SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

ESTIMACIJA KINEMATIČKIH PARAMETARA OBJEKATA S POKRETNIM DIJELOVIMA

Diplomski rad

Dijana Ivezić

Osijek, 2022.

SADRŽAJ

1. UVOD

Robotski sustavi su područje široke primjene, od mobilnih robota do dronova. Pomažu čovjeku obavljati zamorne i repetitivne zadatke kao i one koji su opasni za ljudsko zdravlje. Primjenjuju se u mnogim područjima ljudskog djelovanja od kojih je najpoznatija primjena u industriji u obliku robotskih manipulatora. Lako možemo zamisliti velika postrojenja sa grupom robota koji svaki obavlja svoj dio posla koji je dio nekog proizvodnog procesa. Masa velikih i glomaznih dijelova, zamor i visoka preciznost dobro dizajniranim robotskim sustavima ne predstavljaju prepreku.

Često očekujemo od robota da manipulira svojom okolinom, da pouzdano i precizno izvršava svoje zadatke prilikom čega je u interakciji sa različitim tipovima objekata. Robot koji primjerice kuha mora pravilno prepoznati relevantne objekte na sceni, kao što su štednjak, hladnjak, mikrovalna pećnica, kojekakvo posuđe i dr. Osim detekcije objekata na sceni što je zadatak sam po sebi, robot mora ispravno uhvatiti ili manipulirati objektom ovisno o njegovoj klasi. Ako robot nema podatke o kakvom objektu se radi, kojim njegovim dijelom treba manipulirati i gdje točno treba uhvatiti objekt, tada sigurno neće biti u mogućnosti izvršiti zadatke kao što su otvaranje ladica, korištenje alata i sl. što znači da je cijeli sustav sa takvim robotom zakazao.

Nakon detekcije objekta na sceni korisno je segmentirati objekt na dijelove kako bi sustav imao informaciju kojim dijelovima je smisleno manipulirati, a koje dijelove možda mora i izbjegavati, kao primjerice užareni dio štednjaka. U ovoj priči o manipulaciji objektima već je zadan velik broj zadataka koje inženjeri i znanstvenici rješavaju određenim algoritmima i niti jedan od tih zadataka nije lak posao. Dodatno se javljaju problemi ukoliko objekti imaju pomične dijelove kao što su škare, ladice, naočale i sl. gdje je osim detekcije objekta, klasifikacije i segmentacije nužno odrediti i parametre pomičnih dijelova kako bi sustav njima ispravno upravljao.

Na primjeru ladica, ako je robotu zadatak spremiti objekt u ormar s ladicama, kao prvu informaciju mora znati gdje se na sceni nalazi ladica, a nakon toga je li ta ladica uopće već otvorena i ako je za koliko je otvorena, možda ju je potrebno još otvoriti kako bi traženi objekt mogao proći kroz otvor u ladicu.

Ovaj diplomski rad bazira se na metodi iz [1] koja je detaljnije objašnjena u potpoglavlju 3.6. U drugom će poglavlju biti će opisane metode koje se bave objektima s pokretnim dijelovima i imaju gotovo jednak cilj kao [1] kako bi u tom pregledu bile predstavljene sličnosti i razlike između metoda kao i dosezi na ovom području unazad maksimalno nekoliko godina. Cilj koji se pokušava postići je uspješna detekcija objekata s pokretnim dijelovima na sceni, segmentacija na dijelove i računanje kinematičkih parametara objekta kako bi se njime u nekom projektu upravljalo robotskim manipulatorom. Treće poglavlje je fokusirano na teorijsku podlogu korištenih algoritama i metoda iz originalnih znanstvenih radova gdje je cilj dati važne informacije za razumijevanje konačne metode za estimaciju kinematičkih parametara objekata. Četvrto i peto poglavlje se bave programskom podrškom i dobivenim rezultatima, a posljednje poglavlje je zaključak i osvrt na ovaj rad.

Zadatak ovog rada je teorijski opisati važne teme koje su potrebne za shvaćanje glavne metode iz [1], opisati programski kod i prikazati postignute rezultate kojima se ispituje učinkovitost metode.

2. PREGLED PODRUČJA TEME

Slično kao [1], metoda iz [2] koristi PointNet++ mrežu [3] kao temelj, no po svemu ostalom se razlikuje. Prednost FlowBot3D metode [2] je što ne koristi već gotove CAD modele objekata, ali ni ne ograničava korištenje objekata s pomičnim dijelovima na instance već definirane kategorije. Eisner et al. [2] ne estimiraju kinematičke parametre objekta, već imaju drugačiji pristup od [1], ali uspješno robotskim manipulatorom sa vakumskom čašicom kao alatom ispravno pomiču objekte kao što su vrata i sl. Svoj rad [2] temelje na vektorima koje pomoću njihove mreže estimiraju za svaku točku ulaznog oblaka točaka. Svaki taj vektor predstavlja smjer gibanja ukoliko se na pokretni dio primjeni sila u pozitivnom smjeru s obzirom na takav objekt. Osim što primjena FlowBot3D metode ne ovisi o poznatom modelu ili kategoriji objekta, ne ovisi niti o skali ili translaciji unutar prostora. Ova metoda je ograničena na rotacijske ili translacijske zglobove s jednim stupnjem slobode s tim da se jedan dio objekta smatra nepomičnim s obzirom na drugi (npr. ormar s ladicama gdje se ladice pomiču ali je ormar uvijek fiksan). U pokusima koriste robotski manipulator sa sedam stupnjeva slobode, a kao senzor koriste RBG-D kameru kojim dobivaju ulazni oblak točaka. Algoritam se sastoji od dva dijela: faza prije kontakta i faza nakon kontakta. Prije kontakta robot na temelju ulaznog oblaka točaka računa opisane vektore za svaku točku koristeći predloženu mrežu, od svih vektora odabere onog najvećeg iznosa i u toj točki se pričvrsti za objekt. Nakon kontakta ciklički računa vektore, pomiče objekt za najveći iznos od izračunatih vektora, ponovno računa i pomiče objekt dok objekt nije potpuno pomaknut (npr. vrata su zatvorena). [2]

Vrlo slično metodi iz [1], Liu et al. [4] koriste PointNet++ mrežu [3] i NOCS metodu [5] (potpoglavlje 3.5.) kako bi estimirali kinematičke parametre objekata. Za razliku od [1], nije potrebno znati kojoj kategoriji objekt pripada, stoga nije važno znati ni točan kinematički lanac za objekt na sceni. Koriste dva PointNet++ modula, prvi segmentira objekt na dijelove i za svaki taj dio određuje NOCS, drugi modul klasificira o kojoj vrsti zgloba se radi i određuje parametre za nj.

Još jedna metoda koja koristi PointNet++ mrežu [3] je OMAD [6] koja na temelju globalnog deskriptora za cijeli objekt računa podatke u tri grane. Prvom granom se dobiju važne točke u ulaznom oblaku točaka (značajke) i parametri zglobova u normaliziranom prostoru, drugom stupanj zakreta pomičnog dijela u prostoru kamere, a trećom korespodentne značajke pomoću kojih se vrši optimizacija kako bi se poboljšali dobiveni kinematički parametri zglobova.

Navedene tri grane čine OMADNet [6]. Postoji podmreža za svaku kategoriju objekata, istraživači [6] svojom mrežom OMAD-PriorNet računaju parametre specifične za određenu kategoriju i zajednički je dio prilikom daljnjeg računanja. Ti parametri se koriste prilikom aproksimacije dvaju funkcija koje nazivaju funkcija oblika i funkcija zgloba čijom kombinacijom dobivaju funkciju deformacije. Funkcijom oblika dobivaju oblak značajki u normaliziranom prostoru, a funkcijom zgloba dobivaju parametre zgloba u normaliziranom prostoru. Deformacijskom funkcijom točke prelaze iz normaliziranog prostora u prostor kamere. Istraživači [6] minimiziraju razliku između izračunatih značajki u prostoru kamere na temelju globalnog deskriptora i dobivenih značajki u prostoru kamere pomoću estimirane deformacijske funkcije.

Osim do sada opisanih metoda, postoje metode koje uspješno manipuliraju objektima sa pomičnim dijelovima na drugačiji način. [7] i [8] pomoću agenta iterativno segmentiraju objekt na pomične dijelove ili estimiraju kinematičke parametre objekta prilikom interakcije s istima i snimajući promijenjenu scenu što omogućava generalizaciju na neviđene objekte.

3. TEORIJSKA PODLOGA

3.1. Relevantni pojmovi i algoritmi iz računalne geometrije

3.1.1. Oblak točaka

Učestali prikaz objekata iz okoline je slika na kojoj najčešće kombinacijom crvene, zelene i plave boje u pojedinom pikselu (RGB slika) opisujemo scenu na ljudima jasan i intuitivan način. Takvi podaci su poredani u već spomenute piksele gdje je svaki jednakih dimenzija i poredani su međusobno u rešetku (matrica piksela). Slike su rasprostranjeni medij s kojima se čovjek svakodnevno susreće, no takvi podaci gube vrijednu informaciju o dubini, točnije nepoznate su 3D koordinate točaka od interesa. Za određene primjene slika je idealan alat, kao primjerice ilustracije, prikaz raznih događaja i sl., dok za druge nije dovoljno informativna. Za ovaj rad važne su informacije o veličini i međusobnoj interakciji objekata, dijelova objekata i agenta (robota) što iz samo RGB slika nije moguće precizno odrediti. Oblaci točaka su trodimenzionalni prikaz, za razliku od slike koja je dvodimenzionalna, i sadrži spomenute važne informacije. S oblacima točaka je nerijetko zahtjevnije računati nego sa slikama i za slike postoji više pouzdanih algoritama za detekciju i segmentaciju objekata što je bitno za ovaj rad. Na jednostavnom primjeru biti će objašnjeno zašto je korisna informacija o dubini.

Zdrav čovjek može odrediti udaljenost nekog objekta i istim manipulirati, primjerice uhvatiti i podići čašu, baciti lopticu i sl. Takve radnje su znatno otežane bez percepcije dubine, osoba bi rukom tražila lopticu polagano joj se približavajući jer ne može ocijeniti dubinu. Moguće je provesti jednostavan eksperiment. Ukoliko osoba uzme bilo kakvu kameru koja u gotovo stvarnom vremenu može prikazati snimljenu sliku (npr. kamera na većini modernih mobilnih uređaja) i gleda isključivo taj prikaz dok pokušava uhvatiti objekt, može se primijetiti kako je uhvatiti objekt nešto otežanije, dobrim dijelom zbog teže percepcije dubine. Algoritmi ovoga rada svoje rezultate postižu nad oblacima točaka, stoga će ovdje biti opisano što su oblaci točaka i na koji način se takvi podaci prikupljaju.

Oblak točaka je neorganizirani skup točaka predstavljenih sa svoje tri prostorne koordinate kojima se prikazuje bilo koji 3D objekt dohvaćen sa scene u stvarnosti ili simulirane scene. Kamere dohvaćaju slike na temelju svjetlosnih zraka koje se prolazeći kroz leću fokusiraju na senzor (CCD ili CMOS senzor). Oblaci točaka se najčešće dohvaćaju pomoću LIDAR senzora. LIDAR tehnologija se također bazira na svjetlosti, no umjesto korištenja svjetla iz scene, spomenuti senzor

odašilje svjetlosne zrake u snopu te mjeri vrijeme koje je potrebno svakoj zraci da se vrati natrag na temelju kojeg se računa udaljenost od senzora. Ponavlja se isti postupak rotirajući glavu senzora i ponovnim odašiljanjem snopa svjetlosti.

3.1.2. Voronoi dijagram

Voronoi dijagrami su geometrijska struktura često korištena u mnogim znanstvenim granama kao što su računalna znanost, kemija, biologija, fizika i dr. Prvo će biti opisani navedeni dijagrami u dvodimenzionalnom prostoru i njihova svojstva, a nakon toga će biti prikazani u trodimenzionalnom prostoru što odgovara podacima koji se koriste u ovom radu (oblaci točaka).

