Kauno technologijos universitetas

Informatikos fakultetas

3D objektų rekonstrukcija naudojant giliojo mokymosi algoritmus

Baigiamasis magistro studijų projektas

Algirdas Pocius

Projekto autorius

dr. Eglė Butkevičiūtė

Vadovė

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

3D objektų rekonstrukcija naudojant giliojo mokymosi algoritmus

Baigiamasis magistro studijų projektas

Dirbtinio intelekto informatika (6211BX007)

Algirdas Pocius

Projekto autorius

dr. Eglė Butkevičiūtė

Vadovė

doc. Gintaras Palubeckis

Recenzentas

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

Algirdas Pocius

3D objektų rekonstrukcija naudojant giliojo mokymosi algoritmus

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Algirdas Pocius

Patvirtinta elektroniniu būdu

Pocius, Algirdas. 3D objektų generavimas naudojant giliojo mokymosi algoritmus. Magistro studijų baigiamasis projektas vadovė dr. Eglė Butkevičiūtė; Kauno technologijos universitetas, informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, informatika (B01).

Reikšminiai žodžiai: Gilusis mokymasis, 3D objektai, „Pix2Vox“, Vokseliais pagrįsta rekonstrukcija, Duomenų trikdymas.

Kaunas, 2025. 57 p.

Santrauka

3D objektų atkūrimas iš vieno dvimačio vaizdo yra labai aktualus uždavinys, plačiai taikomas virtualioje realybėje, autonominėse sistemose ir robotikoje. Nors daug tyrimų skirta tinklo architektūros modifikavimui, santykinai mažiau dėmesio skiriama duomenų rengimo strategijoms ir mokymo rinkinių dydžiams. Šiame straipsnyje pristatomame darbe analizuojama, kaip gerai paruoštas mokymo rinkinys, gausus daugelio vaizdų duomenų šaltinis (pvz., „ShapeNet“) ir tinklo pritaikymo sprendimai gali pagerinti rekonstruojamų 3D modelių tikslumą. Rezultatai vertinami naudojant standartines metrikas, pavyzdžiui, susikirtimų per sąjungą (IoU).

Pocius, Algirdas. 3D Object generation using deep learning algorithms. Master's Final Degree Project supervisor dr. Eglė Butkevičiūtė; Faculty of informatics, Kaunas University of Technology.

Study field and area (study field group): Computer science, Informatics (B01)

Keywords: Deep learning, 3D objects, Pix2Vox, Voxel-based reconstruction, Data perturbation.

Kaunas, 2025. 57 p.

Summary

Reconstructing 3D objects from a single two-dimensional image is a highly relevant task with a wide range of applications in virtual reality, autonomous systems and robotics. While much research focuses on modifying network architectures, relatively less attention has been given to data preparation strategies and training set sizes. The work presented in this paper analyses how a well-prepared training set, a rich source of multi-view data (e.g., ShapeNet) and network adaptation solutions can improve the accuracy of reconstructed 3D models. Results are evaluated using standard metrics such as Intersection-over-Union (IoU).

Turinys

Paveikslų sąrašas

Lentelių sąrašas

Santrumpų ir terminų sąrašas

GAN – Generative Adversarial Network; dviejų neuroninių tinklų (generatoriaus ir diskriminatoriaus) sistema, kuri mokosi generuoti sintetinius duomenis.

CNN – Convolutional Neural Network; konvoliucinių sluoksnių pagrindu veikiantis giliojo mokymosi tinklas, plačiai taikomas vaizdų ir 3D duomenų apdorojimui.

RNN – Recurrent Neural Network; pasikartojantis neuroninis tinklas, skirtas sekoms, kur kiekviena būsena priklauso nuo ankstesnės.

IoU – Intersection-over-Union; metrika, skaičiuojanti prognozuotos ir tikros tūrio ar ploto sankirtos ir sąjungos santykį.

CD – Chamfer Distance; vidutinio atstumo tarp dviejų taškų debesų metrika, naudojama paviršiaus panašumui įvertinti.

