



UNIVERZITET U NOVOM  
SADU

FAKULTET TEHNIČKIH  
NAUKA U NOVOM SADU



Nikola Zubić

# **Efektivna funkcija gubitka za generisanje 3D modela na osnovu jedne 2D slike upotrebom dubokog učenja**

Master rad

Novi Sad, 2021.





UNIVERZITET U NOVOM SADU  
FAKULTET TEHNIČKIH NAUKA  
21000 NOVI SAD, Trg Dositeja Obradovića 6

**ZADATAK ZA IZRADU MASTER  
RADA**

Datum:

List:

1/1

(Podatke unosi predmetni nastavnik - mentor)

Vrsta studija:	<b>Master akademске studije</b>
Studijski program:	<b>Softversko inženjerstvo i informacione tehnologije</b>
Rukovodilac studijskog programa:	<b>vanr. prof. dr Miroslav Zarić</b>

Student:	<b>Nikola Zubić</b>	Broj indeksa:	<b>R2 02/2020</b>			
Oblast:	<b>Računarska grafika, računarska vizija</b>					
Mentor:	<b>Dr Dragan Ivetić, red. prof.</b>					
NA OSNOVU PODNETE PRIJAVE, PRILOŽENE DOKUMENTACIJE I ODREDBI STATUTA FAKULTETA IZDAJE SE ZADATAK ZA MASTER RAD, SA SLEDECIM ELEMENTIMA:						
<ul style="list-style-type: none"><li>- problem – tema rada;</li><li>- način rešavanja problema i način praktične provere rezultata rada, ako je takva provera neophodna;</li><li>- literatura</li></ul>						

**NASLOV MASTER RADA:**

**Efektivna funkcija gubitka za generisanje 3D modela na osnovu jedne 2D slike upotrebom dubokog učenja**

**TEKST ZADATKA:**

Proučiti tehnike rekonstrukcije 3D modela nekog realnog objekta na osnovu samo jedne njegove slike. Naći način da se oni unaprede za primenu u mašinskoj viziji, a zasnovano na neuronским mrežama.

Rukovodilac studijskog programa:	Mentor rada:

Primerak za:  - Studenta;  - Mentora



# SADRŽAJ

1. UVOD.....	7
2. PREGLED PRETHODNIH RJEŠENJA.....	11
2.1 3D reprezentacije.....	11
2.2 Diferencijabilni rendering.....	15
2.3 Nenadzorovano učenje oblika i poze korišćenjem diferencijabilnih <i>point cloud</i> -ova.....	19
3. PREDLOŽENA METODA.....	25
3.1 Intuitivni pregled.....	25
3.2 Detalji implementacije.....	27
4. REZULTATI I OSNOVNE INFORMACIJE.....	34
4.1 Glavni rezultati.....	34
4.2 Osnovne informacije o projektu.....	35
5. STUDIJA SLUČAJA.....	38
5.1 Korišćena arhitektura neuronske mreže.....	38
5.2 Detalji o metrikama.....	38
5.3 Eksperimenti i diskusije.....	40
5.4 Ablaciona studija i efikasnost.....	44
5.5 Detalji treniranja modela.....	45
5.6 Dodatni prikaz vizuelnih rezultata.....	47
6. ZAKLJUČAK.....	50
7. BIOGRAFIJA.....	52
8. LITERATURA.....	54
KLJUČNA DOKUMENTACIJSKA INFORMACIJA.....	58
KEY WORDS DOCUMENTATION.....	60

## 1. UVOD

Jedan od glavnih problema u 3D računarskoj grafici i viziji jeste mogućnost modela da nauči reprezentaciju 3D struktura, kao i njihovu rekonstrukciju [9]. Nadzorovano 3D duboko učenje je vrlo efikasno u direktnom učenju iz 3D reprezentacija [1], kao što su *mesh*-evi, vokseli i *point cloud*-ovi. To zahtijeva veliku količinu 3D podataka tokom procesa treniranja, te takođe, njihova reprezentacija je ponekad isuviše kompleksna za sam zadatak direktnog učenja. Ovi faktori doveli su do napuštanja ovog pristupa. Glavni razlozi su neefikasnost u pogledu performansi i prevelik utrošak vremena. Nenadzorovano 3D strukturno učenje „uči“ 3D strukturu bez 3D supervizije i predstavlja obećavajući pristup.

Diferencijabilni rendering (engl. *Differentiable rendering*) je novo polje izučavanja unutar kojeg je omogućeno korišćenje gradijenata 3D objekata kako bi se izvršila propagacija kroz slike [11]. Ovo smanjuje potrebu za skupljanjem i anotiranjem 3D podataka, te omogućuje veću efikasnost prilikom primjene u mnogim drugim sistemima. Sposobnost ovakvih sistema da kreiraju sponu između 3D i 2D reprezentacija, računajući gradijente 2D funkcija gubitaka po 3D strukturi, čini ih ključnom komponentom u nenadzorovanom 3D strukturnom učenju. Ove funkcije gubitka su zasnovane na razlikama između *RGB* vrijednosti piksela [13]. Renderovanjem prediktovane 3D strukture iz specifičnog ugla gledanja, te evaluacijom funkcije gubitka zasnovane na pikselima (engl. *pixel-wise loss*) između renderovane i stvarne (engl. *ground – truth*) slike, dolazi do optimizacije parametara modela kako bi se rekonstruisala željena 3D struktura.

Međutim, ove evaluacione tehnike su veoma neefikasne u pogledu vremenskog iskorišćenja. One uopšte ne doprinose preciznoj rekonstrukciji 3D struktura. U ovom radu, predložena je nova ideja za brzu rekonstrukciju 3D strukture (u obliku *point cloud* siluete), koja se potom konvertuje u 3D *mesh*. Transfer tekture objekta sa 2D slike na rekonstruisani 3D objekat se obavlja na kraju postupka. Za razliku od funkcija gubitka zasnovanih na pikselima, naš pristup koristi Efektivnu funkciju gubitka koja je nastala isključivo na osnovu 2D projekcija 3D tačaka, bez ikakve interpolacije zasnovane na pikselima, sijenčenja (engl. *shading*) i rukovanja vidljivošću (engl. *visibility handling*).

Svi modeli prije našeg su zahtjevali rendering korak, zajedno sa rukovanjem vidljivošću i evaluacijom *shading* modela. Glavni cilj ovog rada jeste u demonstraciji da je moguće izbjegići ove korake i opet dobiti rekonstrukciju kao i drugi *state-of-the-art* modeli. Pri tome, ovi rezultati su većinom bolji, a u najgorem slučaju podjednako dobri kao prethodni pristupi koji vrše kategoriski-specifičnu rekonstrukciju.

Na samom početku, koristi se identična arhitektura konvolutivne neuronske mreže (engl. *Convolutional Neural Networks – CNN*) za predikciju *point cloud* oblika i poze kao ona koju su koristili autori Insafutdinov & Dosovitskiy. Potom, kao doprinos ovog rada predlaže se nova funkcija gubitka, nazvana Efektivna funkcija gubitka, čiji je cilj da evaluira koliko dobro projekcije rekonstruisanog 3D *point cloud*-a prekrivaju siluetu objekta na stvarnoj slici. Nakon toga, korišćena je Puasonova rekonstrukcija površi (engl. *Poisson Surface Reconstruction*) kako bi se transformisao rekonstruisani *point cloud* u 3D *mesh*. Na kraju, vrši se mapiranje teksture na konkretan 3D *mesh* koristeći generativne suparničke mreže (engl. *Generative Adversarial Networks – GANs*) kako bi se dobio konačan izlaz, odnosno 3D *mesh* sa odgovarajućom teksturom na osnovu jedne 2D slike. Metoda je evaluirana na različitim skupovima podataka (uključujući *ShapeNet*, *CUB-200-2011* i *Pascal3D+*). Dobijeni su *state-of-the-art* rezultati pri čemu su nadmašene sve ostale nadzorovane i nenadzorovane metode i 3D reprezentacije, u pogledu performansi, preciznosti i vremena treniranja.

Studija slučaja je sprovedena na način da su izvršeni mnogi detaljni eksperimenti. Kao prvo, eksperimentisano je sa hiperparametrima arhitekture neuronske mreže, te sa finesama u vezi metrika. Evaluacija teksture koristeći *Frechet Inception Distance – FID* je sprovedena samo nad *mesh*-om, nad samom teksturom koja je proizvedena i nad teksturiranim 3D *mesh*-om, tj. konačnim rezultatom. Što se tiče pomenutih skupova podataka, korišćene su podjele na trening i test skupove podataka na isti način kako su to radili prethodni radovi (kako bi se rezultati mogli adekvatno poređiti). Izvršena je ablaciona studija i studija efikasnosti modela. Ablaciona studija podrazumijeva uklanjanje pojedinih komponenti ili komponente iz sistema vještacke inteligencije i proučavanje performansi u takvim uslovima. Na taj način smo u stanju da uvidimo koliki je uticaj konkretne komponente na cijelokupan sistem. Konkretno, u našem slučaju, to može biti uklanjanje pojedinih dijelova Efektivne funkcije gubitka i korišćenje broja uglova gledanja (dostupnih slika) kao hiperparametra.

U narednom poglavlju dat je pregled prethodnih rješenja, dok su predložena metoda i implementacioni detalji opisani u poglavlju 3. U poglavlju 4 nalaze se glavni rezultati uz kratak opis skupa podataka, korišćene metrike i link do koda. Sledeće poglavlje je najveće i sadrži mnogo potpoglavlja, a bavi se detaljnom studijom slučaja koja verifikuje ovaj složeni metod. Nakon toga nalazi se zaključak koji sumarizuje rad, te biografija autora rada i relevantna literatura, odnosno reference.





## 2. PREGLED PRETHODNIH RJEŠENJA

U ovom poglavlju predstavljena su prethodna rješenja koja su se bavila problemom rekonstrukcije 3D modela na osnovu jedne slike. U potpoglavlju 2.1 razmotrene su metode koje koriste puni 3D nadzorovani pristup. Potpoglavlje 2.2 sadrži analizu diferencijabilnog renderinga i poznate okvire koji ga koriste. Rad koji je inspirisao da se kreira Efektivna funkcija gubitka i dobije efikasnije rješenje nego prije opisan je u potpoglavlju 2.3.

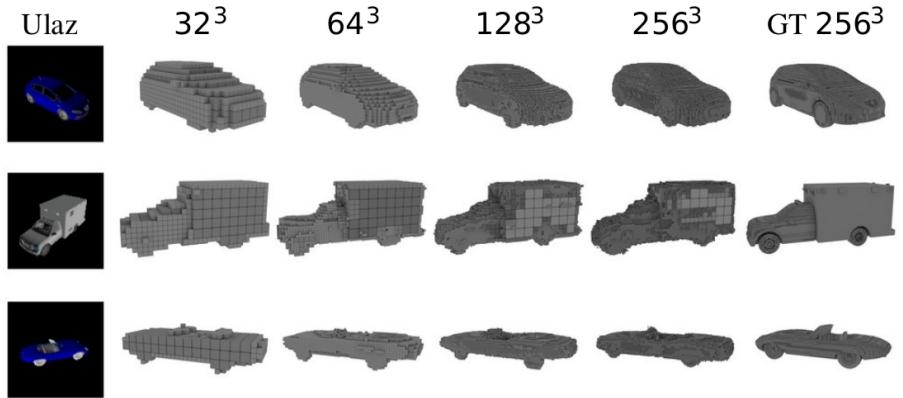
### 2.1 3D reprezentacije

Najraniji radovi [23, 29] koji su riješavali ovaj problem metodama vještačke inteligencije rekonstruisali su *mesh* koristeći puni 3D nadzorovani pristup.

Mreže koje generišu oktalno stablo [23] predstavljaju duboke konvolutivne dekoderske arhitekture koje mogu generisati volumetrične 3D izlaze sa donekle zadovoljavajućim vremenom izvršavanja. Mreža uči da prediktuje i strukturu oktalog stabla, te vrijednosti zauzetosti individualnih celija. Ovo predstavlja prilično važnu tehniku za generisanje 3D oblika. Za razliku od standardnih dekodera koji rade nad regularnim voksel rešetkama, ova arhitektura nema kubičnu kompleksnost. Ovo omogućuje reprezentovanje mnogo veće rezolucije 3D objekata sa ograničenom memorijom.

Ovakve mreže mogu se koristiti za različite namjene, a jedna od njih jeste i 3D rekonstrukcija na osnovu jedne slike. U takvim eksperimentima koristi se dekoder koji operiše nad  $32^3$  voksel rešetkama. Sve je trenirano nad *ShapeNet* skupom podataka, ali koristeći stvarne 3D reprezentacije. Podjela na trening i test skup podataka bila je 80% : 20%. Kao osnova, trenirana je mreža sa gustim (engl. *dense*) dekoderom koji je imao istu konfiguraciju kao i dekoder zasnovan na oktalnim stablima. Oktalna stabla pokazala su se kompetitivnim sa mrežama koje koriste pristupe zasnovane na vokselima nad kompleksnim zadacima.

Takođe, izvršena je evaluacija uticaja različitih rezolucija prilikom korišćenja *ShapeNet* skupa podataka. Na Slici 1 prikazano je da je mreža zasnovana na oktalnim stablima naučila 3D oblike automobila u svim slučajevima, te da su visoko – rezolucione predikcije znatno bolje nego  $32^3$  modeli koji se najčešće koriste za ovaj zadatak. Ovo je podržano i kvantitativnim rezultatima koji su vidljivi u Tabeli 1.  $32^3$  rezultati su znatno lošiji nego ostali. Sa rezolucijom  $256^3$  performanse znatno opadaju zbog šumova (engl. *noise*) gradijenata koji potiču iz viših slojeva mreže, te mogu otežati učenje u dubljim slojevima mreže. Sve to rezultira padom performansi.

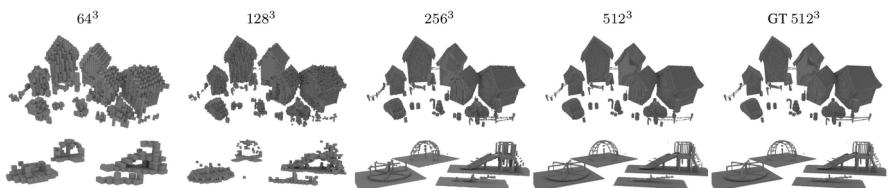


Slika 1. Rekonstrukcija 3D modela na osnovu jedne slike nad *ShapeNet* skupom podataka (kategorija: automobili) koristeći oktalna stabla u različitim rezolucijama.

