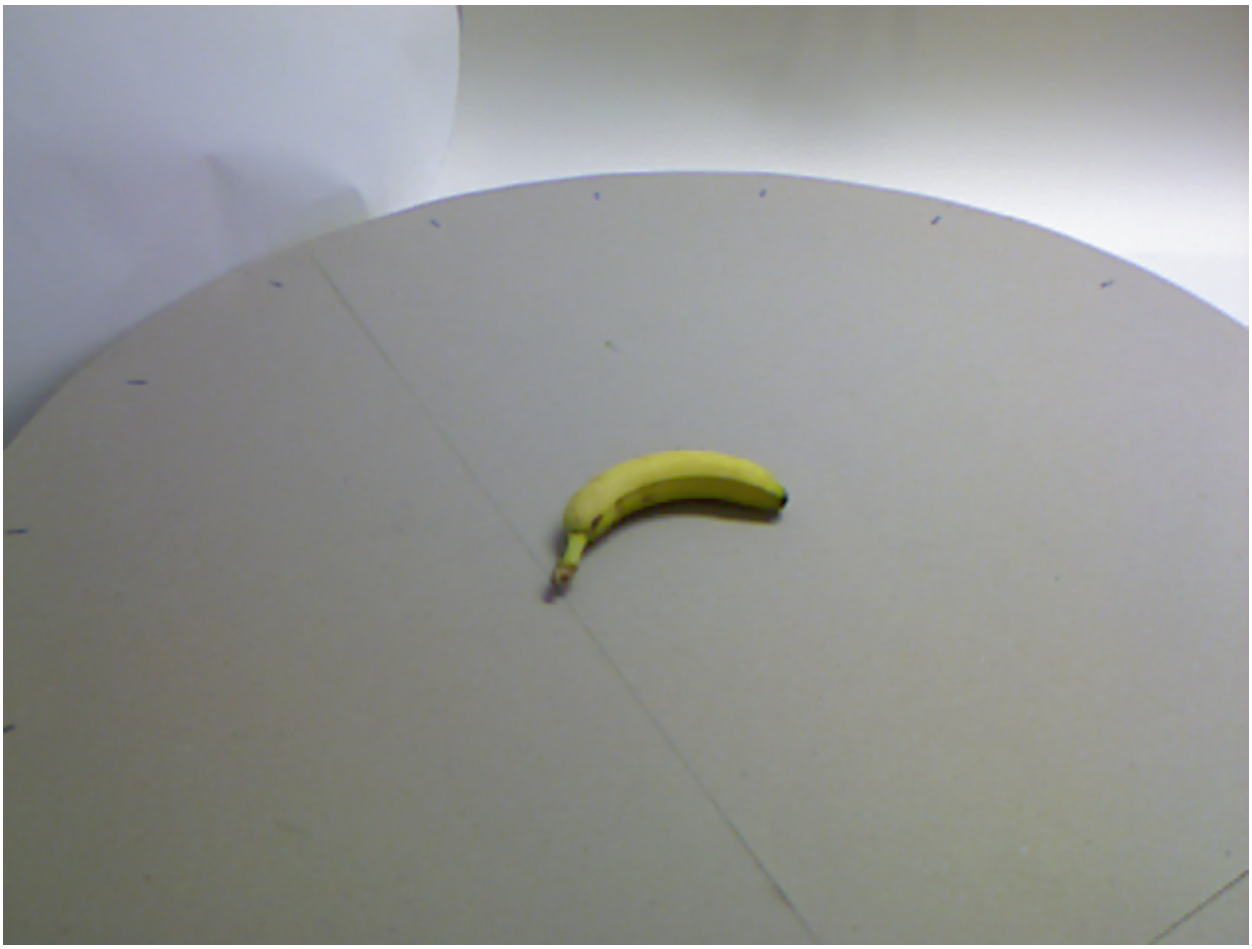
Slika 3.3 *Rezultat VolumeNet algoritma za objekt jabuku.*



Slika 3.4 *Rezultat VolumeNet algoritma za objekt banana.*

4. EVALUACIJA PROGRAMA

Evaluacijom programa se ispituje točnost klasifikacije objekata na RGB-D slikama. Evaluacija programa se odvijala na skupu slika 3Dnet, klase u koje su svrstani objekti nalaze se u nizu: jabuka, banana, boca, zdjela, automobil, krafna i čaša. U originalnom skupu postoje još tri klase objekata, ali one nisu analizirane jer ne postoje u skupu za učenje YOLO mreže. Ukupan broj slika koji je analiziran je 1107. Na primjeru banane bit će objašnjen postupak detekcije. Na slici 4.1 se može vidjeti RGB slika banane.