Jednostavan primjer primjene Voronoi dijagrama bio bi raspored učenika određenoga grada u najbliže im škole. Ukoliko u gradu postoji više škola zajedno označene sa S (skup škola), gdje je pojedina škola označena sa točkom p i koordinatama (p_1, p_2), a učenikova kuća točkom x kao (x_1, x_2). Zadatak je za svaki x pronaći najbližu školu p iz S. Metrika blizine će biti euklidska udaljenost između p i x prikazana jednadžbom (3-1) [9].

$$d(p,x) = \sqrt{(p_1 - x_1)^2 + (p_2 - x_2)^2} \tag{3-1}$$

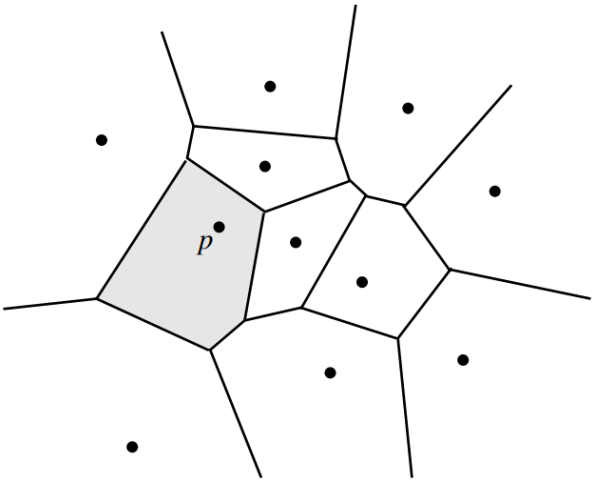
Uzevši u obzir dvije škole, p i q, postoji skup točaka $B(p,q)$ što je simetrala na poveznicu dvije škole takva da bi učeniku bilo svejedno u koju školu polazi da se nalazi na nj jer je točno na pola puta između te dvije škole. Takva simetrala dijeli ravninu (zamišljenu kartu) na dvije poluravnine i u ovisnosti u kojoj poluravnini se učenik nalazi, ta škola mu je bliža. U poluravnini $D(p,q)$ se nalaze oni učenici x kojima je škola p bliža nego škola q. Za fiksnu školu p i svaku pojedinu drugu školu q iz S mogu se odrediti najbliži učenici kao presjek svih tako dobivenih poluravnina koji se označava kao $VR(p,S)$ i predstavlja dio ravnine oko škole p u kojoj su svi učenici najbliži toj školi (jednadžba (3-2) [9]).

$$VR(p,S) = \bigcap_{q \in S, q \neq p} D(p,q) \tag{3-2}$$

Drugim riječima, $VR(p,S)$ predstavlja Voronoi područje oko točke p iz S u kojemu je svaka točka iz R^2 (za dvodimenzionalni slučaj) najbliža i predstavlja konveksni poligon koji može biti omeđen ili neomeđen. Više područja nad skupom S je disjunktno, dijele ih Voronoi rubovi koji su zajednički dio dvaju ili više područja. Svaki rub čini podskup točaka iz $B(p,q)$, a unija svih rubova između područja se naziva Voronoi dijagram što je, matematički, planarni graf čiji su

vrhovi krajnje točke rubova koji imaju stupanj od 3 ili više što znači da je vrh krajnja točka 3 ili više rubova.

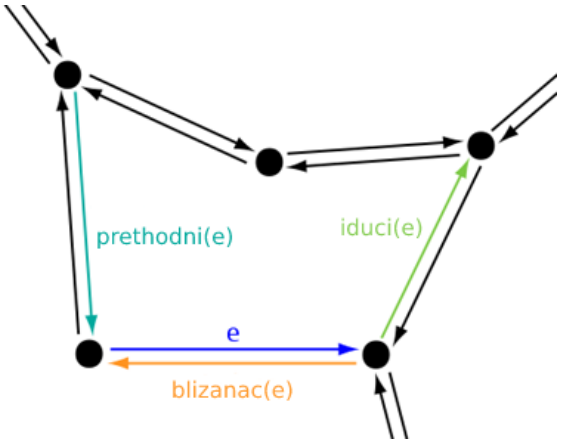
Kao što je intuitivno opisano u [9], u ravnini je odabrana bilo koja točka x i oko nje se širi prsten čiji se radijus povećava počevši od 0 dokle god ne naiđe na jednu ili više točaka iz S. Ako naiđe na točno jednu točku, tada je x dio Voronoi područja i prsten je naišao na pripadajuću točku p iz S tome području. Ukoliko naiđe na dvije točke istovremeno, tada je x dio ruba (pripada $B(p, q)$), a ako naiđe na više točaka tada je x vrh Voronoi dijagrama (slika 3.1.). Voronoi dijagram predstavlja particiju ravnine.



Slika 3.1.: Voronoi dijagram [9]

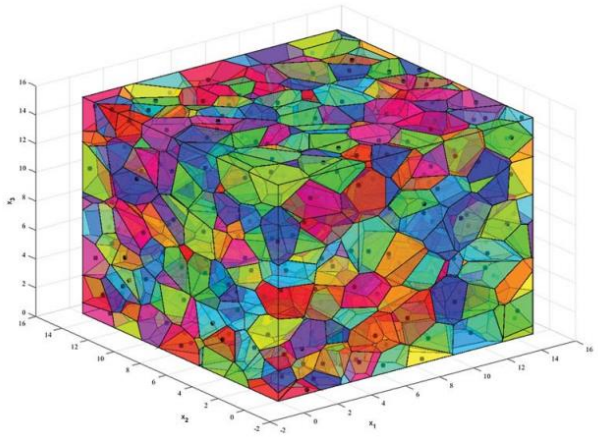
Ovakav graf povezuje točke koje su najviše udaljene od svih p iz S što je korisno kod npr. planiranja putanje mobilnoga robota. Ako su p prepreke, tada je smisleno programirati robota tako da se prostorom giba uzduž Voronoi dijagrama, jer je tako najviše udaljen od svih prepreka, čime se na najjednostavniji način izbjegava kolizija.

Najčešće korištena struktura podataka za rad sa Voronoi dijagramima je lista dvostruko povezanih rubova gdje se svaki rub promatra kao dva poluruba (vektor čija orijentacija ovisi koje područje se opisuje, dva blizanca suprotne orijentacije). Za svaki polurub se zapisuju njegova dva vrha, referenca na idući i prethodni polururub i njegov blizanac (slika 3.2.).



Slika 3.2.: Dvostruko povezani rubovi dijagrama [10]

Slično kao što su definirani Voronoi dijagrami u R^2 definirani su za R^n, s tim da su za ovaj rad značajni Voronoi dijagrami u R^3. Za trodimenzionalni slučaj, $B(p, q)$ postaje ravnina okomita na dužinu koja spaja dvije točke u prostoru i nalazi se točno u polovištu te dužine. Područja postaju poliedri omeđeni plohama koje su definirane svojim rubovima na čijim krajevima se nalaze vrhovi. Na slici 3.3. crno obojane točke se mogu promatrati kao oblak točaka, a šareno su poliedri Voronoi dijagrama. Svako područje sa nekim drugim može dijeliti zajednički vrh, rub ili plohu.



Slika 3.3.: Trodimenzionalni Voronoi dijagram [11]

3.1.3. Uzorkovanje prema najudaljenijoj točki

Za ovaj rad korisni su opisani Voronoi dijagrami (potpoglavlje 3.1.2.) kako bi uspješno bila poboljšana ideja PointNet mreže što će biti opisano u potpoglavlju 3.4.2. Glavno poboljšanje leži u činjenici da oblaci točaka najčešće nisu jednake gustoće u cijelom prostoru kojeg zauzimaju, što se može riješiti tako da se PointNet mreža primjenjuje u lokalnim podskupovima ulaznog skupa točaka oko izračunatih točki od interesa. Te točke od interesa su gušće raspoređene na primjerice

rubovima objekata, a rjeđe u dijelovima objekata bez puno detalja. U ovom potpoglavlju će biti opisan algoritam uzorkovanja tih točaka od interesa.

Uzorkovanje oblaka točaka u ovisnosti o samim podacima se postiže algoritmom uzorkovanja prema najudaljenijoj točki (engl. *farthest point sampling,* FPS) kojeg su detaljno opisali Eldar et al. u svom originalnom radu u kontekstu uzorkovanja slika, no ideja se može proširiti i na oblake točaka. [12]

Autori [12] predlažu neravnomjerno uzorkovanje slike gdje se kao idući uzorak uzima onaj koji je najviše udaljen od ostalih zbog toga što su pokazali kako je na takav način rekonstrukcijska pogreška minimalna (srednja kvadratna pogreška). FPS algoritam se oslanja na Voronoi dijagram, gdje je svaki uzorak točka u dijagramu sa svojim područjem čime je slika segmentirana na konveksne poligone. Točke koje su najviše udaljene od ostalih uzoraka su u ovom kontekstu vrhovi Voronoi dijagrama. Algoritam uzorkuje točku po točku počevši od prvog uzorka na slici koji sa uglovima slike tvori skup S. Nad tim skupom se tvori Voronoi dijagram na temelju čijih vrhova se traži točka najviše udaljena od točaka iz S. Ta točka se dodaje u skup S i algoritam staje ukoliko nije potrebno više uzoraka, ukoliko je onda se ponovno traži najudaljeniji vrh iz dijagrama i nastavlja se kako je opisano. [12]

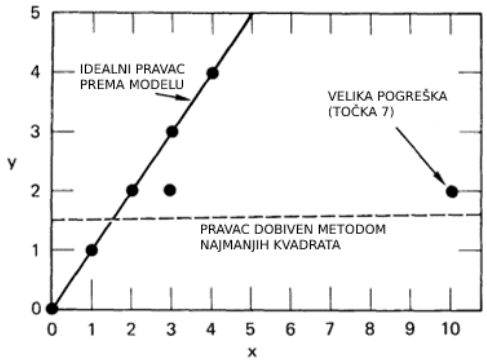
Za sada algoritam uzorkuje sliku bez obzira na svojstva pojedine slike što nije krajnji rezultat koji je predstavljen u prvom odlomku. Naime, nije uzeta u obzir željena promjenjiva gustoća uzorkovanja. Istraživači u istom radu [12] predstavljaju poboljšanje svoje metode gdje koriste otežane euklidske udaljenosti prilikom odabiranja najudaljenijeg vrha u ovisnosti o potrebnoj gustoći lokalnih uzoraka tako da ta nova udaljenost bude veća na područjima sa više detalja nego odgovarajuća euklidska udaljenost (bez koeficijenta) na monotonijim područjima.

FPS se može na sličan način u dorađenom obliku primijeniti i na oblake točaka što je detaljnije opisano u [13].

3.2. RANSAC algoritam

RANSAC (engl. *RANdom SAmple Consensus*) [14] je algoritam kojim se teorijski model preslikava na podatke dohvaćene iz stvarnoga svijeta koji su podložni pogreškama. Autori [14] daju primjer estimacije pravca koji najbolje odgovara danim podacima. Mnogi algoritmi, kao što je npr. metoda najmanjih kvadrata, pretpostavljaju da podataka ima dovoljno i da pogreške nisu

tolike da izglađivanje odstupanja velikim brojem točnih podataka ne igra značajnu ulogu. Svoj primjer su prikazali slikom koja se može vidjeti na slici 3.4.



TOČKA	x	y
1	0	0
2	1	1
3	2	2
4	3	2
5	3	3
6	4	4
7	10	2

Slika 3.4.: Loša estimacija pravca za prikazane podatke [14]

U nekim slučajevima, većina podataka može dobro pratiti matematički model, no uslijed različitih pogrešaka prilikom sakupljanja podataka može doći do velikog odstupanja manjeg broja podatka od stvarne vrijednosti što ugrožava ispravnu estimaciju nad podacima (slika 3.4.). Prvih šest točaka dovoljno dobro prate model i na temelju njih se može dobiti traženi pravac, u slučaju računanja pomoću metode najmanjih kvadrata, sedma točka previše odstupa od ostalih, ta točka je pogrešna i ne bi se smjela uzeti u obzir prilikom računanja. Puna linija predstavlja optimalan pravac, a crtana linija predstavlja estimirani pravac na kojega je previše utjecao pogrešni podatak. Fischler i Bolles [14] predstavljaju RANSAC algoritam koji prilikom računanja odstupanja podataka od matematičkog modela ne uzima u obzir pogrešne podatke (engl. *outliers*).

Algoritam opisuju [14] kako slijedi: za neki matematički model M je potrebno odrediti najmanji broj točaka (najmanje podataka) n kojima se jednoznačno određuju parametri toga modela iz podataka. Nasumično se odabire n točaka iz ulaznog skupa podataka (skup $S1$) na temelju kojih se računaju parametri modela $M1$ pomoću kojega se odabiru točke iz ulaznog skupa podataka koje odgovaraju $M1$ unutar neke zadane tolerancije, skup tih točaka se naziva $S1*$. Ukoliko točaka u $S1*$ ima manje od neke definirane granice t, tada se ponovno nasumično odabire n točaka (ponovno se odabire skup točaka $S1$) i ponovno se računa $M1$. U suprotnome, pomoću točaka iz $S1$ se poboljšava model $M1$ uz korištenje već postojećih algoritama, npr. metoda najmanjih kvadrata. U ovom slučaju metoda najmanjih kvadrata ne računa sa pogrešnim podacima, već sa podacima koji već otprilike odgovaraju modelu, stoga je korišten za fino primicanje optimalnom modelu. Poboljšani model se naziva $M1*$ kojim se može odabrati još točaka koje odgovaraju tom modelu unutar određene tolerancije pomoću kojih se dodatno može poboljšati

model. Algoritam u slučaju nepronalaženja modela $M1$ se ponavlja k puta, a taj parametar i t parametar se računaju kako je objašnjeno u originalnom radu [14].