Rezolucija	32	64	128	256
Rekonstrukcija	0.641	0.771	<b>0.782</b>	0.766

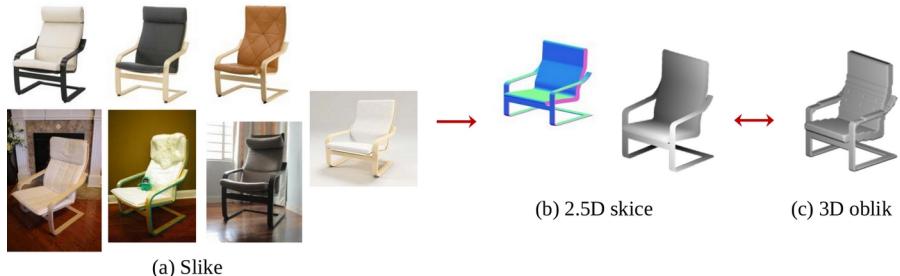
Tabela 1. Rezultati rekonstrukcije nad *ShapeNet* skupom podataka (kategorija: automobili). Nisko-rezolucione predikcije su *upsample*-ovane na rezoluciju  $256^3$ . Često se koriste  $32^3$  modeli koji su znatno lošiji. Najbolje rezultate pokazuje rezolucija sa vrijednošću: 128.

Ova arhitektura se takođe koristi i za generisanje scena. Bitno je napomenuti da su rezolucije  $64^3$  i  $128^3$  neadekvatne za reprezentaciju detalja. Za donje scene, čak i rezolucija  $256^3$  ima probleme sa izrazitim detaljima, što se vidi na Slici 2. Ovaj primjer nam pokazuje da zadaci kao što su učenje sa kraja na kraj (engl. *end-to-end learning*) za problem rekonstrukcije zahtijevaju visoko – rezolucione reprezentacije.



Slika 2. Reprodukcija scena sa mnogo detalja, gdje je visoka rezolucija presudna za dobar kvalitet.

*MarrNet* [29] je *end-to-end* model koji sekvencijalno estimira 2.5D skice i 3D oblik objekta. Ovakav pristup ima prednosti u odnosu na posmatranje samo 3D oblika, jer su 2.5D skice jednostavnije za rekonstrukciju na osnovu 2D slike. Uz to, za 3D rekonstrukciju na osnovu 2.5D skica, sistemi mogu da „uče“ isključivo iz sintetičkih podataka. Ovo je moguće zato što smo u stanju da prilično lako renderujemo realistične 2.5D skice bez modelovanja varijacija koje se odnose na sam izgled objekta, uključujući osvjetljenje, teksturu itd.



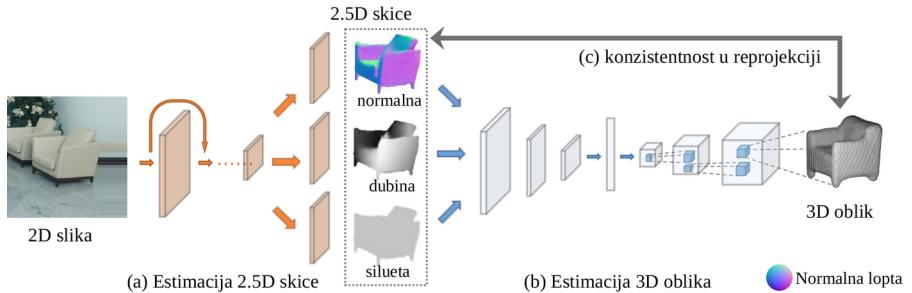
Slika 3. Objekti na stvarnim slikama (a) su skloni varijacijama vezanim za izgled, kao što su boja, tekstura, osvjetljenje, materijali, pozadina itd. Za razliku od ovoga, njihove 2.5D skice kao što su normalne površi i dubinske mape ostaju konstantne (b). Njihove 2.5D skice se mogu posmatrati kao apstrakcije slike koje očuvavaju sve bitne informacije vezane za 3D oblik unutrašnjosti objekta. Kombinovanjem skica sa naučenim priornim oblicima se rekonstruiše puni 3D oblik (c).

Korišćena je enkoder – dekoder struktura za svaku komponentu ovog sistema. Takođe, uzeta je u obzir i dosljednost reprojekcije između estimirane skice i 3D oblika. Model je nazvan *MarrNet* po teoriji percepcije Dejvida Mara<sup>1</sup> (engl. *David Marr's theory of perception*) koja prva spominje 2.5D skice.

Ovaj pristup donosi par jedinstvenih prednosti. Prvo, korišćenje 2.5D skica omogućuje mnogo domenskih prenosa. Učenje 2.5D skica iz slika je mnogo jednostavnije i robustnije u pogledu prenosa sa sintetičkih na stvarne slike. Dalje, pošto drugi korak u postupku generiše 3D model na osnovu 2.5D skice, treniranje je moguće vršiti u potpunosti se oslanjajući na sintetičke podatke. Iako je renderovanje različitih realističnih slika izazovno, lako je dobiti gotovo savršene normalne površi za objekte i dubinske mape na osnovu grafičkog *engine-a*. Ovo doprinosi olakšanju problema koji se tiče domenske adaptacije.

<sup>1</sup> David Marr (1945-1980), britanski neuronaučnik i neurofiziolog

Takođe, sprovedena su diferencijabilna ograničenja između 2.5D skica i 3D oblika, koji omogućuju *end-to-end* treniranje ovog modela, čak i nad pravim slikama bez ikakvih anotacija. Za dati skup nelabeliranih slika, ovaj algoritam, pretreniran nad sintetičkim podacima, formira 2.5D skice objekata na slici i koristi ih kako bi estimirao 3D oblik željenog objekta. Sistem je evaluiran nad sintetičkim slikama objekata iz *ShapeNet* skupa podataka i stvarnim slikama iz *PASCAL 3D+* skupa podataka.



Slika 4. Model je sastavljen od tri komponente: (a) Estimator 2.5D skice, (b) Estimator 3D oblika i (c) funkcija gubitka koja omogućuje ostvarivanje konzistentnosti u reprojekciji.

Prva komponenta sistema (Slika 4a) uzima 2D *RGB* sliku kao ulaz i prediktuje njenu 2.5D skicu: normalnu, dubinsku i siluetu. Cilj estimacije 2.5D skice jeste u očuvavanju bitnih karakteristika koje su potrebne za rekonstrukciju 3D objekta, uz istovremeno odbacivanje svojstava koja nisu bitna za ovaj zadatak, kao što su tekstura objekta i osvijetljenje. Koristi se enkoder – dekoder mreža, koja enkoduje  $256 \times 256$  *RGB* sliku u 512 mapa obilježja čija je veličina  $8 \times 8$ . Dekoder je sastavljen od četiri skupa konvolutivnih i ReLU slojeva čije su dimenzije  $5 \times 5$ , koji su propraćeni sa konvolutivnim i ReLU slojevima čije su dimenzije  $1 \times 1$ . Izlazne 2.5D skice su rezolucije  $256 \times 256$ . Korišćena arhitektura je *ResNet-18*.

Druga komponenta sistema (Slika 4b) izgrađuje oblik 3D objekta estimirajući 2.5D skice. Ovdje se mreža fokusira na učenje priornog oblika. S obzirom da uzima u obzir normalnu i dubinsku sliku kao ulaz, može se trenirati nad sintetičkim podacima, bez domenskog adaptacionog problema. Arhitektura je zasnovana na *TL* mreži i *3D-VAE-GAN*-u sa enkodersko-dekoderskim stilom.

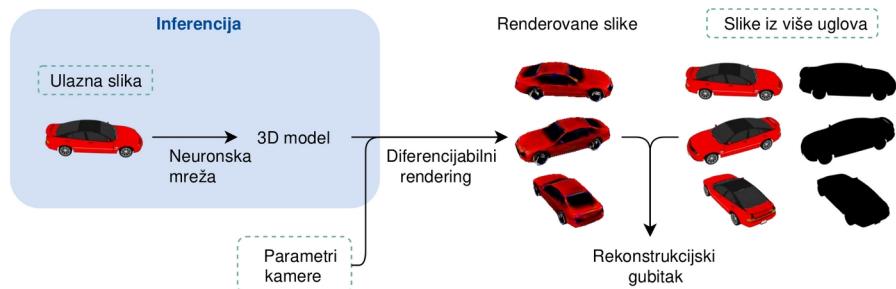
Treća komponenta sistema (Slika 4c) predstavlja uvođenje posebne funkcije gubitka koja se bavi konzistentnošću reprojekcije. Cilj ove funkcije ogleda se u obezbijeđivanju konzistentnosti strukture 3D oblika u odnosu na 2.5D skicu na osnovu koje je taj oblik dobijen. Ova funkcija gubitka sastavljena je od dva dijela, pri čemu jedan dio čini funkciju gubitka koja se tiče projektovane dubine, a druga normalnih površi.

## 2.2 Diferencijabilni rendering

Predikcija 3D modela na osnovu jedne slike uz vizuelne rezultate visokog kvaliteta je moguća korišćenjem diferencijabilnih renderera. Okruženje za diferencijabilni rendering omogućuje da se gradjeni analitički (ili aproksimativno) računaju za sve piksele slike. Poznata okruženja su: *RenderNet* [19] i *OpenDR* [17].

Uprkos tome što je Diferencijabilni rendering nova oblast, iz dana u dan sve više sazrijeva. Glavni razlog jeste kontinualan razvoj novih alata koji pojednostavljaju njegovo korišćenje. Ovo omogućuje novim istraživačima da kreiraju nove modele neuronskih mreža koji su u stanju da interpretiraju 3D informacije na osnovu podataka u vidu 2D slika. Performanse različitih primjena se mogu i dalje unaprijediti čime će se smanjiti i potreba za 3D kolekcijom i anotacijom. U trenutku kada budu dovoljno sazreli da se mogu *deploy*-ovati na embeded uređaje, tada ćemo imati uvid u to kako i sa kojim kvalitetom metode zasnovane na ovom pristupu riješavaju probleme u realnom vremenu (na primjer, autonomna vožnja).

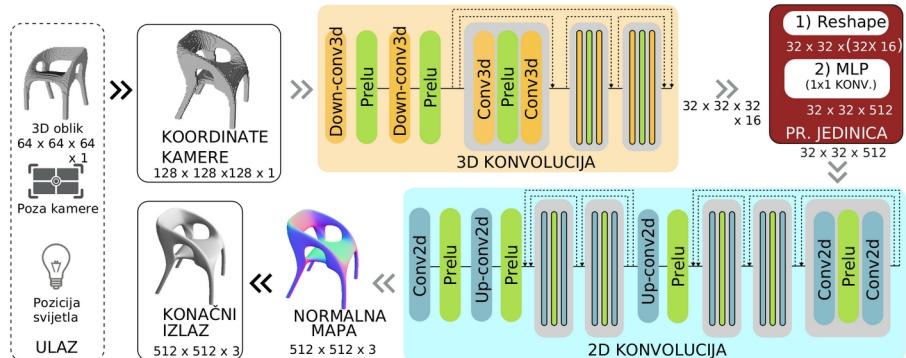
Mogućnosti primjene Diferencijabilnog renderinga su veoma široke. Ona koja nas zanima i koja je tema ovog rada jeste problem rekonstrukcije 3D modela na osnovu jedne slike. Pored toga, ova oblast se koristi i za estimaciju ljudske poze, estimaciju poze ruku i rekonstrukciju lica sa očuvavanjem visokog nivoa detalja. Na Slici 5 prikazan je standardni tok treniranja za problem 3D rekonstrukcije.



Slika 5. Standardni tok treniranja prilikom učenja 3D rekonstrukcije objekta na osnovu samo jedne 2D slike. Isprekidani pravougaonici predstavljaju trening podatke.

*RenderNet* [19] je CNN za Diferencijabilni rendering sa projekcionom jedinicom (koja predstavlja novinu u odnosu na prethodne rade) koja je u stanju da renderuje 2D slike na osnovu 3D oblika. Prostorna okluzija i sjenke su automatski enkodovane unutar mreže. Ovakav sistem je u stanju da uspješno nauči da implementira različite *shader* module i može se koristiti u zadatku inverznog renderinga kako bi estimirao oblik, pozu, osvijetljenje i teksturu na osnovu jedne slike. Dakle, njegova namjena u inverznom renderingu je ono što se poklapa sa temom ovog rada.

Ovaj sistem istovremeno „uči“ oba koraka (rasterizacija i *ray tracing*) procesa renderovanja iz trening podataka. Dodatno inspirisan tradicionalnim tokom renderovanja, koristi strategiju transformacije koordinata i podrazumijeva da je kamera poravnata po svim osama, te da gleda iz smjera negativne z-ose volumetrične rešetke koja diskretizuje ulazni oblik. Umjesto da uči operacije koje su diferencijabilne i luke za implementaciju, kao što su transformacija koordinata krutih tijela ili interakcija svjetlosti sa normalama površi (uzimajući u obzir *Phong*-ov iluminacioni model<sup>2</sup>), ovdje se većina tih informacija čuva i eksplisitno nudi samoj mreži. Ovo omogućuje *RenderNet*-u da se fokusira na kompleksnije aspekte renderovanja, kao što su prepoznavanje vidljivosti objekta i produkovanje osijenčenih boja i sjenki.



Slika 6. Arhitektura *RenderNet* neuronske mreže.

<sup>2</sup> [https://en.wikipedia.org/wiki/Phong\\_reflection\\_model](https://en.wikipedia.org/wiki/Phong_reflection_model)

Kao što se vidi na Slici 6, ova mreža kao ulaz prima voksel rešetku i nad njom izvršava rigidnu transformaciju kako bi izvršila konverziju u koordinatni sistem kamere. Transformisani ulaz, nakon trilinearog semplovanja, se potom koristi kao ulaz za CNN sa projekcionom jedinicom (crveni pravougaonik na slici) kako bi se produkovala renderovana 2D slika. *RenderNet* je sastavljen od 3D konvolucija, projekcione jedinice koja računa vidljivost objekata na sceni i projektuje ih na 2D mape obilježja, praćeno sa 2D konvolucijama koje računaju nivo osijenčenosti (engl. *shading*).

Treniranje ovog modela vrši se računanjem gubitka zasnovanog na pikselima između ciljne slike i izlaza. Opciono, ova mreža može produkovati normalne mape 3D ulaza koji se mogu kombinovati sa svjetlosnim izvorima kako bi se izvršilo osvijetljivanje (engl. *illumination*) scene.