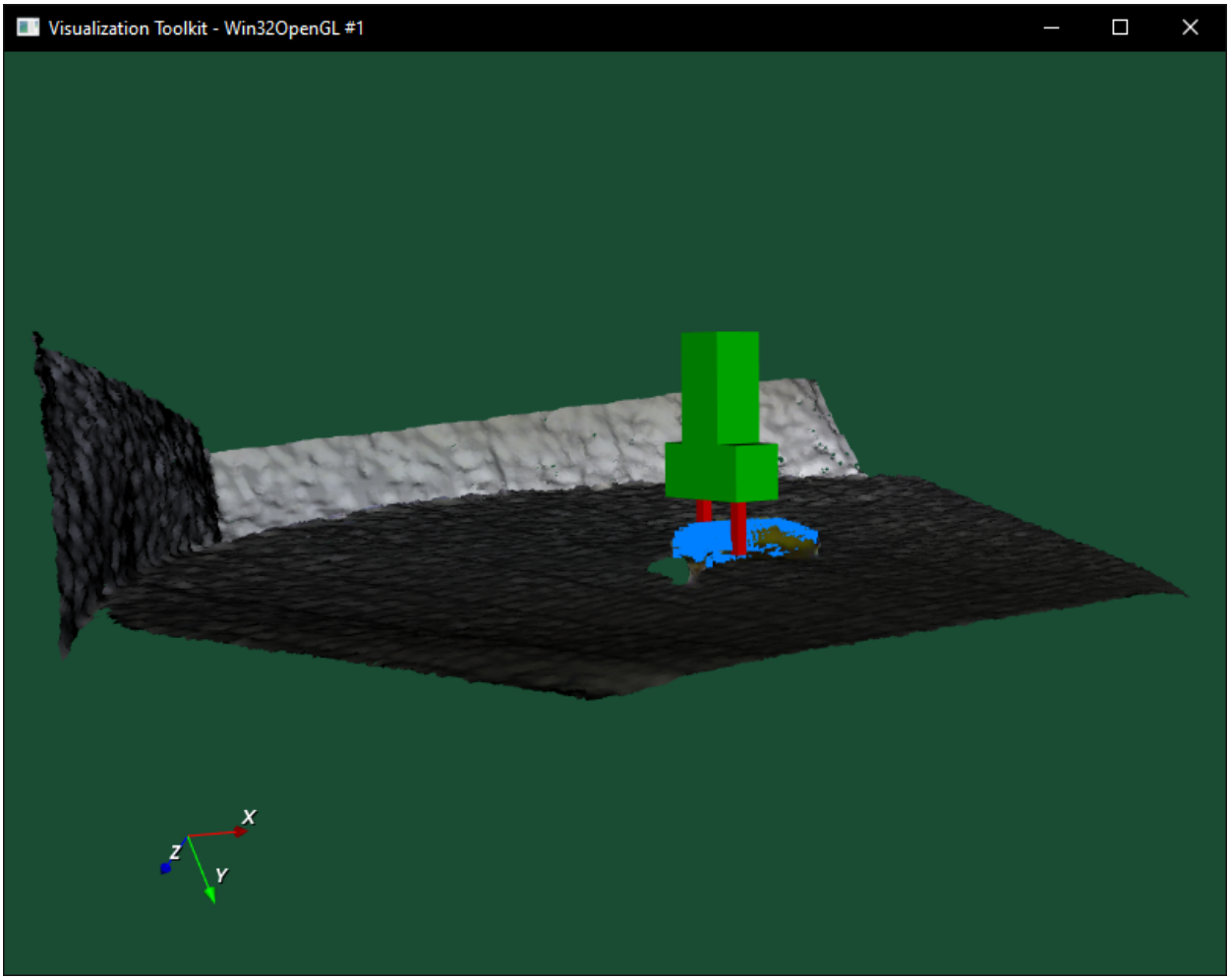


Slika 4.1 *RGB banana.*

Najprije se ova slika banane, odnosno tt_bnn_veg1.png, predaje YOLO mreži kako bi se dobio granični okvir banane te se zanemarili ostali elementi na slici, poput stola, pozadine, itd. U ovom slučaju na slici je detektirana samo banana. Granični okvir ove slike će se zapisati u bbox.txt datoteku zajedno sa svim graničnim okvirima koji su detektirani na ostalim slikama. Nakon toga se koordinate graničnog okvira odvajaju u posebnu datoteku koja se zove tt_bnn_veg1.bbx. Nakon