3.3. Umeyama algoritam

U metodama ovoga rada korišten je Umeyama algoritam [15] pomoću kojega se računa rotacijska matrica, skala i translacijski vektor između dva oblaka točaka. Algoritmom se računaju navedeni parametri tako da srednja kvadratna pogreška između odredišnog oblaka točaka i polaznog oblaka točaka transformiran navedenim transformacijama bude minimalna. Prema njihovom teoremu, ta se minimalna vrijednost može dobiti pomoću rastavljanja matrice koja predstavlja umnožak matrice početnog i transponirane matrice krajnjeg skupa točaka na singularne vrijednosti. Zbog toga teorema, Umeyama algoritam nije iterativan, već je dano analitičko rješenje kao što je opisano u originalnom radu [15]. Za ovaj diplomski rad je korisna kombinacija RANSAC [14] i Umeyama algoritama gdje se pomoću RANSAC-a uklanjaju netočni podaci (engl. *outliers*), a Umeyama algoritmom se računaju optimalni transformacijski parametri.

3.4. PointNet++

3.4.1. PointNet

PointNet++ [3] mreža bazirana je na PointNet [16] mreži, stoga će PointNet mreža biti prva opisana kako bi bolje bila predstavljena sama ideja mreža te kako bi bilo jasnije zašto unaprijeđena verzija mreže daje bolje rezultate na postavljenom problemu ovog rada.

Prvi korak estimacije kinematičkih parametara objekta je iz oblaka točaka jednog pogleda scene detektirati o kojem se objektu radi i segmentirati objekt na dijelove, a te informacije se prosljeđuju na daljnju obradu. Mreža koja uspješno odrađuje taj korak zove se PointNet autora Qi et al. [16]. Ulazni prostor navedene mreže je neorganiziran skup točaka gdje bilo koja permutacija ulaznog skupa daje jednak model objekta.

Problemi klasifikacije često se rješavaju primjenom konvolucijskih neuronskih mreža, jednostavan primjer bi bila detekcija broja na slici. U takvim mrežama ulaz je tenzor koji predstavlja sliku i može imati jedan ili više slojeva veličine slike ovisno je li slika u boji i kako je ta boja kodirana. Konvolucijske neuronske mreže se sastoje od filtera određenih dimenzija koji se primjenjuju na dijelove slike i tako detektiraju značajke iz tih lokalnih receptivnih polja koja su

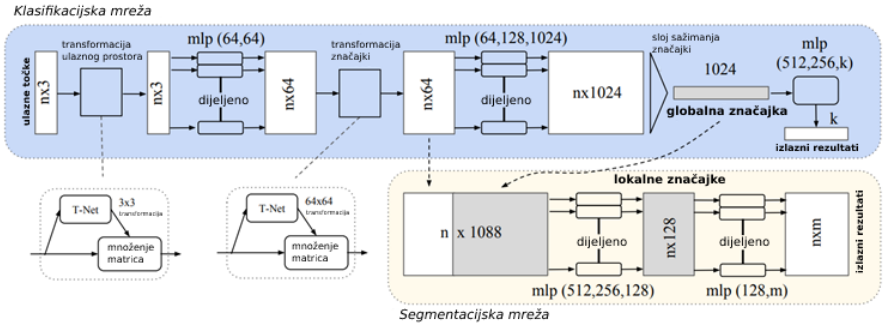
omeđena veličinom filtera. Od važnog značaja je što su na slikama pikseli poredani i ima smisla primijeniti konvolucijski sloj na dijelove slike koji predstavljaju neke lokalne značajke koje mreža uči prepoznati jer se čuvaju veze između piksela. Poredani su u diskretne rešetke gdje su pikseli uvijek jednake gustoće. Međusobni odnosi pojedinih piksela promatranog segmenta, kao i odnosi segmenata su očuvani. Treniranoj mreži kada se na ulaz dovede druga instanca brojke, podešeni filteri mogu pronaći odgovarajuće značajke negdje na slici, lokalno su pikseli vrlo slično poredani. Sami konvolucijski slojevi nisu invarijantni na rotaciju ulaznih podataka, stoga se traže parametri filtera za svaku od rotacija kojima je mreža bila izložena tijekom treniranja nakon čega se kroz sažimanje značajki na temelju maksimuma izlaza iz svih filtera omogućava invarijantnost na translaciju i rotaciju kao i smanjenje dimenzionalnosti podataka koji se dalje propagiraju kroz mrežu. Konvolucijske neuronske mreže su vrlo dobro razrađene u literaturi i za mnoge primjene postižu rezultate gotovo najvećeg dosega.

Oblaci točaka nemaju redoslijed i nisu poredani ravnomjerno u rešetku stoga nije korisno direktno na takvom ulaznom skupu primijeniti konvoluciju. Neke od metoda oblak točaka transformiraju u diskretne rešetke u obliku voxela nad kojima primjenjuju konvolucijske neuronske mreže slično kao sa slikama, ali takav prikaz informacija zahtijeva više računalne memorije i može se dogoditi gubitak podataka uslijed kvantizacije. Prostorna i vremenska kompleksnost takvih rješenja raste sa kubom ulazne rezolucije podataka. [16]

Istraživači [16] predlažu drugačiji pristup rješavanju problema u njihovoj PointNet mreži. Glavna ideja je oblak točaka na ulazu transformirati u standardizirani oblik primjenom rotacija i/ili translacija za svaku točku ulaznog oblaka točaka. Ukoliko je ulazni prostor u obliku mreže točaka, tada se uzorkuje određeni broj točaka sa lica poligona a dobiveni oblak točaka se normira na jediničnu sferu. Svaka točka je najčešće prikazana sa svoje tri koordinate u prostoru. PointNet uči prepoznati kritične točke za svaku klasu ulaznih podataka kao i kodiranje oblika te klase u vektor određene veličine (u radu je uzeto 1024 [16]) koji se naziva globalni deskriptor za tu klasu. Tako dobiveni vektor se koristi za klasifikaciju ili segmentaciju objekata.

Arhitektura mreže se može vidjeti na slici 3.5. i sastoji se od dva dijela: klasifikacijskog dijela mreže (sa plavom pozadinom) i segmentacijskog dijela mreže (sa žutom pozadinom) s tim da segmentacijski dio mreže koristi dobar dio arhitekture klasifikacijskog dijela, značajke se udružuju kako bi uspješno segmentirali ulazni oblak točaka. Na lijevom rubu počinje mreža gdje je označeno da se ulazni prostor sastoji od n točaka koje imaju tri dimenzije (koordinate u euklidskom prostoru). Kako bi mreža imala bolju robusnost na rotaciju i translaciju točaka na

ulazu, svaku točku transformiramo naučenom transformacijom koja je označena na slici kao „transformacija ulaznog prostora“ što predstavlja mrežu koja uči transformacijsku matricu na temelju ulaznih podataka tijekom treniranja. Autori [16] tu mrežu nazivaju „mini-mreža“ i koristi se za transformiranje ulaznih podataka kao i naučenih značajki iz svake točke, „mini-mreža“ će biti objašnjena u narednim odlomcima. Pošto se ista transformacijska matrica koristi za svaku točku zasebno (a ta transformacija je ortogonalna matrica) čuvaju se međusobni odnosi u euklidskom prostoru, a redoslijed kojim transformiramo točke nije važan što je korisno u ovoj primjeni jer ulazni prostor nije sortiran niti ima strukturu. Nakon što su ulazni podaci dovedeni u neutralan položaj, pomoću potpuno povezane unaprijedne mreže (engl. *„multilayer perceptron“*, MLP), čiji se naučeni parametri koriste za svaku točku ulaznog prostora, se dobivaju lokalne značajke ulaznih podataka. Nakon transformacije „mini-mrežom“ lokalnih značajki slijedi ponovno MLP pod-mreža. Teorijski, kao i svaki MLP, ta mreža uči aproksimirati određenu neprekidnu funkciju o čemu će također biti govora u narednim odlomcima. Iz novih dobivenih značajki tražimo jednu značajku koja predstavlja globalni deskriptor za oblik pomoću sloja sažimanja značajki (engl. *„max pooling layer“*) tako da od svih n značajki odabiremo informacije najvećeg iznosa duž vektora veličine K (na slici je 1024). Na takav način dobiva se vektor duljine K što je tzv. „bottleneck“ dimenzija mreže što će biti naknadno objašnjeno. Globalne deskriptore je sada problem klasificirati. Da se uočiti da umjesto ulaznog oblaka točaka imamo njegovu reprezentaciju koja ne ovisi o poretku točaka na ulazu. Klasifikacija se vrši pomoću još jedne MLP pod-mreže koja na kraju kao izlaz daje vektor duljine k koji u sebi sadrži vjerojatnosti pripadnosti određenoj klasi.



Slika 3.5.: Arhitektura PointNet mreže [16]

U slučaju segmentacijske mreže, konkatenacijom globalnog deskriptora svakoj lokalnoj značajki učimo združene značajke smanjenih dimenzija pomoću MLP-a kako bi segmentirali

ulazni oblak točaka. U ovom koraku se vidi na koji način segmentacijska mreža koristi dobar dio arhitekture klasifikacijske mreže. Nakon ponovnog prolaska kroz MLP dimenzija izlaznog prostora predstavlja za svaku točku n kojem dijelu m pripada u obliku vektora vjerojatnosti pripadnosti.

Transformacijska matrica se aproksimira pomoću spomenute „mini-mreže“ kako bi ulaz doveli u neutralno stanje iz kojega je smislenije tražiti značajke. Koristi se i na ulaznom prostoru podataka i na lokalnim značajkama točaka s tim da lokalne značajke posjeduju veću dimenzionalnost u svojim podacima (iz slike 3.5. se može uočiti kako ulazni prostor za svaki n sadrži informaciju dimenzionalnosti 3 (koordinate), dok značajke imaju dimenzionalnost 64). Zbog toga povećanja nailazi se na probleme optimizacije i stabilnosti tijekom učenja mreže što su autori [16] poboljšali tako što su dodali regularizacijski parametar ukupnoj funkciji troška koji također osigurava da je aproksimirana matrica ortogonalna. Detaljniji opis arhitekture „mini-mreže“ može se pronaći u originalnom radu [16] kao i korišteni regularizacijski parametri.

Qi et al. [16] daju teorijsko objašnjenje efikasnosti njihove mreže. Kao što je već opisano, MLP mreža je korištena za klasifikaciju objekata ali na vektoru globalnog deskriptora, točnije, reprezentaciji objekta. Eksperimentalno su pokazali da klasifikacijsku MLP mrežu nije moguće primijeniti direktno na ulazni oblak točaka jer to daje loše rezultate [16]. Zanimljiv doseg ovog rada [16] je sjedinjenje informacija svih točaka u jedan globalni vektor značajki (deskriptor) bez obzira na permutaciju ulaznog prostora što je postignuto simetričnom funkcijom nad cijelim skupom. Primjer takve funkcije bi bila suma svih članova skupa gdje zbog svojstva zbrajanja nije važno kojim redoslijedom zbrajamo elemente, uvijek se postigne jednak rezultat. Takvu funkciju modelira sloj sažimanja značajki koji je na slici 3.5. tako i označen. Teorijski su dokazali u svom radu [16] svojstvo aproksimacije kontinuirane funkcije nad skupom njihove mreže, preciznije, funkcije koja aproksimira Hausdorffovu udaljenost[1] između točaka dva skupa (jednadžba (3-3)) [17]).