Ulaz u ovu mrežu jeste voksel rešetka  $V$  čije su dimenzije  $H_V \times W_V \times D_V \times C_V$  (koje odgovaraju visini – *height*, širini – *width* i *RGB* kanalima – *channel*) i izlaz je slika  $I$  čije su dimenzije  $H_I \times W_I \times C_I$  (koje odgovaraju redom visini, širini i *RGB* kanalima). Kako bi se premostila razlika između 3D ulaza i 2D izlaza, osmišljena je posebna projekcionalna jedinica. Projekcionalna jedinica sastavljena je od *reshape* sloja i višeslojnog perceptron-a (engl. *Multi-layer Perceptron* – *MLP*). Često se koristi *max pooling* kako bi se 3D ulaz spljoštio po dubinskoj dimenziji, ali uz pomoć ove ideje možemo samo da napravimo mapu siluete 3D oblika. Ova jedinica ne samo da vrši projekciju, nego određuje i vidljivost različitih dijelova 3D ulaza po dubinskoj dimenziji nakon projekcije.

Što se tiče koraka vezanog za *reshaping*, vrši se skupljanje dubinske dimenzije sa mapama obilježja kako bi se mapirao nadolazeći 4D tenzor u 3D stisnuti tenzor  $V'$  čije su dimenzije  $W \times H \times (D \cdot C)$ . Nakon ovoga, odmah dolazi *MLP*, koji je u stanju da uči kompleksnije strukture unutar lokalnog receptivnog polja nego konvencionalni linearni filter. Primjeni se *MLP* nad svakim  $(D \cdot C)$  vektorom, koji je implementiran koristeći  $1 \times 1$  konvolucije. *Reshaping* korak omogućuje svakoj jedinici *MLP*-a da pristupi obilježjima po različitim kanalima i dubinskoj dimenziji ulaza, omogućujući time mreži da uči projekcionalnu operaciju i da računa vidljivost po dubinskoj osi. Ukoliko je dat 3D stisnuti tenzor  $V'$  sa  $(D \cdot C)$  kanala, projekcionalna jedinica formira 3D tenzor sa  $K$  kanala na sledeći način:

$$I_{i,j,k} = f \left( \sum_{dc} w_{k,dc} \cdot V'_{i,j,dc} + b_k \right) \quad (1)$$

gdje su  $i, j$  koordinate piksela,  $k$  je kanal slike,  $dc$  je stisnuti dubinski kanal, gdje su  $d$  i  $c$  dubina i dimenzija kanala originalnog 4D tenzora respektivno, a  $f$  je neka nelinearna funkcija (na primjer, parametarski ReLU).

*OpenDR* [17] je aproksimativni diferencijabilni renderer koji eksplicitno modeluje vezu između promjena u parametrima modela i opservacije na slikama. Na veoma lak način omogućuje opisivanje grafičkog modela sa propagacijom unaprijed i potom automatski preuzima izvode po parametrima modela, te vrši optimizaciju nad njima. Izgrađen je po principu auto-diferencijabilnih paketa i na *OpenGL*<sup>3</sup>-u. Predstavlja lokalnu optimizacionu metodu koja se može inkorporirati u probabilistička programska okruženja. Uz sve navedeno, njegove prednosti su jednostavnost prilikom korišćenja i efikasnost.

Definišimo opservacionu funkciju  $f(\Theta)$  kao proces renderovanja unaprijed koji zavisi od parametara  $\Theta$ . Najjednostavnija optimizacija koja bi pronašla rješenje ovog problema, vršila bi minimizaciju razlike između intenziteta renderovane i opservirane slike,  $E(\Theta) = \|f(\Theta) - I\|^2$ . Naravno, ovaj model vrši složeniju optimizaciju, ali bazična ideja je iskazana upravo ovime – potrebno je izvršiti minimizaciju razlike između sintetizovanih i opserviranih podataka. Dakle, ovaj sistem je proces koji (1) predstavlja piksele kao funkciju parametara modela kako bi simulirao sistem za fizičko snimanje (engl. *physical imaging system*) i (2) omogućuje izračunavanje izvoda vrijednosti piksela po datim parametrima modela.

Neka je  $f(\Theta)$  rendering funkcija, gdje je  $\Theta$  kolekcija svih parametara koji se koriste kako bi se kreirala slika. Ovdje se  $\Theta$  particioniše u lokacije tjemena  $V$ , parametre kamere  $C$  i osvijetljenost po tjemu  $A$ , odnosno  $\Theta = \{V, C, A\}$ . Inverzna grafika je inherentno aproksimativna i veoma je bitno uspostaviti aproksimacije u renderingu unaprijed i njegovoj diferencijaciji. Ovaj model vrši sledeće aproksimacije:

- Izgled (engl. *Appearance – A*): izgled površi po pikselima modeluje se kao proizvod tekture sa preslikavanjem (engl. *mipmapped texture*) i osvijetljenosti po tjemu.
- Geometrija (engl. *Geometry – V*): podrazumijevamo da je 3D scena aproksimirana preko trouglova, koji su parametrizovani sa tjemenima  $V$ . Postoji opcija da se doda pozadinska slika koja se

---

<sup>3</sup> <https://www.opengl.org/>

može naći iza odgovarajuće geometrije. Ne postoji konkretni limit u pogledu broja dozvoljenih objekata.

- Kamera (engl. *Camera – C*): vrši se aproksimacija kontinualnih intenziteta piksela uz pomoć njihovih izsloženih centralnih vrijednosti.

### 2.3 Nenadzorovano učenje oblika i poze korišćenjem diferencijabilnih *point cloud*-ova

Rad koji je inspirisao ovaj master rad se bavi učenjem preciznih 3D oblika i poze kamere na osnovu kolekcije nelabeliranih kategoriski-specifičnih slika [8]. Koristi specifičnu *CNN* arhitekturu kako bi prediktovao oblik 3D modela i pozu na osnovu jedne slike. Međutim, problem koji i ovaj pristup ima jeste vremenska efikasnost zato što koristi diferencijabilnu projekciju *point cloud*-a. U nastavku je dat detaljan opis ovog pristupa.

Ovdje se bavimo zadatkom predikcije 3D oblika objekta i poze kamere na osnovu jednog pogleda (engl. *view*) objekta. Pretpostavimo da nam je dat skup podataka  $D$  sastavljen od pogleda  $K$  objekata, sa  $m_i$  pogleda koji su dostupni za  $i$ -ti objekat: 
$$D = \bigcup_{i=1}^K \left\{ \langle \mathbf{x}_j^i, \mathbf{p}_j^i \rangle \right\}_{j=1}^{m_i}$$
 Ovdje  $\mathbf{x}_j^i$  predstavlja sliku u boji, a  $\mathbf{p}_j^i$  projekciju (može biti silueta, dubinska mapa slike) iz istog pogleda. Uz svaki pogled može biti dostupna i odgovarajuća poza kamere  $c_j^i$ , ali interesantniji slučaj jeste onaj u kojem nisu poznate poze kamere. Ovdje je fokus na tom slučaju.

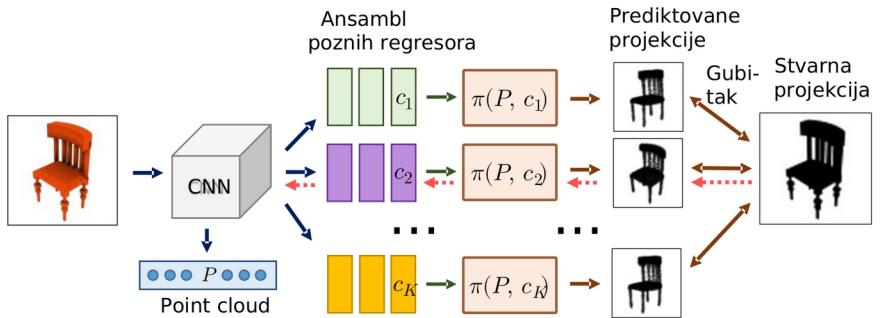
Neka su date dvije slike  $x_1$  i  $x_2$  istog objekta. Koriste se parametarski funkcionalni aproksimatori kako bi se prediktovao 3D oblik (reprezentovan kao *point cloud*) jednog od njih  $\hat{P}_1 = F_P(\mathbf{x}_1, \theta_P)$ , te poza kamere za drugog:  $\hat{c}_2 = F_c(\mathbf{x}_2, \theta_c)$ . U ovom slučaju,  $F_P$  i  $F_c$  su konvolutivne mreže koje dijele većinu parametara. I oblik i poza se predikuju kao vektori fiksne dužine korišćenjem potpuno-povezanih slojeva.

Ukoliko su date predikcije, renderuju se prediktovani oblici na osnovu određenog pogleda:  $\hat{\mathbf{p}}_{1,2} = \pi(\hat{P}_1, \hat{c}_2)$ , gdje je  $\pi$  diferencijabilni renderer za *point cloud*. Funkcija gubitka je raskorak između prediktovane projekcije i stvarnog izlaza. Koristi se standardna srednja kvadratna greška (engl. *Mean Squared Error – MSE*), tako što se vrši sumiranje po cijelom skupu podataka:

$$\mathcal{L}(\theta_P, \theta_c) = \sum_{i=1}^N \sum_{j_1, j_2=1}^{m_i} \|\hat{\mathbf{p}}_{j_1, j_2}^i - \mathbf{p}_{j_2}^i\|^2 \quad (1)$$

Intuitivno gledano, ova treninig procedura zahtijeva da za sve parove pogleda istog objekta, rezultati renderovanja prediktovanog *point cloud*-a se poklapaju sa stvarnim pogledima.

Bazična implementacija ovog pristupa nije u stanju da predikuje poze na precizan način. Ovo je uzrokovano problemom lokalnog minimuma: pozni prediktor konvergira ili tako da estimira sve objekte gledajući ih otpozadi, ili sprjeda. Posmatrajući dvije siluete, teško je razlikovati različite poglede, čak i ljudskim okom.



Slika 7. Ansambl poznih regresora specifično dizajniran kako bi razriješio problem sa sličnim pozama. Mreža vrši predikciju raznolikog skupa  $\{c_k\}_{k=1}^K$  poznih kandidata, od kojih se svaki koristi kako bi se izračunala projekcija prediktovanog *point cloud*-a  $P$ . Ažuriranje težina (propagacija unazad označena isprekidanim crvenim crticama na slici) se vrši samo za pozognog kandidata koji se najbolje poklapa sa stvarnim iz skupa podataka.

Ovaj problem riješava se tako što se umjesto jednog pozognog regresora  $F_c(\cdot, \theta_c)$  uvodi ansambl  $K$  poznih regresora  $F_c^k(\cdot, \theta_c^k)$ , što se vidi na Slici 7. Potom se ovaj sistem trenira koristeći „*hindsight*“ gubitak na sledeći način:

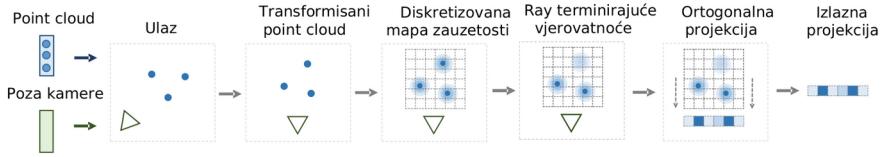
$$\mathcal{L}_h(\theta_P, \theta_c^1, \dots, \theta_c^K) = \min_{k \in [1, K]} \mathcal{L}(\theta_P, \theta_c^k) \quad (1)$$

Glavna ideja je da svaki od prediktora „uči“ da se specijalizuje nad podskupom poza i da oni zajedno prekriju kompletan interval mogućih vrijednosti. Nisu potrebne nikakve dodatne mjere kako bi se ovo osiguralo: ovo proizilazi prirodno kao rezultat nasumične inicijalizacije težina ukoliko je arhitektura neuronske mreže adekvatna. Različiti pozni prediktori bi trebalo da imaju par (makar 3) slojeva koji nisu dijeljeni.

Iz ansambla poznih regresora potrebno je odabrati jedan regresor korišćenjem najboljeg modela iz ansambla kao učitelja. Ovaj najbolji model se bira na osnovu gubitka definisanog u gornjoj jednačini. Tokom testnog vremena, ne koristi se ansambl nego najbolji odabrani regresor kako bi se estimirala poza kamere. Gubitak za treniranje studenta se izračunava kao razlika između uglova dvije rotacije koje su reprezentovane preko kvaterniona:  $L(q_1, q_2) = 1 - \text{Re}(q_1 q_2^{-1} / \|q_1 q_2^{-1}\|)$ , gdje  $\text{Re}$  predstavlja realan dio kvaterniona.

Ključna komponenta ovog modela jeste diferencijabilni *point cloud* renderer  $\pi$ . Ukoliko je dat *point cloud*  $P$  i poza kamere  $c$ , generiše se pogled  $\mathbf{P} = \pi(P, c)$ . *Point cloud* može imati signal, kao što je boja, koji je pridodat naknadno, te se u tom slučaju on može projektovati na pogled.

Ideja ovog modula, gledano na visokom nivou, jeste da izvrši izglađivanje *point cloud-a* reprezentujući tačke sa funkcijama gustine. Formalno, podrazumijevamo da je *point cloud* skup  $N$  torki  $P = \{\langle \mathbf{x}_i, s_i, \mathbf{y}_i \rangle\}_{i=1}^N$ , od kojih svaka ima sledeće informacije: pozicija tačke  $\mathbf{x}_i = (x_{i,1}, x_{i,2}, x_{i,3})$ , parametar veličine  $s_i$ , te pridruženi signal  $\mathbf{y}_i$  (na primjer, *RGB* boje). U većini eksperimenata parametar veličine je dvodimenzionalni vektor koji uključuje informaciju o kovarijansi izotropičnih Gausijana i faktor skaliranja. Uopšteno gledano,  $s_i$  može predstavljati proizvoljnu parametarsku distribuciju (na primjer, korišćenje Gausijana sa punim matricama kovarianse). Parametri veličine se mogu specificirati manuelno ili naučiti zajedno sa pozicijama tačaka.



Slika 8. Diferencijabilni rendering *point cloud-a*. Prikazane su 2D u 1D projekcije radi ilustracionih razloga, a u praksi se vrši 3D u 2D projektovanje. Tačke se transformišu prema parametrima kamere, podrazumijevamo da su glatki i diskretizovani. Vrši se rezonovanje o okluziji preko jednog vida *ray tracing-a* i rezultat se projektuje ortogonalnim putem.