toga se u direktorij iz kojeg VolumeNet čita .ply datoteke i .bbx datoteke stavljaju tt_bnn_veg1.bbx i tt_bnn_veg1.ply te se u datoteku sceneSequence.txt stavi naziv tt_bnn_veg1.ply kako bi VolumeNet znao da treba uzeti u obzir točno taj .ply i njegov odgovarajući .bbx. Kada se slika analizira algoritmom VolumeNet dobije se slika 4.2 koja prikazuje položaj banane na stolu, te položaj robotske ruke takav da uhvati zadani objekt. VolumeNet također vraća postotak detekcije objekta na slici i klasu kojoj pripada. Slika 4.3 prikazuje također isti objekt ali iz drugog kuta promatranja.

Slika 4.2 *Rezultat VolumeNet algoritma za tt_bnn_veg1.ply objekt.*



Slika 4.3 Rezultat VolumeNet algoritma *za tt_bnn_veg1.ply objekt iz drugog kuta gledanja.*

4.1. Usporedba Yolo algoritma detekcije i VolumeNet-a

Kroz oba algoritma je proveden jednak broj slika i to njih 1107. Od njih 1107 bilo je 128 jabuka, 73 banane, 262 boce, 68 zdjela, 154 automobila, 50 krofni, 372 čaša. Ako neki od objekta nije detektiran označen je kao *nedetektiran (undetected)*, a ako je objekt detektiran na slici, ali ne pripada nijednoj od navedenih klasa, onda je označen kao *ostalo (other)*. U tablici 4.1 i tablici 4.2 mogu se vidjeti rezultati klasifikacije objekata primjenom ova dva algoritma. U tablici 4.1 nalaze se rezultati objekata detektiranih koristeći YOLO mrežu, dok se u tablici 4.2 nalaze rezultati klasifikacije pomoću VolumeNet algoritma.

Tablica 4.1 *Rezultati YOLO mreže.*

	Jabuka	Banana	Boca	Zdjela	Automobil	Čaša	Krafna	Ostalo	Ne detektirano	SUM	%
Jabuka	111	0	0	0	0	0	0	2	15	128	86.72
Banana	0	69	0	0	0	0	0	0	4	73	94.52
Boca	0	0	112	0	1	0	0	51	98	262	42.75
Zdjela	0	0	0	41	0	7	0	3	17	68	60.29
Automobil	0	0	0	0	71	0	0	48	35	154	46.1
Čaša	0	0	0	2	0	212	0	53	105	372	56.99
Krafna	0	0	0	0	0	0	4	23	23	50	8
Sum %											56.48

Tablica 4.2 *Rezultati VolumeNet algoritma.*

	Jabuka	Banana	Boca	Zdjela	Automobil	Čaša	Krafna	Ostalo	Ne detektirano	SUM	%
Jabuka	71	0	0	30	0	7	1	2	17	128	55.47
Banana	0	66	1	2	0	0	0	0	4	73	90.41
Boca	0	5	123	0	2	0	1	33	98	262	46.95
Zdjela	0	2	0	50	2	0	0	3	11	68	73.53
Automobil	0	2	19	2	66	1	1	31	32	154	42.86
Čaša	0	17	1	3	4	213	1	44	89	372	57.26
Krafna	0	0	0	1	1	2	13	9	24	50	26
Sum %											56.07

Tablica 4.3 *Rezultati originalnog VolumeNet-a*

	Jabuka	Banana	Boca	Zdjela	Automobil	Krafna	Čekić	Čaša	Tetrapak	WC papir	UKUPNO
Jabuka	80	0	0	0	0	0	0	0	0	0	80
Banana	0	100	0	0	0	0	0	0	0	0	100
Boca	0	0	99	0	0	0	0	0	0	0	99
Zdjela	0	0	0	100	0	0	0	0	0	0	100
Automobil	0	0	0	0	72	0	0	0	0	0	72
Čaša	0	0	0	0	0	0	0	94	0	0	94
Krafna	0	0	0	0	0	100	0	0	0	0	100
Sum											92

4.2. Usporedba s rezultatima dobivenima na temelju originalnog VolumeNet-a

Uspoređeni su rezultati tri eksperimenta: originalni VolumeNet, YOLO i kombinacija YOLO i VolumeNet.