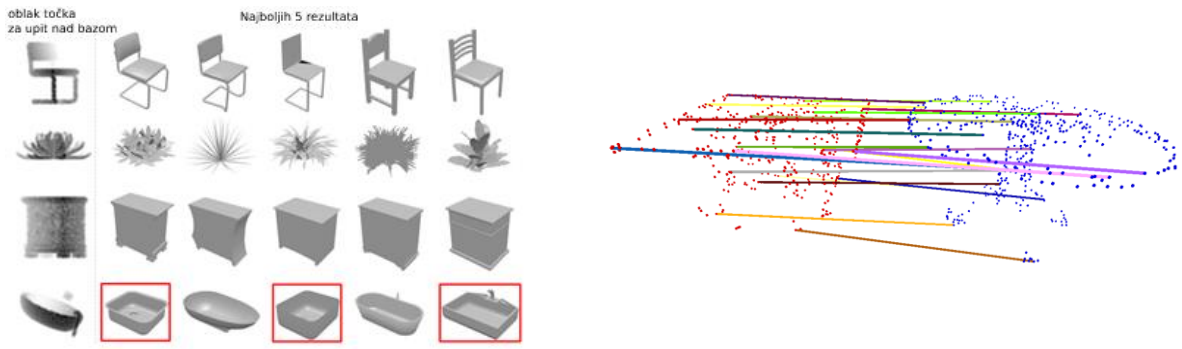
$$d_H(S_1, S_2) = \max\{max_{T \in S_1} min_{P \in S_2} d(T, P), max_{P \in S_2} min_{T \in S_1} d(P, T)\} \qquad (3\text{-}3)$$

Za klasifikaciju objekata to znači, prema teoremima autora iz [16], da mreža uči opisati oblik određene klase ograničenim vektorom globalnog deskriptora čija je veličina ograničena parametrom K sloja sažimanja značajki koja se ujedno zove „bottleneck“ dimenzija aproksimirane

[1] Definira se kao mjera udaljenosti dva skupa

funkcije Hausdorffove udaljenosti. Za svaku klasu objekata postoje margine (dva podskupa ulaznog prostora) između kojih mreža daje jednak vektor globalnog deskriptora što omogućava robusnost na šum. Donja margina se naziva skup kritičnih točaka kojih ima K ili manje i predstavlja kostur modela iz iste klase.

Dodatne primjene PointNet mreže leže u vektoru globalnog deskriptora. Autori [16] očekuju da slični oblaci točaka (oni koji prikazuju objekte koji pripadaju istoj klasi) imaju gotovo jednak spomenuti vektor i slične kritične skupove točaka između kojih se može pronaći korespodencija (slično primjerice SIFT značajkama slike koje se koriste u mnogim primjenama u računalnom vidu). Taj vektor se može koristiti kao upit nad bazom modela prilikom traženja sličnih modela, jasno se vidi na slici 3.6. na kojoj je prikazana i korespodencija točaka dobivenih kritičnih skupova kroz mrežu. Oblaku točaka za koje se traže slični modeli pronalazi se vektor globalnih značajki kao i svim modelima u bazi. Na temelju pretrage uzimaju se najbliži susjedi polaznom modelu, na slici je prikazano prvih 5 najsličnijih modela. Crvenim pravokutnikom su označeni modeli koji pripadaju pogrešnoj klasi modela.



Slika 3.6.: Upit nad bazom modela (lijevo) i korespodencija točaka (desno) [16]

3.4.2. Teorijski opis mreže PointNet++

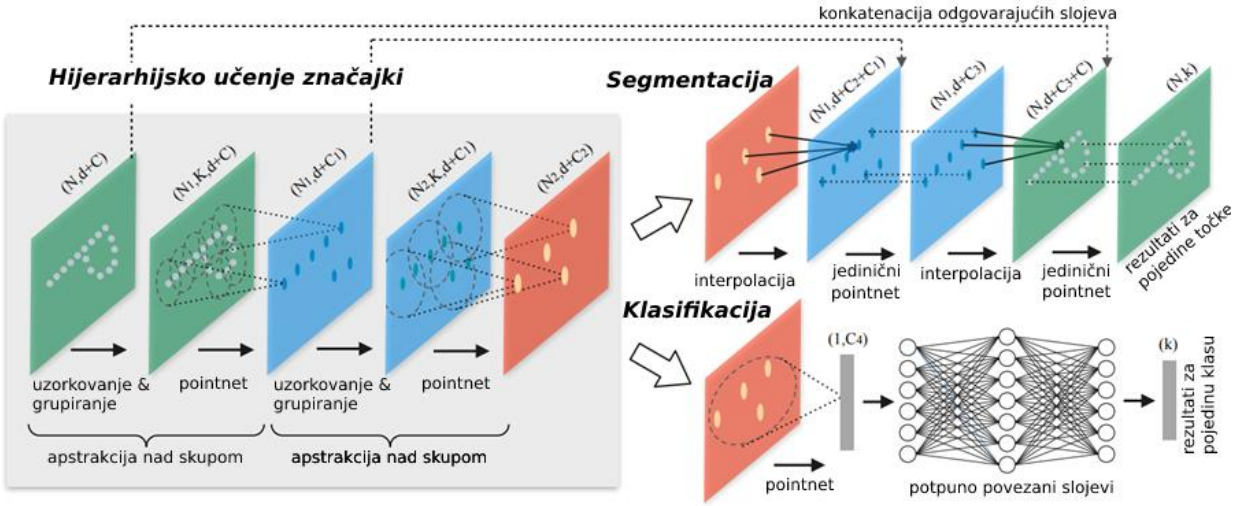
Konvolucijske neuronske mreže, kako je bilo kratko objašnjeno u potpoglavlju 3.4.1., postižu vrhunske rezultate u brojnim domenama kao što su klasifikacija, segmentacija itd. i često su korištene nad podacima u obliku slika koje su dvodimenzionalni ili trodimenzionalni tenzori. Slike su poredani skup podataka u rešetke gdje su sve ćelije (pikseli) jednake veličine, gustoća podataka se ne mijenja. Takve mreže primjenom filtera traže značajke po slici koje se mogu nalaziti bilo gdje na slici zbog toga što se filter primjenjuje na područje slike ispod pomičnog prozora. Promjenom veličine filtera mreža prepoznaje značajke na niskoj razini apstrakcije kao što su sitniji detalji slike do apstraktnijih značajki koje mogu biti kombinacije drugih značajki ili neki globalni fenomeni na slici. U praksi se pokazalo kako višestruko korištenje već naučenih i provjerenih dijelova mreže daje jako dobre rezultate. Korisno je već naučeni filter primijeniti na druge dijelove slike ili potpuno druge slike ako je poznato da dobro detektira tražene značajke i dobro propagira važne informacije dalje kroz mrežu. Pošto su za ovaj rad važni algoritmi koji rade s oblacima točaka, a ne sa slikama, nije moguće koristiti mnoštvo algoritama baziranih na konvolucijskim neuronskim mrežama koji su dobro testirani i daju dobre rezultate u raznim područjima, kao npr. segmetacija slike.

U teorijskom opisu PointNet mreže [16], objašnjeno je kako direktna primjena konvolucijskih neuronskih mreža nad oblacima točaka nije moguća zbog svojstva neporedanosti ulaznog skupa podataka čemu su istraživači u [16] doskočili kako je opisano u potpoglavlju 3.4.1. Qi, Yi, Su i Guibas [3] proširuju PointNet mrežu tako da već poznatu ideju o korištenju filtera (ili dijela mreže) u različitim lokalnim okolinama u ulaznim podacima, kao što je već ustaljeno u radu sa konvolucijskim neuronskim mrežama, primjene na oblake točaka. Analogno, istraživači [3] koriste PointNet mrežu kao dio mreže koji dobro prepoznaje lokalne značajke, a umjesto pomičnog prozora dijele ulazni prostor na hipersfere unutar kojih se traže značajke. PointNet daje lošije rezultate nad ulaznim prostorima promjenjive gustoće podataka i nad podacima gdje su prisutni i važni sitniji detalji, ali inovativno rješava problem neporedanosti podataka u oblacima točaka te se ta dobra karakteristika može iskoristiti u kombinaciji sa navedenim poboljšanjima što se u originalnome radu [3] naziva PointNet++ mreža. Cilj mreže je klasificirati ulazni oblak točaka ili ga segmentirati.

Prilikom rada sa slikama, lokalno receptivno polje se pomiče za definirani korak u pikselima. U slučaju oblaka točaka (npr. 3D snimka realne scene), podaci direktno predstavljaju podatke u Euklidskom prostoru, pomicanje za točno određeni iznos po podacima nema smisla

zbog različite gustoće ulaznih podataka. Primjerice, 3D objekt može sadržavati malo točaka na ravnim dijelovima, a veći broj na mjestima gdje se objekt savija ili ima neku kompleksniju lokalnu strukturu. Podjela ulaznog prostora podataka (analogno, definicija položaja lokalnih receptivnih polja) znatno je složenija nego u slučaju rada sa slikama. Korišteni alat za definiciju centara klastera je algoritam uzorkovanja prema najudaljenijoj točki (FPS) koji ih ravnomjerno odabire kroz čitavi skup ulaznih podataka i uzima u obzir Euklidski prostor u kojemu se ti podaci nalaze, algoritam optimalno odabire točke s obzirom na njihov međusobni položaj i gustoću podataka. Istraživači [3] su pokazali kako nasumična priroda algoritma (nasumično odabiranje početne točke) ne utječe značajno na rezultate (navode 0,17% devijacije u preciznosti njihovog algoritma). Nakon definiranih centara, važno je pronaći podatke koji tome klasteru pripadaju. Qi et al. [3] koriste algoritam k najbližih susjeda ili odabiru točke koje se nalaze unutar nekog radijusa hipersfere s centrom u centru klastera, s tim da drugi način daje bolje rezultate nad određenim skupovima ulaznih podataka. Također, zaključili su da veći radijus pozitivno utječe na preciznost mreže [3].

Sada kada se ulazni prostor može uzorkovati na ključne točke pomoću FPS algoritma i grupirati oko tih točaka, nakon ta prva dva koraka slijedi korištenje PointNet mreže nad svakom tom manjem podskupu podataka što kao rezultat daje vektor značajki za svaki klaster čime se uvodi apstrakcija nad ulaznim podacima. Točke u toj okolini prije prolaska kroz PointNet dio mreže su centrirane oko centra klastera (koordinate točaka su transformirane tako da imaju novi koordinatni sustav sa ishodištem u centru klastera). Slika 3.7. prikazuje načelo rada dijela PointNet++ mreže [3] kojeg nazivaju hijerarhijsko učenje značajki ulaznog skupa podataka.



Slika 3.7.: Hijerarhijsko učenje značajki ulaznog skupa podataka [3]

Nakon odabiranja centara klastera i grupiranja podataka oko nj (engl. *sampling and grouping*) i provođenja takvih podataka kroz PointNet mrežu uvedena je određena apstrakcija nad ulaznim podacima (engl. *set abstraction*) gdje se kroz mrežu ne propagiraju svi podaci već odabrani centri klastera i dobiveni vektori značajki. Dalje kroz mrežu se ne radi sa originalnim podacima nego se, u slučaju klasifikacije, ponovno vrši apstrakcija nad skupom, ponovno se podaci provode kroz PointNet dio mreže (svi podaci iz prethodnih slojeva su svedeni u jedan zajednički klaster) i dobiva se globalni deskriptor ulaznih podataka koji se provodi kroz MLP mrežu nakon čega je izlaz iz mreže vjerojatnost pripadanja određenoj klasi od k klasa (slično kao kod PointNet mreže).

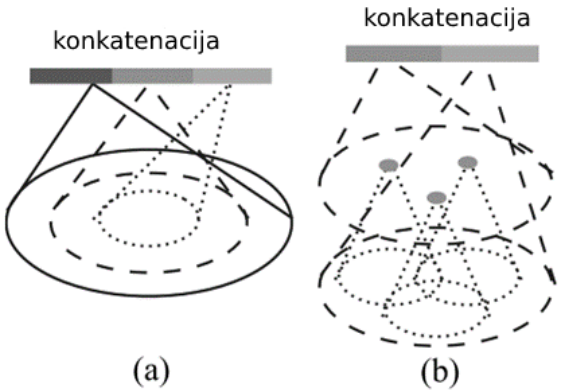
Segmentacija ulaznih podataka se vrši malo drugačije od klasifikacije (slika 3.7., gornji dio slike). Pošto se do izlaza iz mreže sve manje radi sa ulaznim skupom podataka, već se kroz mrežu propagiraju deskriptori lokalnih područja čime se smanjuje brojnost značajki, nije moguće izračunati kako vrijednost koja je pridodana značajki u zadnjem sloju prilikom segmentacije utječe na pojedine ulazne značajke (ili ulazni skup podataka) iz prethodnih slojeva. Značajke u zadnjem sloju nemaju objašnjivo značenje. To su deskriptori lokalnih okolina iz kojih nije moguće jednostavno odrediti koja točka ulaznog prostora, koliko i na koji način utječe na tu značajku a da se iz pridodane vrijednosti nakon segmentacije može odrediti koje točke trebaju dobiti tu vrijednost ili kombinaciju vrijednosti. Autori [3] predlažu dva načina segmentacije ulaznog prostora za svaku točku toga prostora. Prvi način je da klastera ima onoliko koliko ima točaka u ulaznom prostoru čime se dobivaju značajke za svaku točku i jasno je na koju točku se vrijednost na kraju mreže odnosi, no tako dobivena segmentacija iziskuje puno računalnih resursa.