EMD – Earth Mover's Distance; optimalaus „masės perkėlimo" atstumas tarp dviejų taškų pasiskirstymų.

MSE – Mean Square Error; kvadratinių skirtumų vidurkis tarp prognozuotų ir tikrų reikšmių.

BCE – Binary Cross-Entropy; dvejetainė kryžminė entropija, nuostolių funkcija klasifikacijai ir vokselinių tūrių rekonstrukcijai.

F-score – Harmoningas tikslumo ir pasikartojimo vidurkis, taikomas vertinant atkurtų paviršių sutapimą.

PSNR – Peak Signal-to-Noise Ratio; santykis, matuojantis signalo kokybę, dažnai naudojamas vaizdų rekonstrukcijai.

GPU – Graphics Processing Unit; vaizdo plokštės procesorius, spartinantis giliojo mokymosi skaičiavimus.

CPU – Central Processing Unit; bendros paskirties procesorius, valdantis kompiuterio operacijas.

JSON – JavaScript Object Notation; lengvas duomenų mainų formatas, naudojamas anotacijoms saugoti.

Blender – atvirojo kodo 3D kūrimo programa, naudota sintetiniams vaizdams generuoti.

PyTorch – atvirojo kodo gilaus mokymosi biblioteka, naudota tinklų kūrimui ir mokymui.

Voxel (liet. Vokselis) – tūrinis 3D pikselis kubinėje gardelėje; naudojamas objekto tūriui reprezentuoti.

Point Cloud (liet. Taškų debesis) – netvarkingas 3D taškų rinkinys, apibrėžiantis objekto paviršių.

Mesh (liet. Tinklas) – viršūnių, briaunų ir paviršių tinklas, apibrėžiantis objekto geometriją.

Neural Field (liet. Neuroninis laukas) – pastovus funkcijos vaizdavimas neuroniniu tinklu, kuris grąžina, ar taškas priklauso objektui.

Octree – hierarchinė duomenų struktūra, kuri rekursyviai dalija erdvę į aštuonias dalis, taupant atmintį retuose vokseliuose.

Sparse Convolution – konvoliucinis skaičiavimas, atliekamas tik ne-nulinėse gardelės vietose, mažinant resursus.

Diffusion Model (liet. Difuzijos modelis) – tikimybinis generatyvinis modelis, kuris laipsniškai pašalina triukšmą, kad sukurtų naujus duomenis.

DDPM – Denoising Diffusion Probabilistic Model; difuzijos modelio formulė, kurioje mokomasi invertuoti triukšmo procesą.

SDS – Score Distillation Sampling; metodas, naudojamas 3D Gaussian Splatting optimizacijai.

Gaussian Splatting – 3D scenos rekonstrukcijos technika, atvaizduojanti paviršių kaip Gaussinių branduolių rinkinį.

Konvoliucinis sluoksnis (Convolutional layer) – filtravimo sluoksnis naudojamas CNN architektūroje, kad apmokyti dirbtinį intelektą. Lietuviškai dažnai vadinamas konvoliuciniu.

Batch size (liet. Partijos dydis) – dydis nurodantis kiekį duomenų kurį apdoroja mokymosi procese, vienoje partijoje.

Epoch (liet. Epocha) – vienas pilnas viso treniravimo duomenų rinkinio perėjimas per tinklą.

Loss Function (liet. Nuostolių funkcija) – kriterijus, matuojantis skirtumą tarp tinklo išvesties ir pagrindinės tiesos; naudojamas mokymo procese optimizavimui.

Mode Collapse – GAN mokymo problema, kai generatorius sukuria ribotą išvesties įvairovę.

Posterior Collapse – VAE problema, kai dekoderis ignoruoja latentinę erdvę, sumažindamas modelio gebėjimą generuoti.