Generalni tok diferencijabilnog renderinga je ilustrovan na Slici 8. Proces počinje transformisanjem pozicija tačaka u standardni koordinatni okvir koristeći projektivnu transformaciju  $T_c$  koja odgovara pozicijskoj kamere  $c$  od interesa:  $\mathbf{x}'_i = T_c \mathbf{x}_i$ . Transformacija  $T_c$  se koristi i za vanjske (engl. *extrinsic*) i za svojstvene (engl. *intrinsic*) parametre kamere. Pored toga, računavaju se transformirani parametri veličine  $\mathbf{s}'$  (konkretno transformaciono pravilo zavisi od korišćene distribucije). Podešavanje matrice za transformacije kamere je takvo da nakon transformacije odgovara ortogonalnoj projekciji po trećoj osi.

Kako bi se omogućio tok gradijenata (engl. *gradient flow*), reprezentuje se svaka tačka  $(\mathbf{x}_i, \mathbf{s}_i)$  glatkom funkcijom  $f_i(\cdot)$ . U ovom radu, ta funkcija je podešena da radi kao skalirane gustinske Gausijane. Funkcija zauzetosti za *point cloud* je klipovana suma individualnih funkcija koje su posebno definisane za svaku tačku:

$$o(\mathbf{x}) = \text{clip} \left( \sum_{i=1}^N f_i(\mathbf{x}), [0, 1] \right) \quad (1)$$

$$f_i(\mathbf{x}) = c_i \exp \left( -\frac{1}{2} (\mathbf{x} - \mathbf{x}'_i)^T \Sigma_i^{-1} (\mathbf{x} - \mathbf{x}'_i) \right) \quad (1)$$

gdje su  $\langle c_i, \Sigma_i \rangle = \mathbf{s}_i$  parametri veličine. Vrši se diskretizacija rezultujuće funkcije u rešetku čija je rezolucija  $D_1 \times D_2 \times D_3$ . Treći indeks odgovara projekcionoj osi, gdje je indeks 1 najbliži kameri, a  $D_3$  najudaljeniji od kamere.

Prije nego što se izvrši projekcija rezultujućih zapremina na površ, potrebno je biti siguran da se signal iz tačaka kod kojih je prisutna okluzija ne „sudara“ sa unutrašnjim tačkama. Rezonovanje vezano za okluziju se radi korišćenjem diferencijabilnog *ray tracing-a*. Vrši se konverzija zauzetosti (engl. *occupancy*)  $O$  u *ray* terminirajuće vjerovatnoće  $r$  na sledeći način:

$$r_{k_1, k_2, k_3} = o_{k_1, k_2, k_3} \prod_{u=1}^{k_3-1} (1 - o_{k_1, k_2, u}) \text{ ako } k_3 \leq D_3(1)$$

$$r_{k_1, k_2, D_3+1} = \prod_{u=1}^{D_3} (1 - o_{k_1, k_2, u}) (1)$$

Intuitivno, ćelija ima veliku terminirajuću vjerovatnoću  $r_{k_1, k_2, k_3}$  ukoliko je njena vrijednost zauzetosti  $o_{k_1, k_2, k_3}$  velika i sve prethodne vrijednosti zauzetosti  $\{o_{k_1, k_2, u}\}_{u < k_3}$  su male. Dodatna pozadinska ćelija  $r_{k_1, k_2, D_3+1}$  se koristi kako bi bili sigurni da je suma terminirajućih vjerovatnoća jednaka 1.

Konačno, vrši se projekcija zapremine na površ, što zapisujemo na sledeći način:

$$p_{k_1, k_2} = \sum_{k_3=1}^{D_3+1} r_{k_1, k_2, k_3} y_{k_1, k_2, k_3} (1)$$

Ovdje je  $y$  signal koji se projektuje, čime se definiše modalnost rezultata. Kako bi se dobila silueta, vrši se podešavanje:  $y_{k_1, k_2, k_3} = 1 - \delta_{k_3, D_3+1}$ . Za dubinske mape, podešava se  $y_{k_1, k_2, k_3} = k_3/D_3$ . Na kraju, kako bi projektovali signal  $\mathbf{Y}$  koji je povezan sa *point cloud-om*, kao što je boja, podesimo da je  $\mathbf{y}$  diskretizovana verzija normalizovane distribucije signala:

$$\mathbf{y}(\mathbf{x}) = \sum_{i=1}^N \mathbf{y}_i f_i(\mathbf{x}) / \sum_{i=1}^N f_i(\mathbf{x}) (1)$$

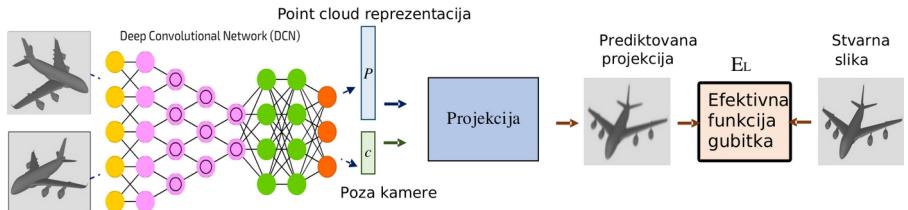


### 3. PREDLOŽENA METODA

U ovom poglavlju detaljno je predstavljena predložena metoda koja vrši optimizaciju metode iz poglavlja 2.3 na takav način da izbacuje diferencijabilni *point cloud* renderer kao komponentu i mijenja ga sa Efektivnom funkcijom gubitka, što dovodi do efikasnijeg riješenja. U prvom potpoglavlju 3.1 dat je intuitivni pregled rješenja gdje je cilj da se na visokom nivou opiše ideja ovog master rada. U potpoglavlju 3.2 dati su implementacioni detalji vezani za kompletan proces, sa fokusom na Efektivnu funkciju gubitka koja je glavna novina rada.

#### 3.1 Intuitivni pregled

Kako bi prevazišle probleme strukturnog 3D učenja, nenadzorovane metode su uvele različite diferencijabilne renderere [8, 9, 11, 3, 16, 17, 19]. Koristeći renderere prvo se izvrši renderovanje rekonstruisanog 3D oblika u 2D slike iz različitih uglova gledanja, a potom se oni koriste kao rezultat koji bi inače bio dobitjen da je vršena kompletna supervizija. Nakon ovoga, računaju se gubici zasnovani na pikselima (engl. *pixel – wise loss*) između tih 2D slika iz različitih uglova gledanja i stvarnih (engl. *ground – truth*) slika iz skupa podataka. Pošto je renderer diferencijabilan, gubitak između tih slika se koristi kako bi se mreža trenirala propagacijom unazad.



Slika 9. Ova metoda ne koristi rendering proces i zahtijeva samo 2D projekcije 3D *point cloud*-ova. Tokom generisanja 3D oblika koristeći višestruke slike silueta (iz različitih uglova gledanja), 2D projekcije svih tačaka na obliku treba da uniformno prekrivaju siluetu iz svakog od dostupnih uglova gledanja. Ovo je implementirano koristeći dvije ključne ideje (koje zajedno formiraju Efektivnu funkciju gubitka). (1) Za 3D oblike koji su formirani od strane 3D tačaka, njihove projekcije za svaki pogled treba da budu locirane unutar siluete. (2) Sve projekcije za svaku siluetu treba da se uniformno distribuiraju. Ovo se ostvaruje maksimizacijom gubitka između svakog od parova tih 2D projekcija. **P** - *Point cloud* reprezentacija, **C** - poza kamere.

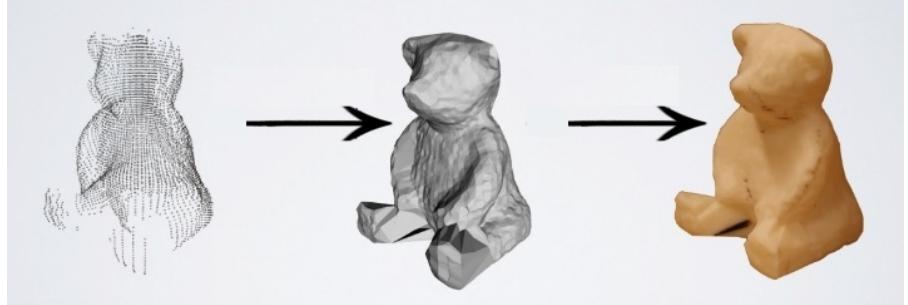
Kako bi evaluirali gubitak zasnovan na pikselima, prethodni diferencijabilni rendereri renderovali su slike uzimajući u obzir neku formu interpolacije [3] rekonstruisane 3D strukture po svakom pikselu, kao što je rasterizacija i rukovanje vidljivošću.

Ovdje se trenira mreža koja „uči“ da generiše 3D *point cloud* na osnovu jedne slike koristeći slike iz skupa podataka (iz različitih uglova gledanja) kao superviziju, što je suprotno onim pristupima koji su koristili stvarne *point cloud*-ove kao superviziju.

Trenutne metode vrše renderovanje zasnovano na diferencijabilnim rendererima koji renderuju slike rekonstruisanih 3D oblika i potom minimizuju gubitak zasnovan na pikselima između njih i stvarnih slika (iz skupa podataka).

Ukupna Efektivna funkcija gubitka nam govori koliko dobro projektovane tačke prekrivaju ciljnu siluetu. Ovaj proces čine dva dijela, jedan koji „tjera“ sve projekcije u siluetu (gdje se one na početku nasumično inicijalizuju), te drugi koji pomjera projekcije tako da je distanca između svake dvije od njih maksimalna moguća, što omogućuje projekcijama da uniformno prekrivaju siluetu. Počevši od nekog *point cloud*-a (nasumično inicijalizovanog), možemo „gurnuti“ sve projekcije u siluetu koristeći prvi dio funkcije gubitka, a potom, koristeći drugi dio funkcije gubitka, uniformno distribuirati projekcije tako da prekriju cijelu siluetu.

Nakon što se završi proces prikazan na Slici 9, generiše se 3D *point cloud* za željenu sliku. Poslije ovog procesa, primjeni se Puasonova rekonstrukcija površi [12] kako bi se generisao 3D *mesh* na osnovu 3D *point cloud*-a i potom se koristi GAN za mapiranje teksture za konkretni 3D *mesh* čime se produkuje teksturirani 3D *mesh*, čija je tekstura zasnovana na teksturi ulazne slike. Kompletan proces prikazan je na Slici 10. Glavni dio ovog rada bavi se upravo implementacionim detaljima Efektivne funkcije gubitka  $E_L$  koja predstavlja glavnu novinu. Rezultati se porede sa ostalim pristupima, a više detalja i studija slučaja nalazi se u poglavlju 5.



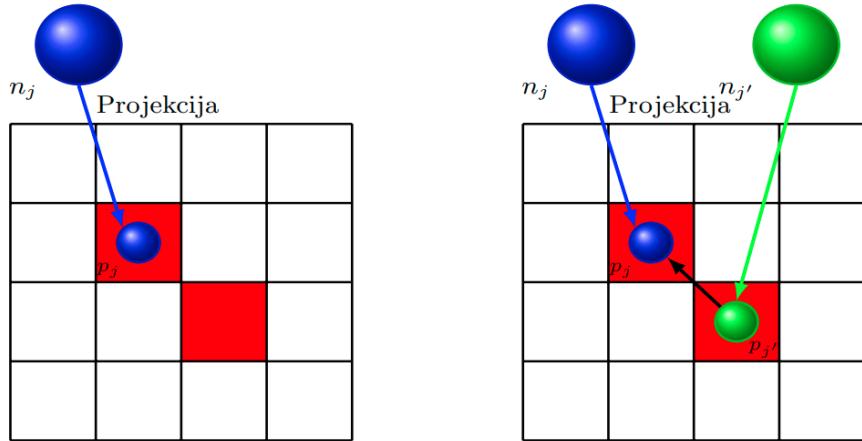
Slika 10. Generisani 3D *point cloud* se transformiše u 3D *mesh* i potom teksturira.

### 3.2 Detalji implementacije

Cilj je naučiti strukturu 3D *point cloud*-ova ( $P$ ) koji su formirani od strane  $N$  tačaka  $n_j$  samo na osnovu  $G_t$  stvarnih slika siluete  $S_i$ , gdje je  $j \in [1, N]$  i  $i \in [1, G_t]$ . Tekući diferencijabilni rendereri se oslanjaju samo na *point cloud*-ove ( $P$ ) koji se renderuju u rasterske slike  $S'_i$  iz  $i$ -tog ugla gledanja, koji se onda koriste kako bi se produkovao gubitak poređenjem  $S'_i$  i  $S_i$  piksel po piksel. Ovi koraci nisu neophodni kako bi se dobilo precizno riješenje.

Neka je projekcija tačke  $n_j$  u pogledu  $i$  označena sa  $p_j^i$ . Greška evaluira koliko dobro skupovi projektovanih tačaka  $\{p_j^i \mid j \in [1, N]\}$  prekrivaju siluetu objekta. Gubitak je sastavljen iz dva dijela.

Ukoliko imamo prediktovani 3D *point cloud* i binarnu sliku siluete, gubitak se računa na sledeći način: Prvo, vrši se projekcija tačaka  $n_j$  kako bi se dobole projekcije  $p_j$  (piše se skraćeno bez  $i$ , misli se na  $p_j$ ) gdje se piksel vrijednosti svake projekcije  $p_j$  označavaju sa  $\pi_i$ . Prvi dio funkcije gubitka vrši penalizaciju tačaka koje se nalaze van unutrašnjosti tako što se računa razlika  $1 - \pi_i$ , uzimajući u obzir da je glavna jedinica unutrašnjosti na binarnoj slici siluete jednaka 1. Minimizacija ovog gubitka će gurnuti sve projekcije u unutrašnjost (prostor) siluete. Dodatno, drugi dio podešava prostornu distribuciju projektovanih tačaka. On „gura“ parove projekcija u unutrašnjost tako da se nalaze što je moguće dalje jedni od drugih (desna rešetka vidljiva na Slici 11). Dakle, ovakav sistem uređuje 3D lokacije tačaka  $n_j$  kroz njihove projekcije  $p_j$  simultanom optimizacijom oba dijela gubitka.



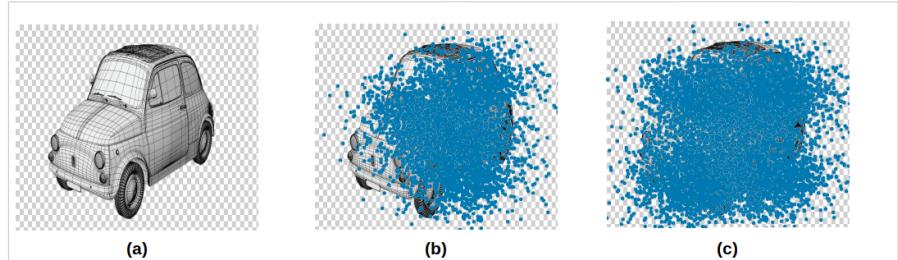
Slika 11. Ljeva i desna rešetka reprezentuju dvije stvarne slike siluete  $S_i$ . Tačka  $n_j$  se projektuje na sliku, a njena projekcija je  $p_j$ . **Lijevo:** Za 3D oblike koji su formirani od strane 3D tačaka, njihove projekcije za svaki pogled trebaju biti locirane unutar siluete, gdje je cijela bijela rešetka silueta, a njene vrijednosti piksela su 1 (crveni kvadratič). Dakle, vrši se minimizacija razlika između piksel vrijednosti projekcija i 1 za svaku projekciju. **Desno:** Pored toga, mora se izvršiti i minimizacija drugog dijela koji maksimizuje distancu između dvije različite projekcije tačaka koje su dobijene projekcijom iz *point cloud-a* ( $p_j$  i  $p_{j'}$ ).