Drugi način propagira vrijednost segmentacije značajki „unazad“ do točaka u ulaznom skupu podataka tako da interpolira vrijednosti segmentacije više značajki trenutnog sloja za sve značajke „prethodnog“ sloja. Za interpolaciju koriste [3] prosjek otežanih vrijednosti segmentacije, s tim da se težine računaju kao recipročna vrijednost udaljenosti značajke za koju se računa vrijednost segmentacije i značajke „idućeg“ sloja na potenciju p. U obzir se uzimaju samo k najbližih značajki iz „idućeg“ sloja prilikom računanja vrijednosti segmentacije.

PointNet++ mreža predstavlja hijerarhijsku primjenu PointNet mreže kao sloja, koji se primjenjuje na različitim skalama. Kombinacijom takvih slojeva moguće je osigurati dobru klasifikaciju ili segmentaciju ulaznih podataka različitih gustoća točaka. Autori [3] opisuju kako u idealnom slučaju mreža na manjoj skali dobiva optimalne značajke za gušće raspoređene detalje objekta, a na većoj skali za rjeđe raspoređene monotonije dijelove objekta. Drugim riječima,

poželjno je pobliže raditi sa gušće raspoređenim podacima koji nose detalje o objektu, ali tako detaljan pregled u slučaju rjeđe raspoređenih podataka nije moguć zbog gubitka informacije pri uzorkovanju podataka.

Istraživači [3] predlažu dva načina kombiniranja značajki pri različitoj skali koja se određuje prema gustoći podataka. Prvi način nazivaju grupiranje prema različitoj skali (engl. *multi-scale grouping*, MSG), a drugi grupiranje prema različitoj rezoluciji (engl. *multi-resolution grouping*, MRG). Hijerarhijsko učenje značajki ulaznog skupa podataka je bazirano samo na jednoj skali, stoga je ono dio PointNet++ mreže [3] gdje se kombinacijom takvih modula dobiva kompletna mreža na MSG ili MRG način što je prikazano na slici 3.8.



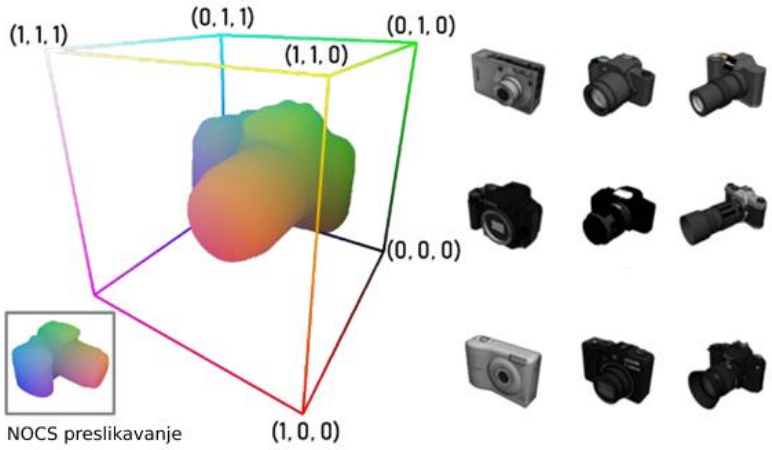
Slika 3.8.: (a) MSG i (b) MRG [3]

Kod MSG načina, autori [3] grupiraju ulazni oblak točaka od manjih prema većim klasterima gdje zadnji PointNet daje jedan deskriptor za svaku od skala. Takvi deskriptori se spajaju u zajednički deskriptor za više skala. MRG daje vektor koji je spoj dva vektora. Lijevi vektor (na slici) je deskriptor nakon apstrakcije nad skupom (PointNet nad značajkama odabranih centara klastera, a ne svim točkama sloja), a desni je deskriptor nad svim točkama tog sloja.

3.5. Normalizirani koordinatni prostor objekata

Detekcija i manipulacija objektima u realnoj sceni robotom kao preduvjet zahtijeva da sustav ima informaciju gdje se objekti nalaze, kako su orijentirani i koje su veličine. Često se takav problem rješava uz pomoć već poznatih CAD modela objekata, što u općenitom slučaju nije optimalno jer u novom okruženju nisu poznati svi CAD modeli objekata. [5] Korisno je dobiti takve informacije za prvi put viđene objekte.

Wang et al. [5] uspješno estimiraju poziciju neviđenih objekta u prostoru na temelju RGB-D slike u obliku graničnog okvira koji gotovo savršeno opisuje objekt svojom veličinom i smještajem u prostoru (engl. *bounding box*). Ideja njihovoga rada je sve objekte preslikati u novi prostor kojeg su nazvali normalizirani koordinatni prostor objekata (engl. *Normalized Object Coordinate Space*, NOCS) gdje su slični objekti (objekti u istoj kategoriji) predstavljeni u jednakom neutralnom položaju. NOCS (slika 3.9) predstavlja trodimenzionalni prostor ograničen unutar jedinične kocke (sve koordinate mogu imati vrijednosti između 0 i 1) unutar kojega se objekti iz iste kategorije jednako orijentiraju, centrirani su u tu jediničnu kocku, a dijagonala graničnog okvira objekta je točno 1. Kako je opisano u njihovome radu [5], iz podataka sa RGB-D kamere na temelju detektirane maske objekta na RGB slici dobiju oblak točaka objekta sa scene. Osim toga oblaka točaka, također računaju 3D reprezentaciju objekta u NOCS-u kao drugi oblak točaka između kojih koristeći Umeyama algoritam traže transformaciju kako bi dobili informacije o položaju i veličini objekta.

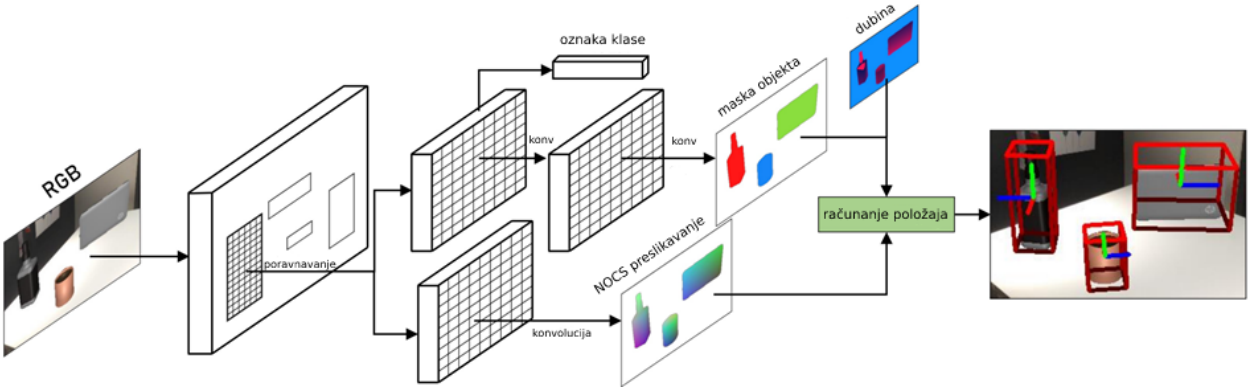


Slika 3.9.: NOCS [5]

U svrhu vizualizacije, u opisanoj jediničnoj kocki, NOCS prostoru, objektu se na svakom mjestu pridodaje određena boja tako da promjena x, y ili z koordinate mijenja R, G ili B vrijednost boje što je vizualizirano na slici 3.9. Njihova [5] mreža uči prepoznati NOCS preslikavanje (engl. *NOCS map*) što je preslikavanje NOCS prostora iz određenog pogleda na RGB sliku što uz podatke o dubini omogućava estimaciju veličine i pozicije objekta.

U početnim fazama, njihova mreža radi samo na RGB slikama na čijim područjima od interesa estimiraju NOCS preslikavanje, masku predmeta (engl. *instance mask*) i kojoj klasi predmeta pripada. Pomoću dodatnih informacija o dubini dobivaju se oblaci točaka kako je već prethodno objašnjeno. Posebno od interesa na slici 3.10. je dio NOCS preslikavanje (engl. *NOCS*

Map) gdje se vide različite boje koje kodiraju koordinate u NOCS prostoru i zadnji desni dio slike gdje se vidi krajnji rezultat metode, a to je granični okvir objekta kojim je definirana veličina i pozicija objekta na sceni i pripadajući koordinatni sustav za objekt.

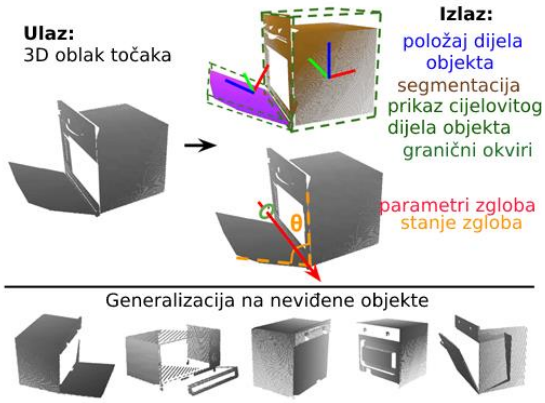


Slika 3.10.: Načelo rada estimacije veličine i pozicije objekta pomoću NOCS-a [5]

Istraživači [5] osim što su uveli novi prostor objekata, predložili su i do tada novi način generiranja sintetičkih podataka nad kojima treniraju svoju mrežu o čemu se može više pročitati u njihovome radu. Osim navedenoga, također proširuju Mask R-CNN arhitekturu [18] kako bi uspješno estimirali NOCS preslikavanje. Glavna ideja Mask R-CNN-a je paralelno određivanje graničnih okvira i klasificiranje objekata sa segmentacijom objekata na slici u različitim granama mreže što je nastavak na prethodne radove o čemu se detaljnije može pročitati u [18]. Wang et al. [5] predlažu proširenje Mask R-CNN arhitekture [18] u obliku tri nove grane koje svaka estimiraju x, y ili z koordinatu za NOCS preslikavanje prilikom segmentacije objekta. Također su ustanovili [5] kako je estimacija koordinata stabilnija i brže konvergira traženom rješenju ukoliko se na to gleda kao problem klasifikacije, a ne kao problem regresije. Umjesto kontinuiranog izlaznog prostora iz mreže, predlažu diskretizaciju na 32 područja u koje se na izlazu iz mreže smještaju koordinate.

3.6. Estimacija kinematičkih parametara objekata određene kategorije

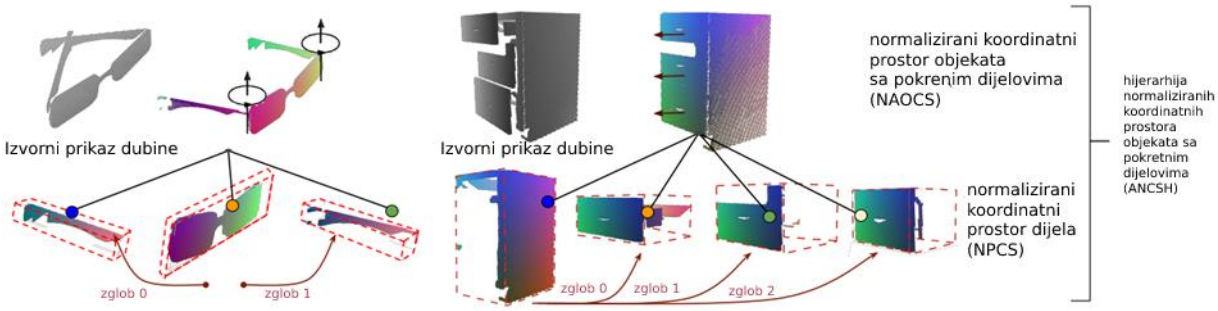
Istraživački rad koji objedinjuje metode opisane u ovom diplomskom radu može se pronaći u [1]. Koristeći PointNet++ mrežu [3] i nadopunjujući ideju o novom koordinatnom prostoru objekata, NOCS [5], za objekte sa pomičnim dijelovima estimiraju poziciju i kinematičke parametre. Slika 3.11. detaljnije opisuje ciljeve rada iz [1].