Įvadas

Gilusis mokymasis per pastarąjį dešimtmetį sukėlė revoliuciją įvairiose kompiuterinės regos ir grafikos srityse, įskaitant 3D modeliavimo sritį. 3D objektų generavimas iš vienos nuotraukos tapo itin svarbiu uždaviniu, turinčiu plačias pritaikymo galimybes. Nuo virtualios ir papildytos realybės iki autonominių transporto priemonių ir robotikos – gebėjimas generuoti tikslius 3D modelius iš ribotos 2D informacijos atveria naujas galimybes interaktyviose ir išmaniosiose sistemose.

Pastaraisiais metais mokslininkai pristatė daugybę metodų, skirtų šiai problemai spręsti. Pavyzdžiui, Choy ir kt.[1] pristatė 3D-R2N2 modelį, kuris naudoja pasikartojantį neuroninį tinklą 3D objektams rekonstruoti iš vieno ar kelių vaizdų. Wang ir kt.[2] pasiūlė Pixel2Mesh metodą, kuris generuoja 3D tinklelius iš vieno RGB vaizdo, naudojant konvoliucinius neuroninius tinklus ir grafų konvoliucijas. Be to, difuzijos modeliai, tokie kaip Magic3D[3], pradėjo rodyti pažadus generuojant aukštos kokybės 3D turinį iš tekstinės informacijos.

Tačiau nepaisant šių pažangų, 3D objektų generavimas iš vieno vaizdo vis dar susiduria su daugybe iššūkių. Ribota informacija, gaunama iš vieno vaizdo, apsunkina tikslaus objekto formos ir struktūros atkūrimą. Be to, sudėtingos tekstūros, apšvietimo sąlygos ir objektų persidengimas gali dar labiau komplikuoti generavimo procesą. Difuzijos modeliai, tokie kaip Denoising Diffusion Probabilistic Models[4], nors ir pasiekė puikių rezultatų 2D vaizdų generavime, jų pritaikymas 3D srityje dar yra tyrimų objektas.

Tikslas: ištirti giliojo mokymosi pagrindu veikiančius kompiuterinės regos ir grafikos algoritmus, generuojančius tikslius 3D objektus, ir įvertinti jų efektyvumą skaitmeninio modeliavimo srityje.

3D objektų generavime egzistuojančios problemos:

- **Geometrija**: Modeliai dažnai susiduria su sunkumais atvaizduojant sudėtingas geometrijas, ypač kai objektai turi smulkių detalių, plonų struktūrų ar netaisyklingų formų. Dėl to generuojami 3D modeliai gali prarasti svarbias savybes, kurios yra kritiškos tam tikrose taikymo srityse, pavyzdžiui, medicininiuose vaizduose ar inžineriniuose brėžiniuose.
- **Permatomi/uždengti modeliai**: Daug objektų gali būti permatomi arba iš dalies uždengti kitų objektų, o tai gali apsunkinti modeliui tiksliai atkurti visą objekto formą ir struktūrą. Be to, šešėliai ir atspindžiai gali klaidinti modelį, nes jie gali būti interpretuojami kaip objekto dalis.
- **Fonas**: Kompleksinis ar triukšmingas fonas gali trukdyti modeliui atskirti objektą nuo aplinkos. Pavyzdžiui, jei objektas yra fotografuotas mieste ar gamtoje, aplinkiniai elementai gali būti interpretuojami kaip objekto dalis, kas sumažina generuojamo modelio tikslumą.
- **Modelio sudėtingumas**: Giliojo mokymosi modeliai gali tapti labai sudėtingi, turintys milijonus parametrų. Tai apsunkina modelio mokymą, reikalauja daug skaičiavimo išteklių ir gali sukelti per didelį pritaikymą prie mokymo duomenų (angl. overfitting), kas sumažina modelio gebėjimą generalizuoti.
- **Skaičiavimo sąnaudos**: 3D objektų generavimas yra skaičiavimo požiūriu brangus procesas, ypač kai dirbama su aukštos rezoliucijos modeliais. Tai gali būti kliūtis realaus laiko programose, kur greitas atsakas yra kritinis, pavyzdžiui, žaidimuose ar interaktyviose simuliacijose.