Prvi dio funkcije gubitka se računa kao razlika između 1 i vrijednosti piksela  $\pi_i$  za svaku projekciju  $p_j^i$  na slici siluete  $S_i$ . Koristi se bilinearna interpolacija kako bi se izračunala vrijednost  $\pi_i$  korišćenjem binarnih piksel vrijednosti najbližih piksela oko  $p_j^i$ . Sve projekcije se „guraju“ u unutrašnjost za sve slike siluete minimizacijom sledeće  $L_1$  norme:

$$L_1(\boldsymbol{\pi}_i) = \|1 - \boldsymbol{\pi}_i\|_1(1)$$

Međutim, nemoguće je „gurnuti“ sve projekcije u unutrašnjost minimizacijom  $L_1$  norme. Ukoliko izvršimo optimizaciju *point cloud-a* prema nekoj slici siluete (a) i pri tome krenemo od nekih nasumično

inicijalizovanih tačaka (b), onda ćemo dobiti neadekvatne projekcije tačaka ukoliko koristimo  $L_1$  normu, kao što se vidi na Slici 12.



Slika 12. Data nam je slika siluete (a) i nasumično inicijalizovane projekcije (b). Tada je nemoguće „gurnuti“ projekcije u unutrašnjost siluete (c) zato što standardni prvi dio funkcije gubitka ima problem sa lokalnim minimumom, koji rezultuje neuniformnom dispozicijom projekcija (plave tačkice). Ovaj problem se riješava ugladivanjem originalne siluete kako bi preuzela piksel vrijednosti projekcija i onda se sračunava razlika.

Postoje dva razloga zašto se ovo dešava. Jedan razlog je činjenica da je  $L_1$  norma nediferencijabilna. Čak ukoliko gledamo samo razliku, drugi razlog jeste što možemo samo ispitati intenzitet piksela na osnovu razlike između 1 i interpoliranih piksel vrijednosti  $\pi_i$  koje su bazirane na četiri najbliže binarne piksel vrijednosti. Ovo onemogućuje treniranje ukoliko su projekcije  $p_j^i$  suviše daleko od unutrašnjosti.

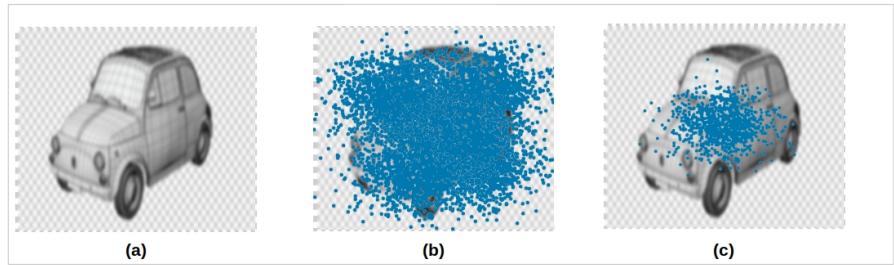
Naš cilj je produkcija ne-nula gradijenata bilo gdje u pozadinskom dijelu, dok piksel vrijednosti u unutrašnjem dijelu ne zahtijevaju modifikaciju. Označimo procesirane slike siluete sa  $S_i^G$ , kako bi napravili razliku između originalne slike siluete  $S_i$  i procesirane slike. Za svaki piksel  $x$  na pozadini slike siluete  $S_i$ , pišemo:

$$S_i^G(x) = \begin{cases} 1, & x \in \mathcal{F} \\ 1 - d(x, \partial \mathcal{F}), & x \in \mathcal{F}^c \end{cases}$$

gdje je  $\mathcal{F} = \{\mathbf{x} \mid \pi_i(\mathbf{x}) = 1\}$  unutrašnjost, dok je  $\bar{\mathcal{F}} = \{\mathbf{x} \mid \pi_i(\mathbf{x}) = 0\}$  pozadina,  $\partial\mathcal{F}$  je granica unutrašnjosti, a  $d(\mathbf{x}, \partial\mathcal{F})$  predstavlja  $L_2$  distancu između  $\mathbf{x}$  i njegovog najbližeg  $\partial\mathcal{F}$ , koji je normalizovan rezolucijom od  $S_i$ .

Normalizacija se takođe izvršava nad procesiranim piksel vrijednostima pozadine kako bi bili u intervalu  $(0, 1)$ . Za ovaj pod-zadatak koristi se *min-max* normalizacija:  $\mathbf{S}_i^G(\bar{\mathcal{F}}) = \text{minmax}(\mathbf{S}_i^G(\mathcal{F}))$ . Konačno, modifikovani prvi dio funkcije gubitka je:

$$\mathcal{M}_1(\pi_i) = \|1 - \pi_i^G\|_1(1)$$



Slika 13. Sa modifikovanim prvim dijelom funkcije gubitka, moguće je „gurnuti“ sve projekcije nasumično inicijalizovanih tačaka (b) sa uglađene slike siluete (a) u unutrašnjost (c) minimizacijom  $\mathcal{M}_1(\pi_i)$ . Plave tačkice predstavljaju projekcije.

Prema Slici 13, korišćenje samo prvog dijela funkcije gubitka dovodi do neuniformnih projekcija tačaka u unutrašnjosti. Kako bi precizno reprezentovali 3D oblik i prekrili siluetu, koristićemo drugi element u funkciji gubitka. Kroz ovaj gubitak, modeluje se prostorna veza između svaka dva para projekcija. Taj gubitak će „gurnuti“ projekcije u unutrašnjost objekta. One bi trebalo da su što udaljenije jedna od druge.

Kako bi se riješio ovaj problem, uvodi se drugi dio u funkciju gubitka koji povećava distancu između parova projekcija koji se nalaze dublje u unutrašnjosti i redukuje ga za parove projekcija koji se nalaze oko granice unutrašnjosti. Preskače projekcije koje se nalaze unutar pozadine.

Za svaki par projekcija  $p_j^i$  i  $p_{j'}^i$ ,  $L_2$  rastojanje se računa na sledeći način:

$$d(p_j^i, p_{j'}^i) = \|p_j^i - p_{j'}^i\|_2, \quad (1)$$

gdje se nakon ovog izračunavanja vrši normalizacija prema rezoluciji slike siluete.

Ovim pristupom se teži da se maksimizuje distanca  $d(p_j^i, p_{j'}^i)$ . Koristi se Gausova funkcija [14] kako bi se dobio gubitak zasnovan na invarijansi strukture koji se smanjuje sa povećanjem distance. Suštinski, uz minimizaciju modifikovanog prvog dijela gubitka  $\mathcal{M}_1$ , vrši se i minimizacija gubitka zasnovanog na invarijansi.

Za svaku projekciju  $p_j^i$ , gubitak zasnovan na invarijansi strukture modeluje njenu prostornu vezu sa svim ostalim projekcijama  $p_{j'}^i$ :

$$\mathcal{L}_2(p_j^i, \{p_{j'}^i\}) = w_j^i \sum_{j'=1}^N \left[ w_{j'}^i \cdot \exp \left( \frac{-d(p_j^i, p_{j'}^i)}{\theta} + \mu_j^i \right) \right], \quad (1)$$

gdje su  $w_j^i$  i  $w_{j'}^i$  težine koje odgovaraju projekcijama  $p_j^i$  i  $p_{j'}^i$  respektivno,  $\theta > 0$  je parametar stapanja,  $\mu_j^i > 0$  je granični bijas za projekciju  $p_j^i$ .

$w_j^i$  govori do kojeg nivoa se projekcija  $p_j^i$  spaja („utapa“) sa pozadinom. Ukoliko je ta težina jednaka nuli, projekcija  $p_j^i$  je takva da je invarijansa strukture kompletno uklonjena, tako da modifikovani prvi dio fukcije gubitka  $\mathcal{M}_1$  momentalno „gura“  $p_j^i$  u unutrašnjost. Parametar stapanja kontroliše interval stapanja (intenzitet invarijanse strukture) date pozadine 3D modela. Granični bijas projekcije  $\mu_j^i$  za projekciju  $p_j^i$  kontroliše distancu do granice unutrašnjosti gdje se invarijansa po toj projekciji redukuje.

Težina  $w_j^i$  se računa korišćenjem bilinearne interpolacije zasnovane na najbližim binarnim vrijednostima piksela na slici siluete  $S_i$ , što je vidljivo na Slici 11. Koriste se gradijenti sa više skala (engl. *multi-scale gradients*) [21] kako bi se sračunao  $\mu_j^i$ . Binarne piksel vrijednosti se ekstrahuju iz susjednih tačaka koje su locirane na tjemena kvadratića na rešetci (oko projekcije  $p_j^i$ ) - Slika 11. Nad njima se vrše interpolacije i uzima se srednja vrijednost svih tih interpolacija kako bi se sračunao  $\mu_j^i$ .

Ovaj pristup progresivno redukuje invarijansu strukture kako se  $p_j^i$  približava granici unutrašnjosti (objekta).

Konačno, Efektivna funkcija gubitka  $E_L$  se računa istovremenom minimizacijom modifikovanog prvog dijela funkcije gubitka  $\mathcal{M}_1$  i drugog dijela funkcije gubitka  $\mathcal{L}_2$  koji je zasnovan na invarijansi strukture. Ukupna greška  $E_L$  se dobija na sledeći način ( $\alpha$  i  $\beta$  se koriste za balansiranje dijelova funkcije gubitka, uprosječavanje se vrši po tačkama  $N$  i pogledima  $G_t$ ):

$$E_L = \frac{\sum_{i=1}^{G_t} \sum_{j=1}^N (\alpha \mathcal{M}_1(\pi_i) + \beta \mathcal{L}_2(p_j^i, \{p_{j'}^i\}))}{G_t \cdot N} \quad (1)$$

Nakon ovog procesa, dobija se 3D *point cloud* koji se potom transformiše u 3D *mesh* koristeći Puasonovu rekonstrukciju površi [12]. Iskorišćen je *GAN* [30] za mapiranje teksture na konkretni 3D *mesh* i produkovanje konačnog izlaza – teksturiranog 3D *mesh*-a koji je zasnovan na ulaznoj teksturi slike, što se vidi na Slici 10. Generator generiše mape pomjeranja (engl. *displacement maps*) i teksture, dok diskriminator diskriminiše između stvarnih / lažnih mapa pomjeranja i tekstura.



## 4. REZULTATI I OSNOVNE INFORMACIJE

U ovom poglavlju osvrnućemo se na glavne rezultate, primarno kroz poređenje sa ostalim pristupima (nadzorovanim i nenadzorovanim). Više detalja i studija slučaja nalaze se u poglavlju 5. Pored toga, u potpoglavlju 4.2 date su osnovne informacije u vezi korišćenih skupova podataka, metrika i link do programskog koda sa upustvom o pokretanju riješenja, te diskusija o mogućim ekstenzijama i ograničenjima ovog pristupa.

### 4.1 Glavni rezultati

Kvantitativni rezultati korišćenjem Čamferovog rastojanja (engl. Chamfer's distance) [22] su prikazani u Tabeli 2. *Point cloud* koji je izgenerisan našom metodom (Naš) nadmašio je svoj voksel ekvivalent (Naš-V) u svim slučajevima. Čamferovo rastojanje se unaprijeđuje sa povećanjem rezolucije. Takođe, nadmašena je prethodna najbolja metoda koja je koristila rendering [8] i DRC metoda [24] (koja je inspirisala ovaj rad).

	Rezolucija 32			Rezolucija 64			Rezolucija 128	
	DRC [25]	DPC [8]	Naš-V	Naš	DPC [8]	Naš	DPC [8]	Naš
Avion	8.35	4.52	4.49	<b>3.99</b>	3.50	<b>3.15</b>	2.84	<b>2.63</b>
Automobil	4.35	4.22	3.75	<b>3.79</b>	2.98	<b>2.86</b>	2.42	<b>2.37</b>
Stolica	8.01	5.10	5.34	<b>4.64</b>	4.15	<b>3.99</b>	3.62	<b>3.46</b>
Proslek	6.90	4.61	4.53	<b>4.14</b>	3.55	<b>3.33</b>	2.96	<b>2.82</b>

Tabela 2. Kvantitativni rezultati predikcije oblika sa poznatom pozom kamere (na *ShapeNet* skupu podataka). Data su Čamferova rastojanja između normalizovanih *point cloud*-ova, pomnoženo sa 100. Korišćene su tri kategorije: Avioni, Automobili i Stolice. Naš rezultat nadmašuje sve ostale metode u pogledu Čamferovog rastojanja. Niža vrijednost je bolja; **bold**-ovano = najbolje.

Dobijeni rezultati nadmašuju *state-of-the-art* diferencijabilne renderere i u Volumetričnoj IoU metriči [20] dok su istovremeno manje vremenski zahtjevni tokom faze treniranja, što se vidi u Tabeli 3. Za automobile, ishod je bolji nego kod renderera zasnovanih na vokselima, ali je veoma sličan sa rendererima koji su zasnovani na *mesh*-evima zato što *mesh*-evi predstavljaju superiornu inicijalnu 3D reprezentaciju za velike površine ravnih površi [10] (kao što su automobile).

	Nenadzorovano učenje			Nadzorovano učenje						
	SoftRas [16]	DIB-R [3]	Naš	P2M [27]	IN [15]	RN [4]	AN [6]	DSN [32]	3DN [28]	ON [18]
Avion	58.4	57.0	<b>62.4</b>	51.5	55.4	42.6	39.2	57.5	54.3	57.1 <b>75.3</b>
Automobil	77.1	<b>78.8</b>	75.6	50.1	74.5	66.1	22.0	74.3	59.4	73.7 <b>75.1</b>
Stolica	49.7	52.7	<b>58.3</b>	40.2	52.2	43.9	25.7	54.3	34.4	50.1 <b>57.8</b>
Prosjek	61.7	62.8	<b>65.43</b>	47.3	60.7	50.9	29.0	62.0	49.4	60.3 <b>64.97</b>

Tabela 3. Kvantitativno Volumetrično IoU [20] poređenje sa diferencijabilnim rendererima za različite 3D reprezentacije i nadzorovane metode (na ShapeNet skupu podataka). Korišćene su tri kategorije: Avioni, Automobili i Stolice. Veća vrijednost je bolja; *bold*-ovano = najbolje.