Slika 3.11.: Ciljevi rada iz [1]

Li et al. [1] iz informacija o dubini objekata (udaljenosti od kamere) iz jednog pogleda estimiraju poziciju svakog dijela objekta, segmentiraju objekte sa više pomičnih dijelova na dijelove, koje prikazuju bez obzira na njihovu zaklonjenost. Osim navedenoga, definiraju granični okvir objekta kojim je definirana veličina i pozicija dijelova, kao i kinematičke parametre rotacijskih ili translacijskih zglobova i njihovo stanje na realnoj sceni. Kroz svoju metodu [1] uklanjaju ovisnost o poznatim CAD modelima objekata kao i ovisnost o specifičnoj instanci objekta određene kategorije (generalizacija u određenoj kategoriji) što će biti opisano u idućim odlomcima. Važno je napomenuti kako je njihova mreža trenirana za svaku kategoriju objekata zasebno, estimiraju reprezentaciju tih objekata u novom prostoru objekata koja je različita za objekte drugih kategorija (različit je broj pomičnih dijelova, kao i neutralan položaj objekta i fizička ograničenja zglobova; objekti iz jedne kategorije imaju jednak kinematički lanac). Stoga se koriste različiti modeli za različite kategorije objekta, no općeniti postupak je jednak.

NOCS rad [5] kojim se uspješno estimiraju veličina i pozicija objekta na sceni ograničen je isključivo na objekte bez pomičnih dijelova, stoga istraživači proširuju tu ideju na tzv. hijerarhiju normaliziranih koordinatnih prostora objekata s pomičnim dijelovima (engl. *Articulation-aware Normalized Coordinate Space Hierarchy*, ANCSH) [1] što je dvorazinski model koordinatnih prostora. Jedan dio u ANCSH-u je zadužen za općenitu reprezentaciju objekta u novom koordinatnom prostoru sa svim dijelovima i naziva se normalizirani koordinatni prostor objekata s pokretnim dijelovima (engl. *Normalized Articulated Object Coordinate Space*, NAOCS). U drugom dijelu su predstavljeni samo dijelovi objekta u novom koordinatnom prostoru koji se naziva normalizirani koordinatni prostor dijelova (engl. *Normalized Part Coordinate Space*, NPCS). ANCSH sa svojim dijelovima, NAOCS i NPCS, je prikazan na slici 3.12.

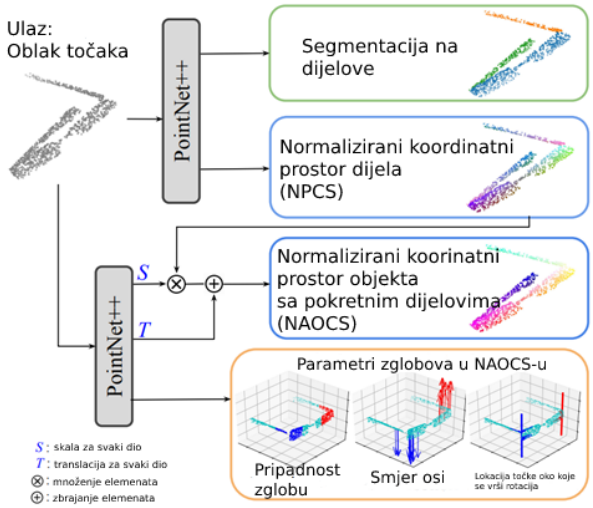


Slika 3.12.: ANCSH [1]

NPCS [1] se može smatrati kao NOCS [5] za dijelove objekata. Oblak točaka dijela je centriran unutar prostora koji je ograničen jediničnom kockom, a veličina dijela je normalizirana tako da granični okvir objekta ima dijagonalu točno 1. Isto kao i u NOCS-u svakoj točki se može pripisati određena boja ovisno o koordinatama unutar NOCS-a kako je opisano u pripadajućem poglavlju. Tako obojani objekti su prikazani na slici 3.12. NAOCS [1] je također baziran na NOCS-u i svojevrsno mu je proširenje. U tome prostoru je prikazan čitav objekt (svi dijelovi zajedno) u neutralnom položaju za taj objekt i također se cijeli objekt normalizira kao i u NPCS-u [1] (i NOCS-u). Na primjeru naočala, nožice su zakrenute za 90 stupnjeva od baze, a naočale su okrenute uspravno gore kao i vektor koji definira dvije rotacijske osi što je prikazano na lijevom dijelu slike 3.12. Takav neutralni položaj olakšava računanje kinematičkih parametara objekata. Dijelovi objekata u NPCS-u su orijentirani jednako kao i njihova reprezentacija u NAOCS-u, točke iz NPCS-a se mogu transformirati u NAOCS množeći sa odgovarajućom skalom i translatiranjem odgovarajućim vektorom.

ANCSH mreža je vizualizirana slikom 3.13. i bazirana je na dva PointNet++ [3] modula. Prvi modul za svaku točku ulaznog oblaka točaka daje vjerojatnost pripadnosti pojedinom dijelu objekta na temelju koje se segmentira ulazni prostor i vraća NPCS estimaciju za svaki dio. Drugi modul računa za svaku točku ulaznog prostora skalu i translacijski vektor između NPCS prostora u NAOCS prostor na temelju kojih se računa prosječna skala i rotacijski vektor svih točaka dijela kako bi se dobili parametri koji vrijede za čitavi dio. Osim računanja preslikavanja iz NPCS-a u NAOCS, drugi modul estimira parametre zglobova u NAOCS prostoru tako što za svakoj točki u NAOCS-u pridružuje pripadnost jednom od zglobova ili nijednom zglobu (u ovisnosti o udaljenosti od zgloba). Na slici 3.13. je pripadnost jednom zglobu označena crveno, drugom plavo, a niti jednom od zglobova svijetlo plavo. One točke koje pripadaju promatranom zglobu glasuju za parametre zgloba na temelju izračunatog 7-dimenzionalnog vektora za svaku točku, a konačni parametri se računaju kao prosjek svih glasova pripadajućih točaka. Prve tri dimenzije

predstavljaju jedinični vektor rotacije ili translacije (os), a zadnje četiri dimenzije predstavljaju točku na pravcu na kojemu leži vektor oko koje se vrši rotacija u slučaju rotacijskog zgloba.



Slika 3.13. ANCSH mreža [1]

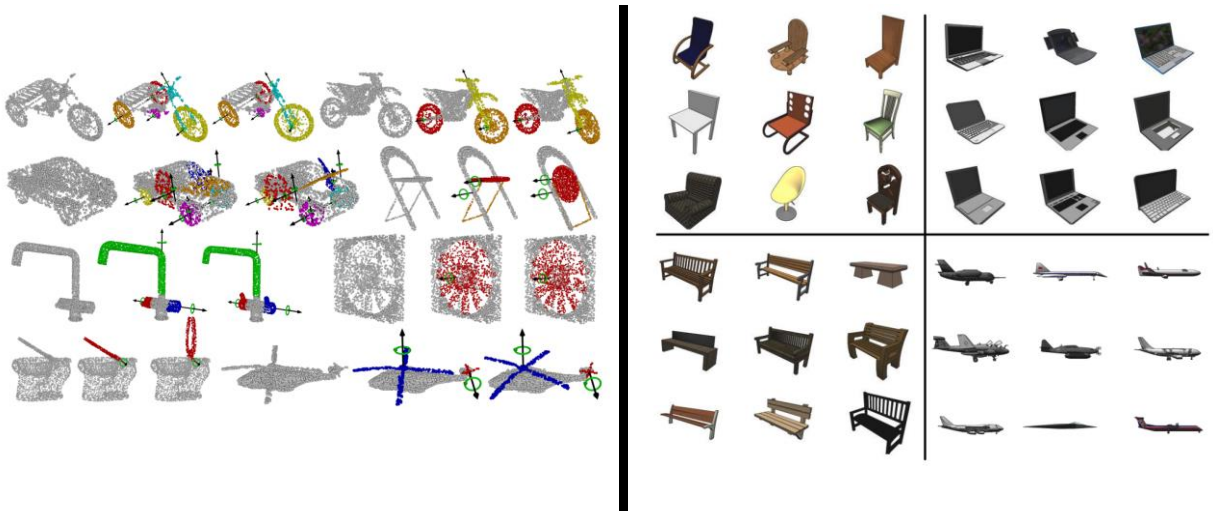
Slično kao i u [5], autori [1] koriste Umeyama algoritam kako bi pronašli rotacijsku matricu, skalu i translacijski vektor između NPCS reprezentacije i pripadajućeg originalnog oblaka točaka s tim da dodatno minimiziraju njihovu predloženu kriterijsku funkciju kako bi uzeli u obzir ograničenja zglobova za objekt dane kategorije. Sa izračunatim parametrima računaju granični okvir objekta. U svom radu [1] analitički su opisali dobivanje kinematičkih parametara zglobova u stvarnom svijetu na temelju dobivenih rotacijskih matrica, translacijskih vektora i kinematičkih parametara u NAOCS prostoru, kao i stanja zgloba na sceni (translacija ili kut rotacije).

4. PROGRAMSKA IMPLEMENTACIJA

4.1. Korišteni skup podataka

Istraživači [1] u svome radu su se bazirali na Shape2Motion [19] skup podataka u kojemu su podaci grupirani u kategorije, najviše je korištena kategorija naočale za koju će biti prikazani rezultati i u ovome radu. U Shape2Motion [19] svaki objekt je segmentiran na svoje pomične dijelove i za svaki objekt je poznat broj zglobova, tipovi i osi zglobova. Taj skup podataka je dobiven metodom koju su predložili Wang et al. [19] iz 3D CAD modela objekata iz skupa podataka ShapeNet [20] s tim da nisu uzeti objekti iz svih kategorija ShapeNet-a.

Shape2Motion [19] je idealni skup podataka za metodu [1] koja se obrađuje u ovom radu jer sadrži sve potrebne informacije o oblaku točaka kako bi se uspješno trenirali modeli. Slika 4.1. prikazuje rezultate metode iz [19] (lijevo) i 3D CAD modele iz ShapeNet-a [20] (desno) bez kojih bi razvoj [1] i mnogih drugih metoda bio znatno otežan zbog manjka dostupnih podataka.



Slika 4.1.: Shape2Motion (lijevo) [19] i ShapeNet (desno) [20] skup podataka

4.2. Postavke

Autori rada [1] na kojemu se bazira ovaj diplomski rad implementirali su svoju metodu na njihovom GitHub repozitoriju [21] gdje se mogu vidjeti potrebne postavke i njihova programska podrška, kao i već trenirani modeli za pojedine kategorije i referencu na skup podataka kojeg su koristili.

U [21] je opisana korištena dodatna programska podrška i korišteni operacijski sustav. Za potrebe ovoga rada je korišteno računalo sa NVIDIA GeForce GTX 1050 na kojemu je potrebno

tjedan dana neprestanoga rada kako bi se trenirala korištena mreža, stoga su korišteni već trenirani modeli. Kao dodatna programska podrška korišten je Python 3.6, CUDA 9.0 i cuDNN 7.4 što preporučaju na svom repozitoriju [21] jer je ta konfiguracija testirana. Korišteni operacijski sustav je Ubuntu 20.04. Potreban je velik broj Python modula kako bi bila uspješno korištena programska podrška, popis istih je također dan na repozitoriju.

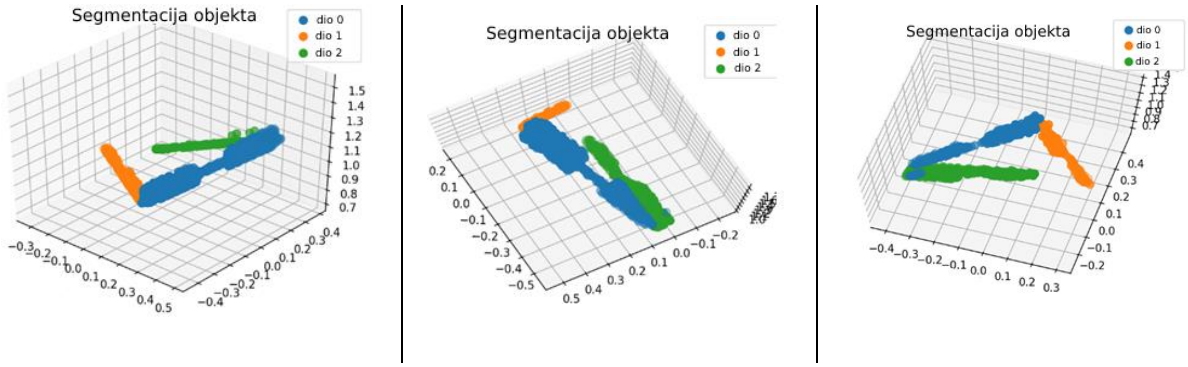
Osim treniranih modela za mrežu, također su objavili već obrađen skup podataka kojega su koristili tijekom treniranja i evaluacije, no u ovome radu nije korišten zbog potrebnog vremena dohvaćanja podataka tolike veličine. Može se pronaći skripta za obradu ulaznih podataka što je korišteno u ovom radu čime se koriste računalni resursi prilikom obrade, ali nije potrebno dohvaćanje velikog broja podataka sa mreže.