- **Universalumas**: Dauguma esamų modelių yra mokomi konkrečioms objektų kategorijoms ir gali nesugebėti generalizuoti naujiems objektams ar klasėms. Universalus modelis, galintis generuoti įvairius objektus iš skirtingų kategorijų, yra sudėtingas iššūkis.
- **Apmokymo greitis ir stabilumas**: 3D objektų generavimo modelių apmokymas gali trukti ilgai, kartais net dienomis ar savaitėmis, priklausomai nuo duomenų rinkinio dydžio ir modelio sudėtingumo. Be to, mokymo procesas gali būti nestabilus, ypač naudojant GAN architektūras, kurios yra linkusios į modų griūtį (angl. mode collapse), kai modelis generuoja ribotą duomenų įvairovę.

Tikslui pasiekti yra iškeltos šios užduotys:

1. **Duomenų rinkimas ir apdorojimas**: Surinkti ir paruošti aukštos kokybės duomenų rinkinį, kuris apimtų įvairias objektų kategorijas ir formas. Atlikti duomenų valymą, normalizaciją ir papildymą, siekiant pagerinti modelio mokymą ir generalizaciją.
2. **Modelio architektūros parinkimas ir įgyvendinimas**: Pasirinkti tinkamą gilaus mokymosi architektūrą, kuri būtų efektyvi ir stabiliai mokytųsi generuoti 3D objektus iš vieno vaizdo. Tai gali būti egzistuojančio modelio pritaikymas arba naujos architektūros kūrimas.
3. **Modelio perturbacijos**: Nustatyti modelio hiperparametrus, pakeisti architektūros sluoksnius ar keisti naudojamų duomenų kiekius ir atsižvelgti į gautus rezultatus.
4. **Tikslumo vertinimas**: Parengti išsamų modelio vertinimo planą, naudojant Intersection-over-Union, ir palyginti rezultatus su kitais metodais literatūroje.
5. **Rezultatų analizė ir dokumentavimas**: Analizuoti gautus rezultatus, identifikuoti modelio privalumus ir trūkumus, pateikti rekomendacijas tolimesniems tyrimams, ir parengti išsamią ataskaitą apie atliktą darbą.

1. 3D objekto generavimui naudojamų įrankių ir technologijų analizė

1.1. Duomenų rinkinys

Visiems bandymams yra parinktas duomenų rinkinys **„ShapeNetCore“**[5]. Šis duomenų rinkinys yra plačiai naudojamas kompiuterinės regos ir 3D modeliavimo srityse dėl savo išsamumo ir įvairovės. Jame yra 55 skirtingos kategorijos ir apie 51,300 unikalių 3D modelių, apimančių tokius objektus kaip lėktuvai, kėdės, automobiliai ir kt. Kiekvienai modelio kategorijai yra priskirtas atitinkamas identifikacijos numeris, sugeneruotas pagal „WordNet 3.0“ sinonimų poslinkį (angl. synset offset). Kiekviena kategorija turi priskirtus skirtingus modelius, kurie yra sužymėti pagal to modelio originalios saugyklos identifikatorius. Giliau yra sutalpintas konkretus modelis ir atitinkami parametrai, apibūdinantys tą modelį.