	3D reprezentacija	Renderovanje	$32^2$ slika 2000 tačaka/ $32^3$ voxel-a	$64^2$ slika 8000 tačaka/ $64^3$ voxel-a	$128^2$ slika 16000 tačaka/ $128^3$ voxel-a
DRC [25]	Voxeli	Da	$\approx 14h$	$\approx 60h$	$\approx 216h$
DPC [8]	Point cloud-ovi	Da	$\approx 14h$	$\approx 24h$	$\approx 72h$
Naš	Point cloud-ovi	Ne	$\approx 6.5h$	$\approx 11h$	$\approx 34.5h$

Tabela 4. Efikasnost vremena treniranja u satima.

Takođe, *FID* vrijednosti za *Mesh-eve* (produkovanе od 3D *point cloud-ova*), Teksturu (ekstrahovanу помоћу *GAN-a*) и Oboje (konačni izlaz – teksturirani 3D *mesh*) su proizvele *state-of-the-art* rezultate.

## 4.2 Osnovne informacije o projektu

*Skupovi podataka.* Korišćeni su sledeći skupovi podataka: ShapeNet<sup>4</sup> [2] (podjela na trening i test skup na osnovu [8]), CUB-200-2011<sup>5</sup> [26] (podjela na trening i test skup na osnovu [10]), te Pascal3D+ skup podataka<sup>6</sup> [31] (podjela na trening i test skup na osnovu [10]).

*Metrike.* Numerička evaluacija za *point cloud-ove* izvršena je koristeći Čamferovo rastojanje [22] između prediktovanih i stvarnih (engl. *ground – truth*) *point cloud-ova*.

Volumetrični IoU [20] se koristi za poređenje 3D rešetki koje su vokselizovane na osnovu prediktovanog *point cloud-a* sa onima koje su vokselizovane na osnovu stvarnih *point cloud-ova*.

*Frechet Inception Distance (FID)* se naširoko koristi kao evaluaciona metrika [7] (ne samo za *GAN-ove* nego i za ovaj problem). *FID* rezultati

<sup>4</sup> <https://shapenet.org/>

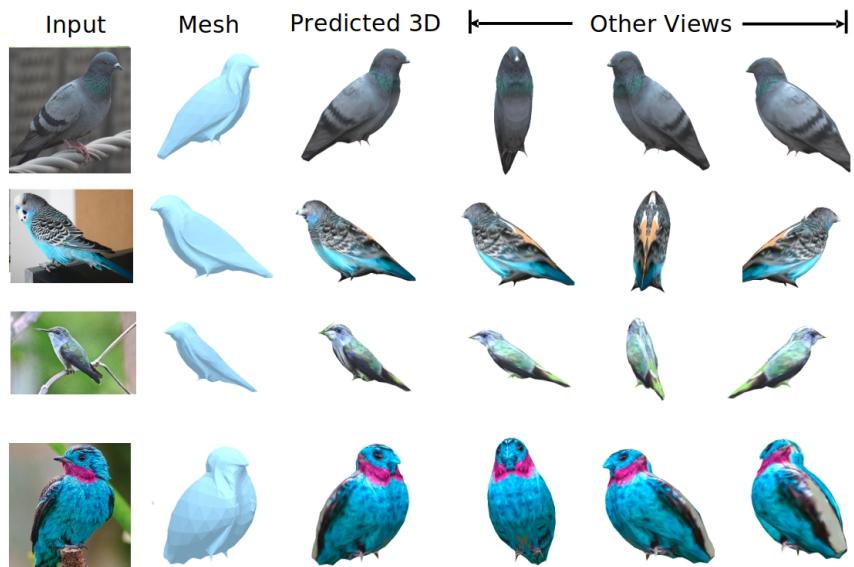
<sup>5</sup> <http://www.vision.caltech.edu/visipedia/CUB-200-2011.html>

<sup>6</sup> <https://cvgl.stanford.edu/projects/pascal3d.html>

evaluiraju 2D projekcije generisanih *point cloud*-ova u *mesh*-eve. 3D *mesh* i teksture se evaluiraju zasebno u ovom procesu.

*Kod.* Implementacija, podaci i trenirani modeli, uz upustvo kako pokrenuti ovaj sistem nalaze se na sledećem linku: <https://github.com/NikolaZubic/2dimage23dmodel>.

*Moguće ekstenzije i ograničenja.* Ovaj rad može se iskoristiti kao dio složenijeg softvera koji se bavi video igrama, animacijom ili bilo kojim aspektom djelovanja gdje je potrebno posjedovati bazni 3D model koji se može dodatno poboljšati procesom skulptovanja (engl. *sculpting*). Moguće je izvršiti proširenje rješenja uzimajući više u obzir uglađene (engl. *smooth*) karakteristike funkcija. Ovaj rad je prvi koji se bavi izazovnim problemom rekonstrukcije 3D modela na osnovu jedne slike bez renderovanja. Rezultati su impresivni, ali ovaj zadatak je daleko od toga da je u potpunosti riješen. Naš model se slabije „snalazi“ sa predikcijom poza kamere koje su rijetke u trening skupu. Takođe, u stanju je da „nauči“ glavne karakteristike oblika svake instance, ali ignoriše neke detalje. Na primjer, noge zebri, krava i konja ne uspije da razdvoji (Slika 20 i Slika 21).



Slika 14. U ovom primjeru su korišćene stvarne 2D slike ptica kao ulazi za generisanje 3D modela. U prvoj koloni je ulaz (engl. *input*) gdje se nalaze stvarne slike ptica, u drugoj koloni, nalazi se generisani 3D *mesh*, te u naredne četiri kolone nalazi se prediktovani 3D model vidljiv u 4 poze.



## 5. STUDIJA SLUČAJA

Ovo poglavlje predstavlja najobimnije poglavlje master rada. U njemu je izvršena detaljna studija slučaja čime je u potpunosti verifikovana validnost ovog složenog metoda. Potpoglavlje 5.1 sadrži detalje o korišćenoj arhitekturi neuronske mreže koja je zasnovana na radu [8], dok su u potpoglavlju 5.2 detaljno opisane metrike koje su korišćene za poređenje sa drugim pristupima. U potpoglavlju 5.3 opisani su eksperimenti i detaljnije prodiskutovani svi rezultati. Ablaciona studija i ispitivanje efikasnosti modela opisano je u potpoglavlju 5.4, a detalji treniranja modela i prikaz dodatnih vizuelnih rezultata dati su i opisani u potpoglavljima 5.5 i 5.6.

### 5.1 Korišćena arhitektura neuronske mreže

Mreža koja je korišćena za 3D rekonstrukciju (u *point cloud* reprezentaciji) na osnovu jedne slike sastavljena je od 2D enkodera i 3D *point cloud* dekodera. 2D enkoder predstavlja 7-slojni CNN. Prvi sloj čini kernel čije su dimenzije  $5 \times 5$  sa 16 kanala i korakom (engl. *stride*) 2. Svaki od preostalih slojeva ima tri kernela i dolazi u parovima, gdje jedan sloj u paru ima korak 2, dok drugi ima korak 1. Broj kanala povećava se sa faktorom 2 nakon svakog sloja sa korakom. Ovi konvolutivni slojevi su propaćeni sa dva potpuno-povezana sloja čije su dimenzije 1024. Oni potom predikuju *point cloud* reprezentaciju. *Point cloud* koji je sastavljen od  $N$  tačaka se predikuje kao vektor čije su dimenzije  $3N$  (koordinate tačke).

Ova arhitektura je odabrana zato što je postigla *state-of-the-art* rezultate za problem 3D rekonstrukcije na osnovu jedne slike, ali je proces diferencijabilnog renderinga bio nepotreban, te je dobijeno preciznije rješenje bez njega. Arhitektura predstavlja optimalno rješenje zato što je u stanju da snažno rekonstruiše podatke iz realnog svijeta, bez obzira na odsustvo preciznih stvarnih poza kamere. Takođe, može se koristiti kao osnova za učenje boja i tekstura, ali to bi zahtijevalo eksplisitno rezonovanje o osvijetljenju i sijenkama.

### 5.2 Detalji o metrikama

*Čamferovo rastojanje.* Numerička evaluacija između prediktovanih i stvarnih (engl. *ground – truth*) *point cloud*-ova izvršena je koristeći

Ćamferovo rastojanje na isti način kao što je to odrađeno u [22]. Rastojanje je izračunato na sledeći način:

$$\begin{aligned} d_{\text{Camfer}}(P_1, P_2) &= \frac{1}{|P_1|} \sum_{r \in P_1} \min_{s \in P_2} \|r - s\|_2 \\ &\quad + \frac{1}{|P_2|} \sum_{s \in P_2} \min_{r \in P_1} \|s - r\|_2, \end{aligned} \quad (1)$$

gdje je  $P_1$  prediktovani *point cloud*, a  $P_2$  je stvarni *point cloud*,  $r$  je tačka na  $P_1$ , a  $s$  je tačka na  $P_2$ .  $|P_1|$  i  $|P_2|$  predstavljaju broj tačaka za point cloud-ove  $P_1$  i  $P_2$ . Prva suma evaluira preciznost prediktovanog *point cloud*-a računanjem koliko je, u prosjeku, udaljena najbliža stvarna tačka od prediktovane tačke. Druga suma mjeri pokrivenost stvarnog sa prediktovanim *point cloud*-om: koliko je, u prosjeku, najbliža prediktovana tačka od stvarne tačke [8].

*Volumetrični IoU.* Kada se vrši poređenje sa rezultatima zasnovanim na vokselima, diskretizuje se 3D prostor, gdje su prediktovani ili stvarni *point cloud*-ovi locirani, u 3D voksel rešetku, gdje je vrijednost voksela jednaka 1 ukoliko sadrži tačku. Volumetrično IoU [20] poređenje se vrši poređenjem 3D rešetke vokselizovane iz prediktovanog *point cloud*-a sa onom koja je vokselizovana iz stvarnog *point cloud*-a.

*Evaluacija teksture.* *Frechet Inception Distance (FID)* se naširoko koristi kao evaluaciona metrika [7] (ne samo kod 2D GAN-ova, nego i za ovaj zadatak). *FID* rezultati evaluiraju 2D projekcije generisanih *point cloud*-ova u *mesh*-eve. 3D *mesh* i teksture su zasebno evaluirani u ovom procesu.

Uz puni *FID*, vršeno je računanje i teksturiranog *FID*-a, gdje su korišćeni *mesh*-evi dobijeni predloženom metodom, te *Mesh FID*, gdje su zamijenjene generisane teksture sa onima koje su pseudo – stvarne. Većinom se za diskutovanje o rezultatima oslanjamо na puni *FID*, ali su individualni konvencionalan alat koji omogućuje analizu kako model reaguje na različite hiperparametre. Generisani primjeri se renderuju na rezoluciji od  $299 \times 299$ , a stvarne slike su skalirane na tu rezoluciju. Date su vizualizacije koje daju više uvida u konceptualne razlike između ovih tipova *FID* metrika, što se vidi na Slici 15.



Slika 15. Slike nad kojima su sračunate *FID* vrijednosti. Generisane su iz ugla koji je odgovarao nasumično odabranoj slici iz trening skupa. U slučaju *Mesh FID*-a, *mesh* je dobijen kao izlaz koristeći pseudo-stvarne tekture slika iz stvarnog svijeta. Kod *Teksturiranog FID*-a, stvarni *mesh* je teksturiran preko *GAN*-a. Što se tiče automobila u krajnjem gornjem-ljevom uglu, za slučaj Punog i *Mesh FID*-a, može se uočiti da *mesh* silueta ima zadovoljavajući kvalitet, ali prave linije i "pruge" predstavljaju nestabilan strukturni efekat koji je fundamentalno uzrokovani *mesh*-om, dok u *Teksturiranom FID* slučaju (koji ne koristi generisane *mesh*-eve) prave linije bolje izgledaju.

### 5.3 Eksperimenti i diskusije

*Detalji o skupu podataka.* Vođeni su eksperimenti koji uključuju 3D oblike u tri kategorije iz *ShapeNet* [2] skupa podataka, uključujući stolice, avione i automobile. Oni se često koriste za evaluaciju od strane drugih autora, te ukoliko metod daje dobre rezultate nad ovim kategorijama, onda možemo reći da će dobro generalizovati i nad ostalim oblicima. Praćen je isti proces za podjelu skupa podataka na trening i test dio kao u radu [8], te je korišćeno pet renderovanih pogleda iz svakog 3D oblika i stvarnih *point cloud*-ova. Preciznije, postoje tri različite rezolucije za renderovane poglede ( $32^2$ ,  $64^2$  i  $128^2$ ), svi odgovaraju istom skupu stvarnih *point cloud*-ova sa različitim brojem tačaka, kao što se vidi u Tabeli 2.

Za *CUB-200-2011* skup podataka [26], korišćena je podjela na trening i test skup na osnovu rada [10]. Ovaj skup podataka čini  $\approx 6k$  trening slika i  $\approx 5.7k$  test slika. Svaka slika ima anotiranu labelu klase (od ukupno 200 klasa). Evaluacija *FID*-a izvršena je nad testnim slikama korišćenjem poza (gdje je to bilo moguće) iz trening skupa, iako je utvrđeno da je *FID* gotovo identičan za oba skupa.

Što se tiče *Pascal3D+* skupa podataka [31], korišćena je ista podjela kao u radu [10] kako bi se trenirao model. 2D *GAN* je treniran samo nad *ImageNet*<sup>7</sup> podskupom (4.7k iskoristivih slika) jer je uočeno da su slike u

<sup>7</sup> <https://www.image-net.org/>

*Pascal3D+* skupu podataka isuviše male za praktične svrhe. Podjela test skupa u radu [10] ne sadrži ni jednu od *ImageNet* slike, pa je evaluacija *FID* rezultata izvršena nad trening slikama, motivisano prethodnom opservacijom *CUB-200-2011* skupa podataka.