Nakon instalacije potrebne dodatne programske podrške, potrebno je prilagoditi programski kod [21] kako bi se uspješno izveo na računalu. Najvažnije je promijeniti apsolutnu putanju do projekta u datoteci *global_info.py*. U ostalim datotekama se najčešće referencira na tu putanju na koju se relativno grade ostale putanje. Imena i struktura direktorija se ne bi trebala mijenjati. Preuzet skup podataka potrebno je smjestiti u *dataset* direktorij u projektu. Posljednji korak pripreme za rad je povezivanje instaliranog Python okruženja i potrebnog CUDA modula (za odgovarajući upravljački program grafičke kartice) što su olakšali u [21] sa *compile_op.sh*.

5. REZULTATI POKUSA

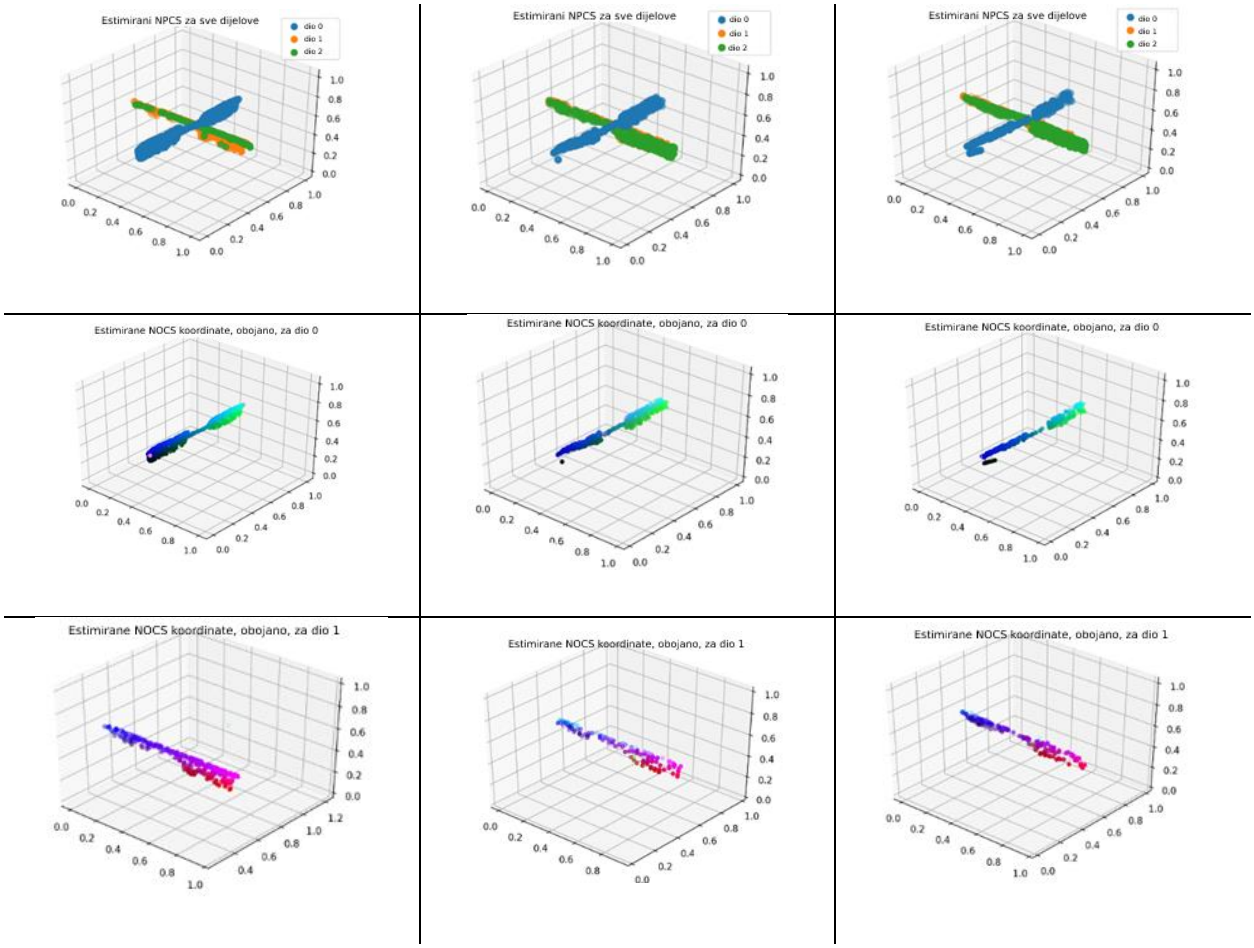
Na slici 3.13. prikazana je arhitektura mreže kojom Li et al. [1] estimiraju kinematičke parametre objekata s pokretnim dijelovima. Prateći tu ilustraciju, ovdje će biti prikazani redom postignuti rezultati korištenjem njihove metode. Za prikaz rezultata najčešće će biti korištena tri primjera nasumično odabranih objekata naočala iz Shape2Motion [19] skupa podataka.

Prvi PointNet++ [3] modul segmentira ulazni oblak točaka na pojedine dijelove i za te dijelove računa njihovu NPCS reprezentaciju. Mreža vrlo dobro segmentira podatke, što se može vidjeti na slici 5.1. Plavom bojom su obojane točke koje pripadaju bazi, a zeleno i narančasto dvjema nožicama naočala.



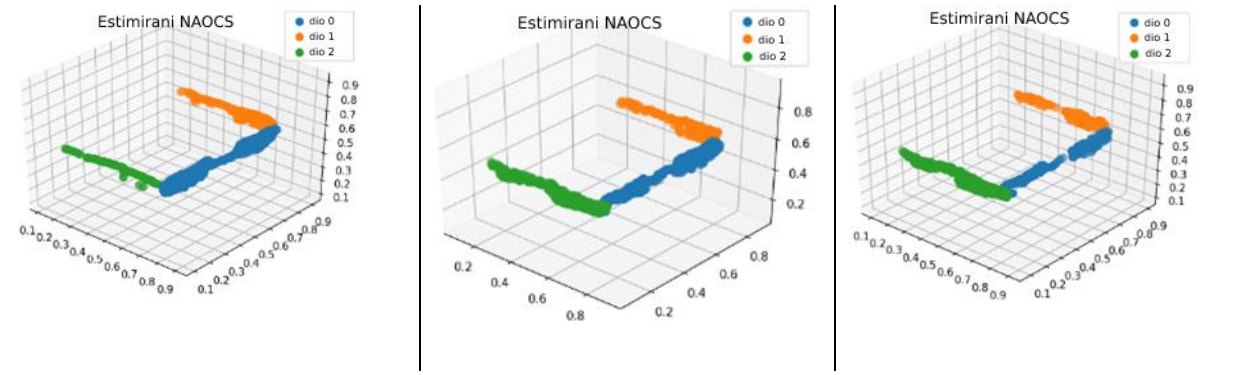
Slika 5.1.: Segmentacija ulaznog oblaka točaka

Nakon segmentacije, svaki dio objekta se prikazuje u NPCS-u (NOCS samo za dio), prvi red slike 5.2. prikazuje sve NPCS za dijelove zajedno gdje se vidi kako su dijelovi skalirani i orijentirani prema njihovom neutralnom položaju, boje predstavljaju kojem dijelu pripadaju. Drugi red slike 5.2. prikazuje NPCS samo za bazu naočala, treći red prikazuje NPCS jedne od nožica, a boje predstavljaju koordinate u NOCS prostoru kako je objašnjeno u potpoglavlju 3.5. Stupci slike 5.2. odgovaraju naočalama sa slike 5.1. redom.



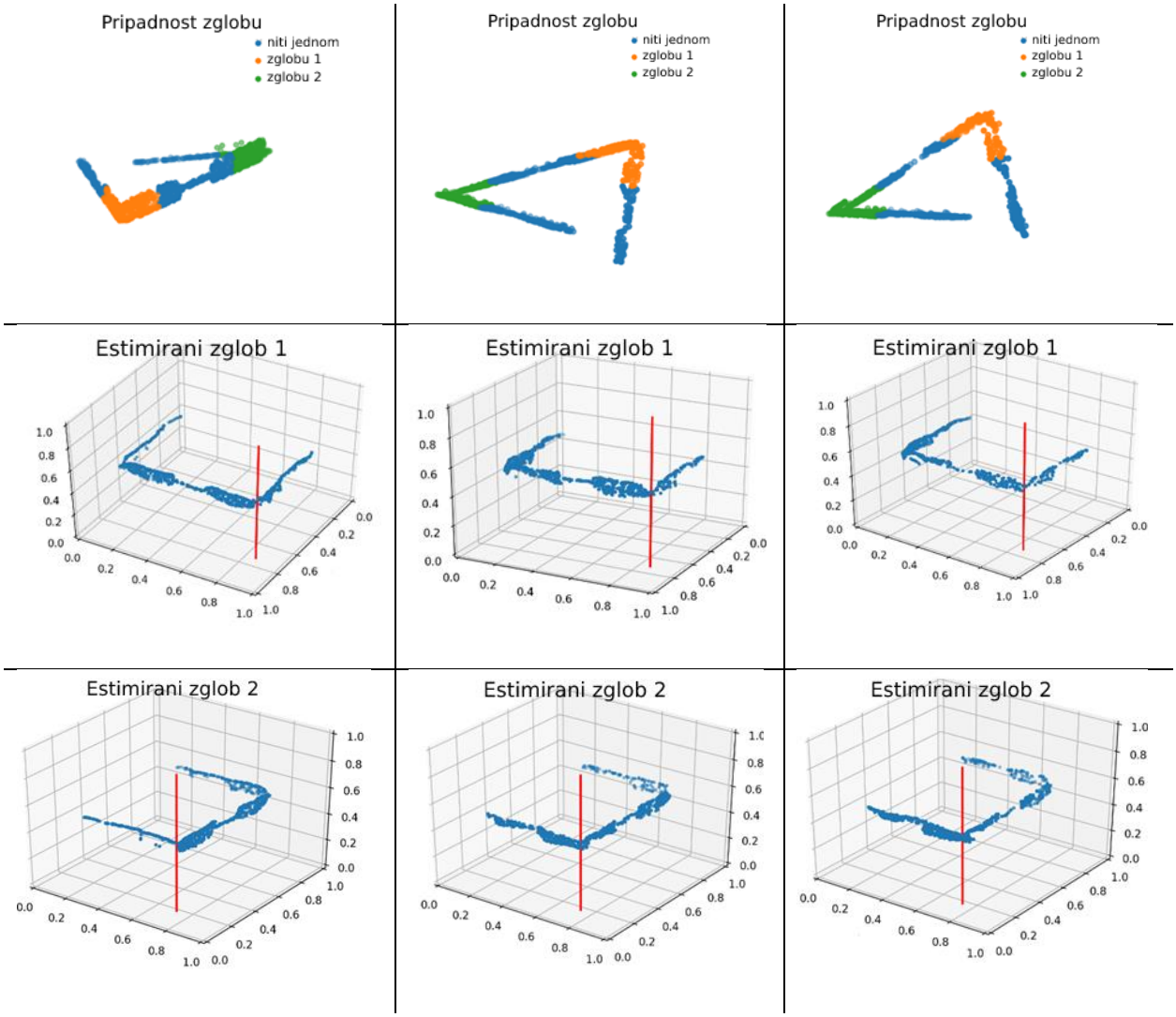
Slika 5.2.: NPCS za sve dijelove zajedno, bazu i jednu od nožica naočala

Nakon što je uspješno segmentiran objekt i njegovi dijelovi postavljeni u NPCS prostor, prelazi se na drugu razinu ANCSH sustava na kojoj se koristi drugi PointNet++ [3] modul sa slike 3.13. Cilj je sve dijelove objekta prikazati u zajedničkom normaliziranom prostoru, NAOCS-u, tako da se izračunaju skale i translacije između NPCS-a i NAOCS-a. Tako dobiveni objekti su postavljeni u neutralan položaj, u ovom slučaju naočale su uspravne a njihove nožice otvorene. Prikaz NAOCS-a za odgovarajuće naočale iz prethodnih slika se nalazi na slici 5.3.



Slika 5.3.: Dobiveni NAOCS

Nakon izračunavanja NAOCS-a za objekte, slijedi pridruživanje točaka jednom od zglobova ili nijednom zglobu koje tada glasuju za rotacijsku os naočala u tome istom prostoru. Na slici 5.4. prvi red prikazuje pripadnost određenom zglobu u prostoru kamere, drugi red prikazuje procijenjenu os rotacije prvog zgloba, a zadnji red prikazuje procijenjenu os rotacije dugog zgloba u NAOCS prostoru. Stupci ponovno pripadaju naočalama sa prethodnih slika.