1.1 pav. Kėdės modelis iš „ShapeNetCore“

Duomenų rinkinys yra esminis veiksnys mokant giliojo mokymosi modelius, nes nuo jo priklauso modelio gebėjimas generalizuoti ir atpažinti įvairias objekto formas. Be to, šis duomenų rinkinys suteikia galimybę palyginti skirtingus modelius, nes jis naudojamas daugelyje tyrimų[5][6]. Duomenų rinkinys yra organizuotas hierarchiškai, kur aukščiausiame lygmenyje yra kategorijos, o kiekvienoje kategorijoje yra daug 3D modelių. Kiekvienas modelis turi savo unikalų identifikatorių. „ShapeNetCore“ duomenų rinkinys yra žinomas dėl savo kokybės ir įvairovės. Jame yra tiek paprastų, tiek sudėtingų objektų, turinčių įvairias formas ir struktūras. Tai leidžia mokyti modelius, kurie gali generalizuoti ir pritaikyti įvairioms situacijoms. Tačiau svarbu atkreipti dėmesį į tai, kad duomenų rinkinyje gali būti ir netikslumų ar trūkumų, todėl duomenų apdorojimas yra esminis žingsnis prieš modelio mokymą. Siekiant užtikrinti sąžiningą modelio vertinimą, duomenų rinkinys yra padalijamas į mokymo, validavimo ir testavimo aibes. Dažnai naudojamas santykis yra 80% mokymui, 20% testavimui. Tai leidžia modelį mokyti ant vieno duomenų rinkinio, hiperparametrus derinti pagal rezultatus. Taipogi yra atitinkami duomenų rinkinio privalumai ir trūkumai

Privalumai:

- **Įvairovė**: Daug kategorijų ir modelių, leidžiančių mokyti universalų modelį.
- **Standartas**: Plačiai naudojamas tyrimuose, leidžiantis palyginti rezultatus su kitais darbais.

Trūkumai:

- **Netikslumai**: Gali būti modelių su klaidomis ar trūkumais.
- **Dydis**: Didelis duomenų rinkinio dydis reikalauja daug saugojimo vietos ir gali apsunkinti apdorojimą.

Taipogi 1.1 lentelėje yra pavaizduotas duomenų rinkinyje surinktų kategorijų ir esančių modelių kiekio fragmentas.

1.1 lentelė. Naudojamo ShapeNet duomenų rinkinio sudėtis

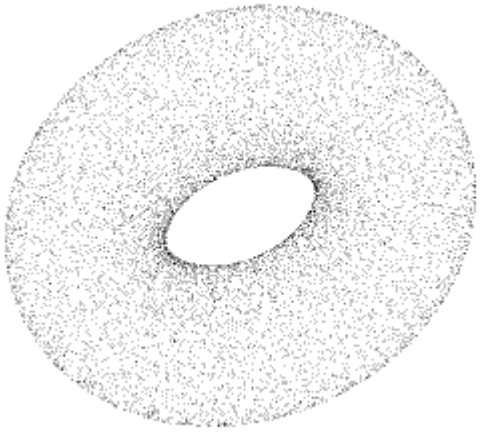
Identifikatorius	Kategorija	Modelių kiekis
02691156	Lėktuvas	4045
02828884	Suolas	1816
02933112	Komoda	1572
02958343	Automobilis	7496
03001627	Kėdė	6778
03211117	Monitorius	1095
03636649	Lempa	2318
03691459	Kolonėlės	1618
04090263	Šautuvas	2373
04256520	Sofa	3173
04379243	Stalas	8509
04401088	Telefonas	1052
04530566	Laivas	1939

1.2. 3D objektų generavimas

3D objektų generavimas iš vieno arba kelių 2D vaizdų yra sudėtingas uždavinys, kuris reikalauja išspręsti dviprasmiškumą, kylančią dėl informacijos praradimo pereinant iš 3D į 2D[7]. Kai objektas yra projekuojamas į 2D vaizdą, prarandama gylio informacija ir kai kurios erdvinės savybės, todėl atkurti originalią 3D formą iš vieno vaizdo yra iššūkis. Pastaraisiais metais giliojo mokymosi metodai, ypač gilieji neuroniniai tinklai, tapo galingu įrankiu šiai problemai spręsti. Jie gali išmokti sudėtingus nelinearinius ryšius tarp 2D vaizdo ir atitinkamos 3D formos, naudodami didelius duomenų rinkinius ir pažangias architektūras.