Rezultati eksperimenta u kojem je testiran VolumeNet su prikazani u tablici 4.3. Gledajući klasu jabuka, od 128 slika koje su provedene kroz mrežu YOLO, njih 111 je prepoznato što daje rezultat od 86,72%, dok gledajući VolumeNet koji je testiran na jednakom broju slika jabuka je prepoznao samo njih 71 što nam daje postotak prepoznavanja od 55,47%, originalan VolumeNet je prepoznao 80% jabuka. Rezultati eksperimenta VolumeNet-a su pokazali kako je na testnom skupo jabuka prepoznato njih 80% što bi značilo da u slučaju klase jabuka algoritam Yolo ima najveći postotak prepoznavanja jabuka. Gledajući ostale klase kao što su banane, zdjelice i krofne, originalni VolumeNet ih je u testiranju prepoznao u iznosu od 100%, što je daleko bolje i preciznije prepoznavanje uspoređujući kako su te klase prepoznali Yolo i VolumeNet ne gledajući klasu banana, koje su oba algoritma podjednako visoko prepoznale. Također postoje klase koje mreža YOLO nije ni trenirana da prepoznaje kao što su toaletni papir, čekić i tetrapak, koje originalni VolumeNet prepoznaje izuzetno dobro. Jedno od najnižeg postotka prepoznavanja u originalnom VolumeNet-u je automobil koji je prepoznat u postotku od 72, dok ni Yolo ni VolumeNet nisu ni blizu. Također gledajući klase koje su najlošije prepoznate testirajući Yolo i VolumeNet su klasa

krafni koje su prepoznate u postotku od 8 i 26 posto, dok je ta klasa u slučaju originalnog VolumeNet-a krafne prepoznata u 100 % slučajeva. Ukupna suma prepoznavanja klasa testirajući Yolo i VolumeNet je podjednaka, oko 56%, dok gledajući ukupnu sumu prepoznavanja originalnog VolumeNet-a daje 93.5 posto prepoznavanja, što je velika razlika u točnosti i preciznosti prepoznavanja navedenih klasa. Učinkovitost VolumeNeta, čiji su rezultati prikazani u tablici 4.2, ovisi o učinkovitosti YOLO mreže, pošto objekti koji nisu detektirani YOLO mrežom uopće ne dolaze do VolumeNeta. Nadalje, u nekim slučajevima, granični okvir koji daje YOLO mreža ne obuhvaća cijeli objekt ili obuhvaća gotovo cijelu sliku s pozadinom, što rezultira pogrešnom klasikifacijom od strane VolumeNeta, koji zahtijeva točno segmentirane objekte. Također je važno napomenuti da originalni VolumeNet koristi informacije u sceni koje nisu dostupne YOLO mreži. Naime, originalni VolumeNet se oslanja na pretpostavku da ciljni objekti leže na dominantnoj ravnoj površini, te da je ciljni objekt na nekoj slici jedini objekt postavljen na tu površinu. Zbog tih pretpostavki se ne može reći da je VolumeNet općenito bolji klasifikator od YOLO mreže.

5. ZAKLJUČAK

U ovom diplomskom radu je razvijen algoritam koji detektira objekte zadanih klasa i na njihovoj površini određuje mjesta za hvatanje. Neuronska mreža YOLO je mreža koja ima mogućnost brze detekcije objekata, ali zauzvrat postoje ograničenja kako bi se ta brzina mogla zadržati. Neka od ograničenja su da YOLO mreža ima problema s detekcijom malih objekata kao što su jato ptica, itd. Kod VolumeNet-a se koriste deskriptori koji opisuju oblik pojedini objekt. Mreža YOLO je modificirana kako bi ispisivala granične okvire, postavljeni su uvjeti za izbor jednog graničnog okvira koji će se proslijediti VolumeNetu, odnosno uvjeti koji govore što da algoritam radi ako se na jednoj slici pojavi više objekata. YOLO je predstavljen i testiran na skupu slika 3D net-a. VolumeNet je testiran na istom skupu slika koristeći granične okvire koje je YOLO označio. Potom je objašnjen tijek postupka detekcije objekata i mjesta prikladnih za hvatanje, od učitavanja svih slika kroz YOLO, preko kreiranja graničnih okvira za svaki detektirani objekt. Zatim je korišten Python kako bi se granični okviri mogli odvojiti u zasebne .bbx datoteke s istim nazivima kao nazivi slika. Na kraju su sve .bbx i .ply datoteke prošle kroz VolumeNet koji je kreirao prikladnu poziciju u kojoj se može uhvatiti određeni objekt robotskom rukom. Cijeli postupak je prikazan na jednom primjeru. U konačnici oba algoritma su pokazala približno jednake rezultate. U nekim slučajevima je YOLO mreža točnija, dok je u drugim slučajevima VolumeNet bio točniji. Također je važno naglasiti da bez uporabe YOLO mreže VolumeNet se ne bi mogao koristiti jer mu je nužno korištenje graničnih okvira.