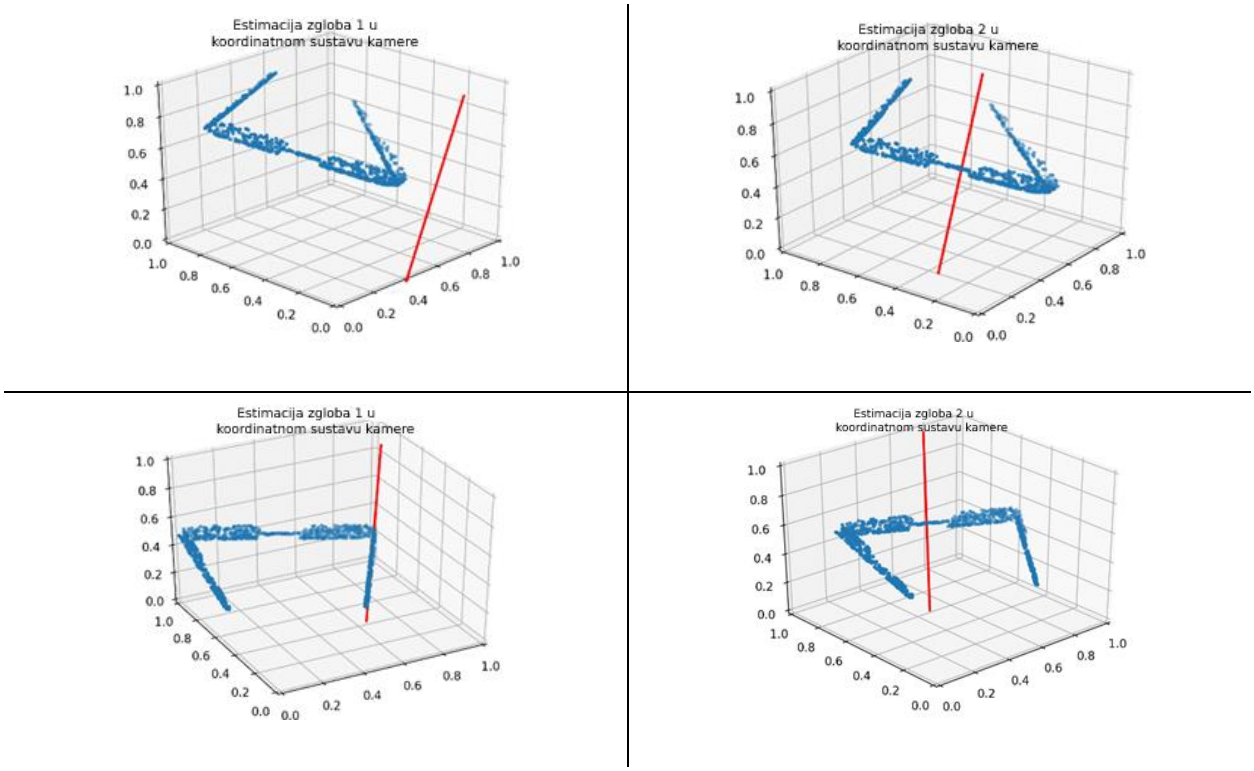
Slika 5.4.: Pripadnost zglobu i procijenjene osi zglobova

Rezultati do sada prikazuju izlaze iz njihove [1] mreže koja daje dobre rezultate. U tablici 5.1. su pokazane informacije o prosječnim pogreškama procjene NPCS prikaza i njihova prosječna odstupanja od idealnog NAOCS prikaza u obliku pogreške rotacije u stupnjevima (zeleno) i translacije (žuto) u odgovarajućem prostoru.

	Baza	Nožica 1	Nožica 2
NPCS	1,46 0,02	4,94 0,04	4,39 0,03
NAOCS	1,81 0,45	7,91 0,69	7,21 0,34

Tablica 5.1.: Odstupanja rotacije i translacije u NPCS i NAOCS procjeni

Problemi se javljaju prilikom prenošenja dobivene osi rotacije u NAOCS prostoru u koordinatni sustav kamere nakon čega se prema njihovoj metodi [1] računa kut otvorenosti naočala naspram neutralnog položaja iz NAOCS-a. Slika 5.5. prikazuje dobivene osi rotacije u koordinatnom sustavu kamere što nije dobar rezultat. Osi su prilično pomaknute i ne odgovaraju stvarnim osima rotacije za naočale. Prikazani su primjeri za obje osi za dvoje naočale. Zbog netočne procjene osi u koordinatnom sustavu kamere nije moguće dobiti kut otvorenosti naočala što je bio krajnji cilj, pogreška procjene otvorenosti naočala iznosi oko 90 stupnjeva što znači da procjene uopće nisu dobre što je očekivano jer osi nisu dobro definirane.



Slika 5.5.: Pogrešna procjena osi rotacije u koordinatnom sustavu kamere

6. ZAKLJUČAK

Objašnjena je teorijska podloga potrebna za rad na ovom projektu koja ujedno služi i kao kratki pregled područja rada i dosega znanosti na polju umjetne inteligencije i robotskog vida. Opisana je metoda na kojoj se temelji rad, kao i slične metode koje rade s oblacima točaka i estimiraju kinematičke parametre objekata. Predstavljen je korišten skup modela i njihovih oblaka točaka na temelju kojih su moguće mnoge druge metode i skup 3D CAD modela pomoću kojih su dobivene potrebne segmentacije objekata na pokretne dijelove i osi rotacija. Opisane su potrebne postavke za implementaciju metode i dana je referenca na programsku podršku. Prikazani su i objašnjeni rezultati dobiveni korištenjem metode na kojoj se temelji ovaj rad.

Korištena neuronska mreža vrlo dobro segmentira objekte na pomične dijelove, procjenjuje NPCS prikaz dijelova objekta kao i transformacije iz dobivenih NPCS prikaza u zajednički NAOCS prikaz što zajedno čini njihov ANCSH model. Osim navedenoga, mreža dobro procjenjuje pripadnost pojedine točke osima i sam položaj osi u NAOCS prostoru.

Nakon korištenja mreže slijedi optimizacijski dio koji korištenjem Umeyama i RANSAC algoritama estimira transformacije iz NAOCS prostora u koordinatni prostor kamere nakon čega se na temelju dobivenih transformacija procjenjuju osi rotacije i kut zakreta dijelova. Rezultati nisu dobri za taj posljednji dio metode počevši od dobivenih osi u koordinatnom sustavu kamere nakon čega, prirodno, ne može biti dobro estimiran kut zakreta dijelova. Prilikom implementacije u ovom diplomskom radu došlo je do pogrešne procjene rotacije, translacije i skale između NAOCS i koordinatnog prostora kamere zbog čega, u ovoj implementaciji, nisu dobri krajnji rezultati. Navedeno ostavlja prostora za daljnji rad na ovoj metodi.

LITERATURA

[1] X. Li, H. Wang, L. Yi, L. Guibas, A. L. Abbott i S. Song, »Category-Level Articulated Object Pose Estimation,« u *Virginia Tech, Stanford University, Google Research, Columbia University*, 2020.

[2] B. Eisner, H. Zhang i D. Held, »FlowBot3D: Learning 3D Articulation Flow to Manipulate Articulated Objects,« 2022.

[3] C. R. Qi, L. Yi, H. Su i L. J. Guibas, »PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,« u *Stanford University*, 2017.

[4] L. Liu, H. Xue, W. Xu, H. Fu i C. Lu, »Towards Real-World Category-level Articulation Pose Estimation,« 2021.

[5] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song i L. J. Guibas, »Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation,« u *Stanford University, Google Inc., Princeton University, Facebook AI Research*, 2019.

[6] H. Xue, L. Liu, W. Xu, H. Fu i C. Lu, »OMAD: Object Model with Articulated Deformations for Pose Estimation and Retrieval,« 2021.

[7] S. Y. Gadre, K. Ehsani i S. Song, »Act the Part: Learning Interaction Strategies for Articulated Object Part Discovery,« 2021.

[8] Z. Xu, Z. He i S. Song, »UMPNet: Universal Manipulation Policy Network for Articulated Objects,« 2022.

[9] F. Aurenhammer, R. Klein i D.-T. Lee, Voronoi diagrams and Delaunay triangulations, World Scientific Publishing Co. Pte. Ltd., 2013.

[10] [Mrežno]. Available: https://commons.wikimedia.org/wiki/File:Dcel-halfedge-connectivity.svg. [Pokušaj pristupa 15 8 2022].

[11] T. Zhao, Y. Feng, J. Zhang, Z. Wang i Z. Wang, »Discrete Element Modelling of Dynamic Behaviour of Rockfills for Resisting Hight Speed Projectile Penetration,« *Computer Modeling in Engineering & Sciences, Tech Science,* 2021.

[12] Y. Eldar, M. Lindenbaum, M. Porat i Y. Y. Zeevi, »The Farthest Point Strategy for Progressive Image Sampling,« IEEE, 1994.

[13] C. Moenning i N. A. Dodgson, »Fast Marching farthest point sampling for point clouds and implicit surfaces,« u *Technical Report, Number 565*, Cambridge, 2003.

[14] M. A. Fischler i R. C. Bolles, »Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,« u *SRI International*, 1981.

[15] S. Umeyama, »Least-Squares Estimation of Transformation Parameters Between Two Point Patterns,« 1991.

[16] C. R. Qi, H. Su, K. Mo i L. J. Guibas, »PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation,« u *Stanford University*, 2017.

[17] T. Marošević, »The Hausdorff distance between some sets of points,« u *MATHEMATICAL COMMUNICATIONS, Department of Mathematics, University of Osijek*, Osijek, Croatia, 2017.

[18] K. He, G. Gkioxari, P. Dollar i R. Girshick, »Mask R-CNN,« Facebook AI Research, 2018.

[19] X. Wang, B. Zhou, Y. Shi, X. Chen, Q. Zhao i K. Xu, »Shape2Motion: Joint Analysis of Motion Parts and Attributes from 3D Shapes,« 2019.

[20] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi i F. Yu, »ShapeNet: An Information-Rich 3D Model Repository,« 2015.

[21] »GitHub,« [Mrežno]. Available: https://github.com/dragonlong/articulated-pose. [Pokušaj pristupa 3 9 2022].

Sažetak

Objekti sa pokretnim dijelovima su svuda oko nas. Korisno je omogućiti robotima da ih prepoznaju i s njima upravljaju na odgovarajući način što je još uvijek otvorena tema u znanstvenoj zajednici. Mjesta za napredak ima. Razvoj neuronskih mreža, robotskih sustava i senzora kroz godine omogućio je dosadašnja i otvara vrata budućim dostignućima. Motivacija ovoga rada je opisati i ispitati algoritme i metode koji se mogu primijeniti za detekciju objekata s pokretnim dijelovima, segmentaciju takvih objekata i estimaciju njihovih kinematičkih parametara kako bi se mogli dizajnirati inteligentni roboti u budućnosti koji sve bolje manipuliraju navedenim tipom objekata. Ukoliko robot uspješno manipulira objektom s pokretnim dijelovima, tada je moguće dizajnirati robote koji uspješno rješavaju ljudima korisne zadatke kao što su otvaranje vrata, otvaranje i postavljanje naočala na osobu koja to ne može samostalno i sl. Algoritmi razmatrani u ovom radu osmišljeni su za rješavanje problema estimacije osi objekata s pokretnim dijelovima, za koliko je taj objekt već zakrenut ili pomaknut i detekcije dijelova objekta za koje robot mora uhvatit svojim alatom.

Ključne riječi

ANCSH, neuronska mreža, NOCS, oblak točaka, PointNet

Articulated object pose estimation

Abstract

Articulated objects are everywhere around us. It is useful to enable robots to recognise and manipulate them in an appropriate way which is still an open subject in the scientific community. There is room for improvement. Development of neural networks, robotics systems and sensors throught the years has enabled past and future achievements. The motivation for this work is to describe and test algorithms and methods that detect articulated objects, segment them into parts and estimate their pose in order to enable future robot design which enables better manipulation of those objects. If a robot successfully manipulates articulated objects, then it is possible to design robots that successfully solve useful tasks, such as opening doors, opening and placing glasses on a person which cannot do so independently etc.. The algorithms considered in this work aim to solve the problem of estimating the axis of articulated objects, how far the object has already been rotated or moved, and the detection of parts of the object that the robot must grasp with its tool.

Keywords

ANCSH, neural network, NOCS, point cloud, PointNet

Životopis

Dijana Ivezić rođena je 5. prosinca 1998. godine u Slavonskom Brodu gdje pohađa osnovnu i srednju tehničku školu kao arhitektonski tehničar. 2017. godine upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek gdje nakon završenog preddiplomskog sveučilišnog studija 2020. godine upisuje diplomski studij na kojemu je trenutno studentica.