1.2.1. Taškų debesis (angl. Point Cloud)

Taškų debesis vaizduoja geometrinę figūrą, paprastai jos paviršių, kaip 3D koordinačių rinkinį Euklidinėje erdvėje[6]. 3D formatu šios vietos apibrėžiamos jų x, y, z koordinatėmis. Taigi, taškų debesies atvaizduojamas objektas arba scena yra $N\times3$ matrica, kur N - taškų skaičius, dar vadinamas taškų debesies skiriamąja geba.



1.2 pav. Taškų debesies pavyzdys [1]

Taškų debesys yra efektyvūs ir gali tiksliai atvaizduoti sudėtingas formas. Tačiau jie neturi aiškios topologijos, todėl sunkiau taikyti standartinius konvoliucinius tinklus. Siekiant įveikti šią problemą, buvo pasiūlyti specialūs tinklai, tokie kaip PointNet[7], kurie tiesiogiai veikia su taškų debesimis.

Taškų debesų privalumai:

- **Efektyvumas:** Reikalauja mažiau atminties nei vokseliai ar tinkleliai, nes saugo tik paviršiaus taškus.
- **Lankstumas:** Gali tiksliai atvaizduoti sudėtingas formas ir struktūras.
- **Plačiai naudojami:** Taškų debesys yra standartinė reprezentacija daugelyje sričių, įskaitant robotiką, autonomines transporto priemones ir virtualią realybę.

Tačiau taškų debesys taip pat turi trūkumų:

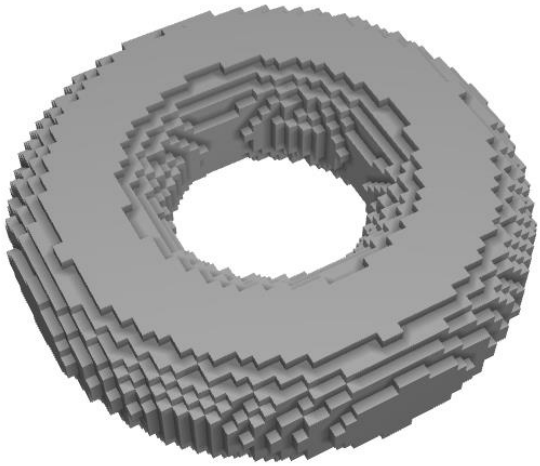
- **Nėra topologijos:** Taškai nėra susieti tarpusavyje, todėl nėra informacijos apie paviršiaus ryšius ar plokštumas.
- **Netvarka:** Taškų debesys yra netvarkingi duomenų rinkiniai, todėl reikia specialių metodų, kad būtų galima juos apdoroti ir analizuoti.

Kadangi taškų debesys neturi nustatytos tvarkos ir topologijos, giliojo mokymosi modeliai turi būti sukurti taip, kad būtų nevarijuotu taškų permutacijoms ir galėtų efektyviai išgauti erdvinę informaciją. PointNet[7] yra vienas iš pirmųjų modelių, tiesiogiai veikiantis su taškų debesimis. Jis naudoja simetrišką funkciją (pvz., maksimalią agregaciją), kad sujungtų informaciją iš visų taškų ir sukurtų globalų objekto aprašymą. Vėlesni modeliai, tokie kaip PointNet++[8] ir DGCNN[9], tobulino šią idėją, įtraukdami vietines kaimynystes ir dinamiškus grafų ryšius, leidžiančius modeliui geriau išmokti vietines struktūras.

[1] https://upload.wikimedia.org/wikipedia/commons/4/4c/Point_cloud_torus.gif

1.2.2. Vokselis (angl. Voxel)

Vokseliai yra panašūs į 3D pikselius, kurių kiekvienas atskiras taškas yra 3D erdvėje. Šie kubiniai vienetai yra suskirstyti į tinklelio struktūrą, o kiekvienam vokseliui priskiriama reikšmė, rodanti, ar jame yra objektas, ar ne. Toks dvejetainis atvaizdavimas leidžia vokseliuose veiksmingai perteikti sudėtingų 3D struktūrų esmę.