*Eksperimentalni detalji.* Korišćena je ista arhitektura kao ona opisana u potpoglavlju 5.1. Ova arhitektura je prvi put predstavljena u radu [8], a u ovom master radu je zamijenjen modul diferencijabilnog renderinga sa Efektivnom funkcijom gubitka  $E_L$ . Njihov pristup koristi strukturno učenje 3D *point cloud*-ova preko parova *RGB* slika. Za svaki par, mreža prvo daje izlaz koji predstavlja *point cloud* reprezentaciju na osnovu prve *RGB* slike i potom renderuje prediktovani *point cloud* iz ugla gledanja druge slike (poza). Njihov modul za diferencijabilni rendering generiše renderovanu sliku siluete i neuronska mreža je trenirana minimizacijom greške zasnovane na pikselima između renderovane slike siluete i siluete druge ulazne slike. Naš pristup uklanja diferencijabilni rendering i eksplatiše projektovane pozicije generisanog *point cloud*-a (tačke) kako bi se kreirao gubitak  $E_L$  neophodan tokom treninga. Tokom testnog vremena, trenirana mreža sa gubitkom  $E_L$  daje kao izlaz 3D *point cloud* na osnovu jedne *RGB* slike.

Prvo, vršeno je poređenje naših rezultata sa pristupima koji koriste rendering na osnovu Čamferovog rastojanja. Svi rendereri sa kojima je vršeno poređenje produkuju siluete prediktovanih oblika kako bi izračunali gubitak vezan za stvarne siluete. Poređenje je vršeno treniranjem mreže koristeći slike siluete za tri različite rezolucije, kao što je već rečeno. Kvantitativni rezultati su predočeni u Tabeli 2. Ovi rezultati pokazuju da naš sistem premašuje sve ostale metode po svim klasama za sve tri rezolucije. Naš pristup je pokazao očiglednu prednost nad diferencijabilnim rendererima koji su zasnovani na vokselima, jer su očuvali više detalja geometrije u memorijski efektivnijem maniru. Dodatno, izostavljanjem procesa renderinga, naš pristup ostvaruje veću preciznost u pogledu rekonstruisanih *point cloud*-ova. Ovi rezultati dalje demonstriraju da je naš metod robustan na promjene u pogledu rezolucije slike i broja tačaka.

Drugo, upoređeni su dobijeni rezultati sa metodama zasnovanim na renderingu za ostale 3D reprezentacije (kao što su *mesh*-evi i voksel rešetke) po IoU metrici. Poređenje uključuje Perspektivne Transformacione Mreže [11], Neuralni *Mesh* Renderer [11], Glatki Rasterajzer [16], te Diferencijabilni Renderer zasnovan na Interpolaciji [3]. Prve dvije metode su bazirane na vokselima, dok su preostale dvije zasnovane na *mesh*-evima. Kako bi produkovali IoU, izvršena je vokselizacija *point cloud*-ova prediktovanih iz slika sa rezolucijom  $128^2$  u Tabeli 2 u voksel rešetke sa

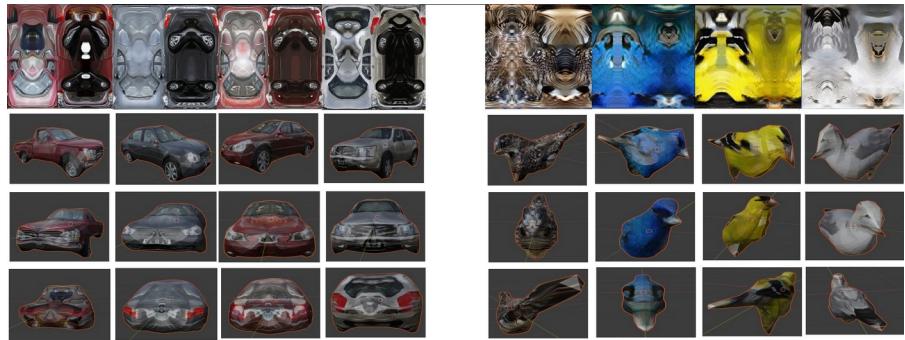
rezolucijom  $32^3$ . Kvantitativno poređenje, dato u tabeli 3, pokazuje značajno nadmašivanje *state-of-the-art* diferencijabilnih renderera u pogledu srednje vrijednosti IoU, gdje su dobijeni do sad najbolji mogući rezultati sa klasama: avion i stolica. Pristupi zasnovani na *mesh*-evima su ograničeni na fiksiranu (većinom sferičnu) *mesh* topologiju. Ovo dovodi do nepravilnosti ukoliko se vrši reprezentacija kompleksnijih površi, kao što su stolice, koje se često prikazuju nesferičnom topologijom.

Konačno, poređeni su dobijeni rezultati sa zadnjim 3D nadzorovanim metodama u Tabeli 3. U prvom eksperimentu, vršeno je poređenje sa Više-prikaznom agregacijom za učenje kategoriski-specifične rekonstrukcije oblika [11]. Rezultati su dobijeni korišćenjem njihovog evaluacionog koda. Prateći isto podešavanje, korišćeni su *point cloud*-ovi rekonstruisani iz ulaznih slika sa rezolucijom  $64^2$  u Tabeli 2. Svaki prediktovani *point cloud* je skaliran tako da je dijagonala njegovog graničnog okvira (engl. *bounding box*) jednaka 1, te potom je izvršeno ponovno semplovanje (engl. *resampling*) stvarnog *point cloud*-a na 8000 tačaka ukoliko bi bio predstavljen sa većim brojem tačaka. Na Tabeli 5 prikazano je da dobijeni rezultati znatno premašuju i Više-prikaznu agregaciju po sve tri posmatrane klase.

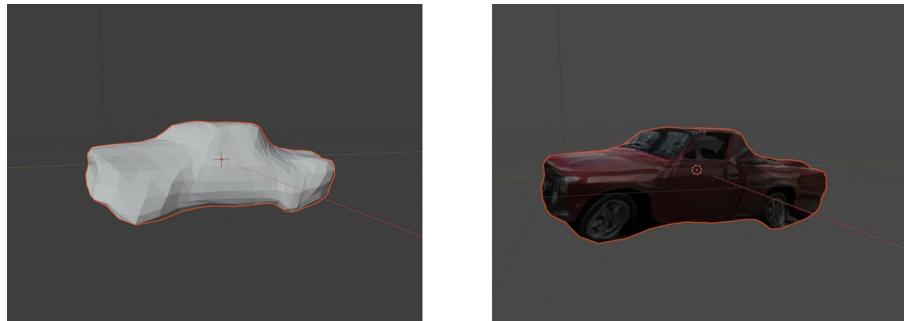
	Automobili	Avioni	Stolice
Više-prikazna agregacija	0.3331	0.2795	0.4637
Naš	<b>0.0342</b>	<b>0.0414</b>	<b>0.0437</b>

Tabela 5. Poređenje sa najnovijom nadzorovanom metodom Više-prikazne agregacije po Ćamferovom rastojanju.

Na slici 16 prikazano je par generisanih, teksturiranih *mesh*-eva koji su renderovani iz više uglova u Blenderu [5], te njihove korespondentne teksture. Rezultati na CUB-200-2011 skupu podataka su u visokoj rezoluciji, ali zadnji dijelovi automobila na Pascal3D+ skupu podataka pokazuju neke neregularnosti. Nakon dalje analize, uočeno je da je skup podataka veoma neizbalansiran, sa samo 20% slika koje prikazuju zadnji dio automobila (većina prikazuje prednji dio). Dakle, ovaj problem moguće je riješiti tako što će se koristiti više trening podataka.



Slika 16. Kvalitativni rezultati na *Pascal3D+* (**lijevo**) i *CUB-200-2011* (**desno**) skupu podataka. Svaki objekat je renderovan iz tri ugla gledanja (u Blenderu), dok prvi red predstavlja teksturu koju je ekstrahovao *GAN*.



Slika 17. Svi generisani 3D modeli mogu se vizualizovati u alatu Blender [5] i posmatrati u realnom vremenu iz korisnički - proizvoljnog ugla. **Lijevo:** 3D *mesh* bez teksture. **Desno:** Teksturirani 3D *mesh* koji predstavlja konačni izlaz ovog modela baziranog na jednoj (u ovom primjeru klasa: automobil) slici.

Skup podataka	Tekst. rezolucija	Uslov	FID (zaokružena $\sigma$ )			
			$\sigma$	Oboje	Tekst.	Mesh
CUB-200-2011	512x512	Bez	1	40.33	43.73	<b>17.89</b>
		Klasa	0.25	33.61	<b>26.67</b>	18.53
	256x256	Klasa	0.25	<b>32.28</b>	29.62	18.88
Pascal3D+	512x512	Bez	1	40.99	30.76	27.17
		Klasa	0.75	<b>26.23</b>	21.89	<b>22.96</b>
		Klasa + Boja	0.5	30.50	<b>21.10</b>	26.86
	256x256	Klasa + Boja	0.5	38.19	25.43	35.71

Tabela 6. *FID* rezultati na *Mesh*-u (dobijen na osnovu 3D *point cloud*-a), Teksturi (ekstrahovana koristeći *GAN* – u tabeli označena skraćeno kao *Teks.*) i Oboje (konačni izlaz – teksturirani 3D *mesh*) grupisani po skupu podataka, rezoluciji tekture i uslovu, zaokruženo na dvije decimale. Manja vrijednost je bolja; *bold*-ovano = najbolje u tom skupu podataka.

## 5.4 Ablaciona studija i efikasnost

Ablaciona studija izvršena je kako bi se potvrdile pretpostavke u pogledu efikasnosti svakog elementa ovog modela koristeći avione sa rezolucijom od 2000 tačaka u Tabeli 2. U tabeli 7, dati su rezultati sa samo nekim dijelovima gubitka, manje pogleda (na primjer,  $G_t = 3$  i  $G_t = 2$ ), te bez težina i bijasa.

	$\mathcal{M}_1$	$\mathcal{L}_2$	Piksel+ $\mathcal{L}_2$	$\mathcal{M}_1 + \text{bez } w_j^i$	$\mathcal{M}_1 + \text{bez } \mu_j^i$	$G_t = 2$	$G_t = 3$	$G_t = 4$
CD	19.50	139.10	24.59	4.58	4.41	4.79	4.54	<b>4.01</b>

Tabela 7. Ablaciona studija je sprovedena korišćenjem Čamferovog rastojanja (engl. *Chamfer's distance*) – CD.

Ova studija pokazuje da gubitak  $E_L$  nije u stanju da nauči strukturu oblika ukoliko koristimo samo  $\mathcal{M}_1$  ili  $\mathcal{L}_2$  gubitak. Takođe, nemoguće je naučiti strukturu korišćenjem standardnog  $L_1$  gubitka zbog problema sa lokalnim minimumom i nediferencijabilnošću. Drugi dio gubitka i njegovi hiperparametri (indikatorske težine i granični bijas) utiču na preciznost rekonstrukcije i efikasnost optimizacije. Parametri  $\alpha$  i  $\beta$  utiču na konflikt i trampu između modifikovanog prvog dijela funkcije gubitka i drugog dijela

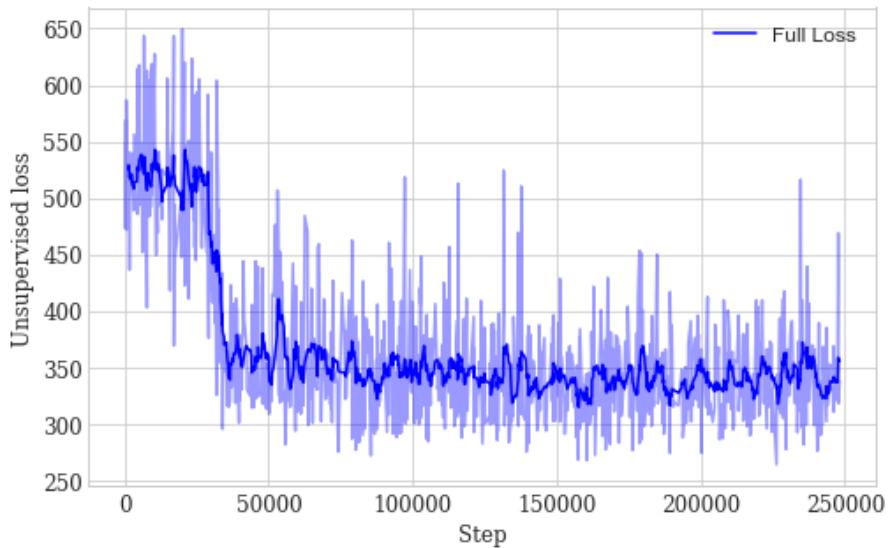
funkcije gubitka zasnovanog na invarijansi strukture. Ukoliko se koristi manje pogleda (uglova gledanja) od  $G_t = 4$  tokom treninga, dolazi do degeneracije performansi struktturnog učenja.

Govoreći o efikasnosti, treniranje ovog modela je poređeno sa *state-of-the-art* diferencijabilnim rendererima za 3D oblike, kao što se vidi u Tabeli 4. Metod zasnovan na vokselima (*DRC*) ima slabosti u pogledu ogromnog računarskog tereta uslijed kubne kompleksnosti voksel rešetki, što ga ograničava da radi samo sa niskim rezolucijama. Te rezolucije su  $32^3$  i  $64^3$  sa sporom stopom konvergencije. Iako je *point cloud* bazirana metoda Insafutdinov-a i Dosovitskiy-a [8] nezavisna od 3D konvolutivnih slojeva (za razliku od *DRC*), rendering procedura još uvijek zahtijeva intenzivno računanje sa diskretnim 3D rešetkama. Dakle, ova metoda zahtijeva više vremena ( $6 \times 10^5$  *mini-batch* iteracija) tokom treniranja nego naša metoda ( $1 \times 10^5$  *mini-batch* iteracija).

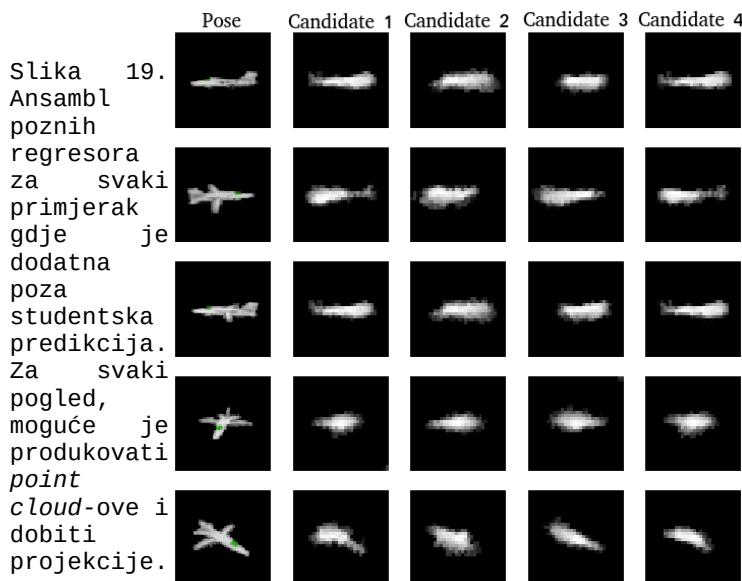
Korišćeni su parametri naučeni tokom različitih koraka tokom treniranja kako bi se rekonstruisao oblik na osnovu korespondentne slike u test skupu. Dodatno, korišćenjem slike iz test skupa demonstrirana je generalizaciona sposobnost koja je naučena tokom optimizacionog procesa, što snažno potvrđuje efikasnost ovog modela. Takođe, pokazano je da se model odlično adaptira i za slike iz realnog svijeta (na primjer, preuzete sa interneta). Više primjera je moguće vidjeti u zadnjem potpoglavlju ovog poglavlja – 5.6.