LITERATURA

[1] https://www.fer.unizg.hr/_download/repository/01-Uvod-1s.pdf, Neuronske mreže, uvod, Prof.dr.sc Sven Lončarić, Fakultet elektrotehnike i računarstva, Zagreb

[2] Girshick, Ross & Donahue, Jeff & Darrell, Trevor & Malik, Jitendra. (2013). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 10.1109/CVPR.2014.81.

[3] Girshick, Ross. (2015). Fast r-cnn. 10.1109/ICCV.2015.169.

[4] Ren, Shaoqing & He, Kaiming & Girshick, Ross & Sun, Jian. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence. 39. 10.1109/TPAMI.2016.2577031.

[5] R. Cupec i P. Đurović, "Volume Net: Flexible Model for Shape Classes," *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2018, pp. 248-255, doi: 10.1109/ROBIO.2018.8665060.

[6] Walter Wohlkinger, Aitor Aldoma Buchaca, Radu Rusu, Markus Vincze. *"3DNet: Large-Scale Object Class Recognition from CAD Models"*. In IEEE International Conference on Robotics and Automation (ICRA), 2012.

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. „You Only Look Once: Unified, Real-Time Object Detection"

SAŽETAK

U ovom diplomskom radu opisan je sustav za detekciju i klasifikaciju objekata na RGB-D slikama snimljenim 3D kamerom, te određivanje mjesta na površini objekta prikladnih za hvatanje objekta od strane robota u svrhu robotske manipulacije. YOLO mreža koja detektira objekte na slikama integrirana je s VolumeNet algoritmom koji uz klasifikaciju objekata predlaže prikladan položaj dohvaćanja određenog objekta sa slike. YOLO mreža vraća kao izlaz granični okvir objekata zadanih klasa. Postavljeni su uvjeti za odabir odgovarajućeg graničnog okvira u slučaju da mreža na jednoj slici pronađe više objekta, tako da vraća samo jedan objekt očekivane veličine koja se nalazi blizu središta slike. Podatak o poziciji ciljnog objekta na slici dobiven YOLO mrežom predstavlja ulaz u VolumeNet.

Ključne riječi: YOLO, VolumeNet, neuronske mreže, konvolucijske neuronske mreže, klasifikacija objekta, detekcija objekta, robotski vid, RGB-D slike

ABSTRACT

Object recognition and detection of their parts on complex scenes

This thesis describes a system for detecting and classifying objects on RGB-D images captured by a 3D camera and determining places on the surface of the object suitable for grasping the object by robots for robotic manipulation. The YOLO network that detects objects in images is integrated with the VolumeNet algorithm, which, along with object classification, returns an appropriate position for grasping a target object. The YOLO network returns the bounding box of instances of given classes as output. In the cases where the network detects more than one object, only one object of the expected size that is close to the centre of the image is considered. The position data of the target object in the image obtained by the YOLO network represents the input to the VolumeNet.

Keywords: YOLO, VolumeNet, neural networks, convolutional neural networks, object classification, object detection, robotic vision, RGB-D images

ŽIVOTOPIS

Hrvoje Varga, rođen 29.siječnja 1997. godine u Osijeku, Pohađao osnovnu školu u Bilju, te nakon završetka osnovne škole pohađao Prirodoslovno-matematičku gimnaziju u Osijeku. Nakon završetka srednje škole, 2015. godine upisuje Fakultet elektrotehnike računarstva i informacijskih tehnologija u Osijeku, preddiplomski studij Računarstva. Nakon završetka preddiplomskog studija 2018. godine, upisuje diplomski studije Automobilsko računarstvo i komunikacijske tehnologije.