## 5.5 Detalji treniranja modela

Evaluacija gubitka rađena je koristeći mrežu sa stvarnim pozama kamere tokom projektovanja. Dodatno, korišćene su *RGB* slike sa tri različite rezolucije kako bi se trenirali i evaluirali generisani *point cloud*-ovi u tri rezolucije koje uključuju: 2000, 8000, te 16000 tačaka. Mreža je trenirana preko Adamovog optimizatora (engl. *Adam Optimizer*) sa veličinom *batch*-a koja čini 16 renderovanih slika (4 pogleda i 4 oblika), gdje se iteriralo po  $1 \times 10^5$  *batch*-eva u svakom eksperimentu.



Slika 18. Ukupni nenadzorovani gubitak se smanjuje tokom vremena (kroz više koraka treninga), što je indikator da naš model uči željenu objektivnu funkciju ukoliko prepostavimo da je ona pravilno zadata. Činjenica da je pravilno zadata je potvrđena kroz detaljne eksperimente, te kroz vrijeme, očekivano, opada.



## 5.6 Dodatni prikaz vizuelnih rezultata



Slika 20. Kvalitativni rezultati nad klasama: konji i krave.



Slika 21. Kvalitativni rezultati nad klasama: pingvini i zebre.



## 6. ZAKLJUČAK

U ovom radu, predložena je 3D rekonstrukcijska metoda na osnovu samo jedne slike, za učenje poze i oblika 3D objekata na osnovu samo njihovih 2D projekcija. Kao inicijalna, korišćena je *point cloud* reprezentacija. Nakon toga, izvršena je konverzija te reprezentacije u 3D *mesh*. *Mesh* je potom teksturiran koristeći *GAN* čime je dobijen konačan izlaz – teksturirani 3D *mesh*.

Obimni eksperimenti su pokazali da se *point cloud* reprezentacija odlično uklapa sa reprezentacijama zasnovanim na vokselima, po pitanju performansi i preciznosti. Predloženo okruženje „uči“ da prediktuje oblik, teksturu i pozu na osnovu jedne slike, bez koraka renderovanja, zasnovano je sve samo na 2D projekcijama 3D *point cloud*-ova i njihove prekrivenosti stvarnih silueta. Dok renderovanje zahtijeva iscrpno računanje, naš ključni nalaz jeste da uopšte ne utiče na preciznost u 3D strukturnom učenju.



## 7. BIOGRAFIJA

Nikola Zubić je rođen 13. decembra. 1997. godine u Banjoj Luci. Osnovnu školu „Sveti Sava“ u Brodu završio je 2012. godine, kao nosilac diplome „Vuk Karadžić“, te je bio učesnik mnogobrojnih takmičenja iz matematike, informatike, istorije i geografije. Opštu gimnaziju „Nikola Tesla“ u Brodu upisao je iste te godine, a završio 2016. godine kao učenik generacije i nosilac diplome „Vuk Karadžić“, te je bio učesnik takmičenja iz matematike i osvojio je specijalnu pohvalu na takmičenju „Inost mladih“ koji organizuje Savez Inovatora Republike Srpske. 2016. godine upisuje Fakultet Tehničkih Nauka u Novom Sadu, smjer Softversko Inženjerstvo i Informacione Tehnologije kao budžetski student. Tokom studija, nastavlja da aktivno i dodatno radi na sebi i stiče znanja iz oblasti Vještačke Inteligencije gdje je znanje proširio *online* specijalizacijama i studentskom praksom. Osnovne akademske studije završio je 15.09.2020. godine sa prosječnom ocjenom 9.68 čime je stekao zvanje: „Diplomirani inženjer elektrotehnike i računarstva“.

2020. godine upisao je Master studije iz računarstva na Fakultetu Tehničkih Nauka, smjer: Inteligentni Sistemi. Na ovom smjeru pažljivo je odabrao izborne predmete tako da svi predmeti budu povezani sa Vještačkom Inteligencijom. Od oktobra 2020. do juna 2021. godine obavljao je, *online*, naučno-istraživačku praksu u Računarskoj Laboratoriji Univerziteta u Kembridžu, pod mentorstvom profesora Pijetra Lija. Ova praksa rezultirala je objavljanjem naučno-istraživačkog rada (na osnovu kojeg je zasnovan i sam master rad) koji je prezentovan na AIAI 2021 konferenciji, dio je knjige „*Artificial Intelligence Applications and Innovations*“ koju je objavio Springer, te će proširena verzija rada biti dostupna u časopisu „*Frontiers in AI*“. Na master studijama, položio je sve ispite predviđene planom i programom sa prosječnom ocjenom 10.00.



## 8. LITERATURA

- [1] Ahmed, E., Saint, A., Shabayek, A.E.R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., Ottersten, B.: A survey on deep learning advances on different 3d data representations (2019)
- [2] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository (2015)
- [3] Chen, W., Ling, H., Gao, J., Smith, E., Lehtinen, J., Jacobson, A., Fidler, S.: Learning to predict 3d objects with an interpolation-based differentiable renderer. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32, pp. 9609– 9619. Curran Associates, Inc. (2019), <https://proceedings.neurips.cc/paper/2019/file/f5ac21cd0ef1b88e9848571aeb53551a-Paper.pdf>
- [4] Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction (2016)
- [5] Community, B.O.: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018), <http://www.blender.org>
- [6] Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Atlasnet: A papier-mâché approach to learning 3d surface generation (2018)
- [7] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium (2018)
- [8] Insafutdinov, E., Dosovitskiy, A.: Unsupervised learning of shape and pose with differentiable point clouds. CoRR abs/1810.09381 (2018), <http://arxiv.org/abs/1810.09381>
- [9] Jimenez Rezende, D., Eslami, S.M.A., Mohamed, S., Battaglia, P., Jaderberg, M., Heess, N.: Unsupervised learning of 3d structure from images. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 29, pp. 4996–5004. Curran Associates, Inc. (2016), <https://proceedings.neurips.cc/paper/2016/file/1d94108e907bb8311d8802b48fd54b4a-Paper.pdf>
- [10] Kanazawa, A., Tulsiani, S., Efros, A.A., Malik, J.: Learning category-specific mesh reconstruction from image collections (2018)
- [11] Kato, H., Beker, D., Morariu, M., Ando, T., Matsuoka, T., Kehl, W., Gaidon, A.: Differentiable rendering: A survey (2020)

- [12] Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing. p. 61–70. SGP ’06, Eurographics Association, Goslar, DEU (2006)
- [13] Kumar, T., Verma, K.: A theory based on conversion of rgb image to gray image. International Journal of Computer Applications 7(2), 7–10 (2010)
- [14] Li, Z., Shafiei, M., Ramamoorthi, R., Sunkavalli, K., Chandraker, M.: Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image (2019)
- [15] Liu, J., Lu, H.: Imnet: A learning based detector for index modulation aided mimo-ofdm systems (2019)
- [16] Liu, S., Chen, W., Li, T., Li, H.: Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction (2019)
- [17] Loper, M.M., Black, M.J.: OpenDr: An approximate differentiable renderer. In: European Conference on Computer Vision. pp. 154–169. Springer (2014)
- [18] Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space (2019)
- [19] Nguyen-Phuoc, T., Li, C., Balaban, S., Yang, Y.L.: Rendernet: A deep convolutional network for differentiable rendering from 3d shapes (2019)
- [20] Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision (2020)
- [21] Sreegadha, G.: Image interpolation based on multi scale gradients. Procedia Computer Science 85, 713–724 (2016). <https://doi.org/https://doi.org/10.1016/j.procs.2016.05.258>, <https://www.sciencedirect.com/science/article/pii/S1877050916306081>, international Conference on Computational Modelling and Security (CMS 2016)
- [22] Sun, X., Wu, J., Zhang, X., Zhang, Z., Zhang, C., Xue, T., Tenenbaum, J.B., Freeman, W.T.: Pix3d: Dataset and methods for single-image 3d shape modeling (2018)
- [23] Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs (2017)
- [24] Tulsiani, S., Efros, A.A., Malik, J.: Multi-view consistency as supervisory signal for learning shape and pose prediction (2018)
- [25] Tulsiani, S., Zhou, T., Efros, A.A., Malik, J.: Multi-view supervision for single-view reconstruction via differentiable ray consistency (2017)

- [26] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.J.: The caltech-ucsd birds-200-2011 dataset (2011)
- [27] Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images (2018)
- [28] Wang, W., Ceylan, D., Mech, R., Neumann, U.: 3dn: 3d deformation network (2019)
- [29] Wu, J., Wang, Y., Xue, T., Sun, X., Freeman, W.T., Tenenbaum, J.B.: Marrnet: 3d shape reconstruction via 2.5d sketches (2017)
- [30] Xian, W., Sangkloy, P., Agrawal, V., Raj, A., Lu, J., Fang, C., Yu, F., Hays, J.: Texturegan: Controlling deep image synthesis with texture patches (2018)
- [31] Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: A benchmark for 3d object detection in the wild. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2014)
- [32] Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: Disn: Deep implicit surface network for high-quality single-view 3d reconstruction (2019)



## KLJUČNA DOKUMENTACIJSKA INFORMACIJA

<b>Redni broj, RBR:</b>	
<b>Identifikacioni broj, IBR:</b>	
<b>Tip dokumentacije, TD:</b>	monografska publikacija
<b>Tip zapisa, TZ:</b>	tekstualni štampani dokument
<b>Vrsta rada, VR:</b>	master rad
<b>Autor, AU:</b>	Nikola Zubić
<b>Mentor, MN:</b>	Dr Dragan Ivetić, redovni profesor
<b>Naslov rada, NR:</b>	Efektivna funkcija gubitka za generisanje 3D modela na osnovu jedne 2D slike upotrebom dubokog učenja
<b>Jezik publikacije, JP:</b>	srpski
<b>Jezik izvoda, JI:</b>	srpski / engleski
<b>Zemlja publikovanja, ZP:</b>	Srbija
<b>Uže geografsko područje, UGP:</b>	Vojvodina
<b>Godina, GO:</b>	2020
<b>Izdavač, IZ:</b>	autorski reprint
<b>Mesto i adresa, MA:</b>	Novi Sad, Fakultet tehničkih nauka, Trg Dositeja Obradovića 6
<b>Fizički opis rada, FO:</b>	8 / 61 / 32 / 7 / 21 / 0 / 0
<b>Naučna oblast, NO:</b>	Softversko inženjerstvo i informacione tehnologije
<b>Naučna disciplina, ND:</b>	Računarska grafika i vizija
<b>Predmetna odrednica / ključne reči, PO:</b>	računarska grafika, računarska vizija, vještačka inteligencija, neuronske mreže
<b>UDK</b>	
<b>Čuva se, ČU:</b>	Biblioteka Fakulteta tehničkih nauka, Trg Dositeja Obradovića 6, Novi Sad
<b>Važna napomena, VN:</b>	
<b>Izvod, IZ:</b>	Zadatak rada je specifikacija, implementacija i evaluacija sistema koji je u stanju da izvrši rekonstrukciju 3D modela na osnovu jedne slike, koristeći samo njihove 2D projekcije, što je nova ideja, tj. doprinos rada. Rad je baziran na već postojećoj arhitekturi uz novu funkciju gubitka koja prevazilazi performanse svih prethodnih pristupa.
<b>Datum prihvatanja teme, DP:</b>	28.06.2021
<b>Datum odbrane, DO:</b>	01.07.2021
<b>Članovi komisije, KO:</b>	
<b>predsednik</b>	Dr Aleksandar Kupusinac, redovni profesor
<b>član</b>	Dr Ksenija Doroslovački, vanredni profesor
<b>mentor</b>	Dr Dragan Ivetić, redovni profesor
	Potpis mentora



## KEY WORDS DOCUMENTATION

Accession number, <b>ANO:</b>	
Identification number, <b>INO:</b>	
Document type, <b>DT:</b>	monographic publication
Type of record, <b>TR:</b>	textual material
Contents code, <b>CC:</b>	MSc thesis
Author, <b>AU:</b>	Nikola Zubić
Mentor, <b>MN:</b>	Dragan Ivetić, full professor, PhD
Title, <b>TI:</b>	An Effective Loss Function for Generating 3D Models from Single 2D Image without Rendering
Language of text, <b>LT:</b>	Serbian
Language of abstract, <b>LA:</b>	Serbian / English
Country of publication, <b>CP:</b>	Serbia
Locality of publication, <b>LP:</b>	Vojvodina
Publication year, <b>PY:</b>	2020
Publisher, <b>PB:</b>	author's reprint
Publication place, <b>PP:</b>	Novi Sad, Faculty of Technical Sciences, Trg Dositeja Obradovića 6
Physical description, <b>PD:</b>	8 / 61 / 32 / 7 / 21 / 0 / 0
Scientific field, <b>SF:</b>	Software Engineering and Information Technologies
Scientific discipline, <b>SD:</b>	Computer Graphics & Vision
Subject / Keywords, <b>S/KW:</b>	computer graphics, computer vision, artificial intelligence, neural networks
<b>UDC</b>	
Holding data, <b>HD:</b>	Library of the Faculty of Technical Sciences, Trg Dositeja Obradovića 6, Novi Sad
Note, <b>N:</b>	
Abstract, <b>AB:</b>	The paper deals with specification, implementation, and evaluation of the system which reconstructs a 3D model based on a single 2D image, by using only their 2D projections, which is a novel idea, ie. the main contribution of the paper. The work is based on the already existing architecture with a new loss function that surpassed the performance of all previous approaches.
Accepted by sci. Board on, <b>ASB:</b>	28.06.2021
Defended on, <b>DE:</b>	01.07.2021
Defense board, <b>DB:</b>	
president	Aleksandar Kupusinac, full professor, PhD
member	Ksenija Doroslovački, associate professor, PhD
mentor	Dragan Ivetić, full professor, PhD
	Mentor's signature