1.**3 pav.** Vokselio pavyzdys [2]

Vokselių pagrindu veikiantys metodai leidžia lengvai taikyti 3D konvoliucinius tinklus. Tačiau didėjant rezoliucijai, vokselių skaičius auga kubiškai, todėl reikalingi dideli skaičiavimo resursai[10]. Dėl to dažnai naudojamos žemos rezoliucijos vokselių reprezentacijos, kurios gali prarasti smulkias detales.

Privalumai:

- **Reguliari struktūra:** Leidžia naudoti standartinius 3D CNN metodus.
- **Tūrinė informacija:** Galima atvaizduoti tiek objekto paviršių, tiek vidinę struktūrą.

Trūkumai:

- **Atminties sąnaudos:** Didėjant rezoliucijai, vokselių skaičius auga kubiškai, reikalaujant daug atminties.
- **Skaičiavimo sąnaudos:** Didelis vokselių skaičius padidina skaičiavimo laiką ir resursus.

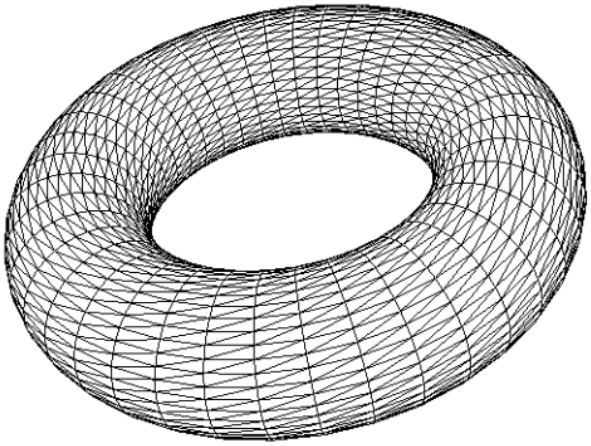
Norint sumažinti skaičiavimo sąnaudas, taikomi tokie metodai kaip:

- **Octree struktūros**[11]**:** Hierarchinė vokselių struktūra, leidžianti efektyviai reprezentuoti retus vokselių duomenis.
- **Sparse convolution**[12]**:** Naudojamos retų duomenų struktūros, kad būtų apdorojami tik ne-nuliniai vokseliai.

[2] https://drububu.com/miscellaneous/voxelizer/?out=obj

1.2.3. Tinklelis (angl. Mesh)

Tai skaitmeninis 3D objekto atvaizdavimas, naudojant tarpusavyje sujungtų viršūnių, briaunų ir veidų rinkinį. Viršūnės apibrėžia erdvėje objekto formą, o briaunos jungia viršūnių poras, sudarydamos linijas išilgai paviršiaus[13]. Tinkleliai yra universali ir plačiai naudojama duomenų struktūra 3D objektams atvaizduoti.



1.4 pav. Tinklelio pavyzdys [3]

Grafų konvoliuciniai tinklai (GCN)[14] leidžia apdoroti duomenis, kurie yra aprašyti grafais, tokius kaip tinkleliai. Kiekviena tinklelio viršūnė yra grafų viršūnė, o briaunos nurodo ryšius tarp jų. Grafų konvoliucijos leidžia skleisti informaciją tarp viršūnių, naudojant jų kaimynystės informaciją. Pixel2Mesh[2] yra vienas iš modelių, kuris naudoja grafų konvoliucijas tinkleliams generuoti iš vieno vaizdo. Modelis pradeda nuo paprasto tinklelio (pvz., sferos) ir iteratyviai deformuoja jį, kad jis atitiktų įvesties vaizde matomą objektą, naudojant grafų konvoliucijas požymių sklidimui per tinklelio paviršių.

Privalumai:

- **Tikslumas:** Gali atvaizduoti sudėtingas paviršiaus geometrijas su dideliu detalumu.
- **Efektyvus atvaizdavimas:** Suderinami su grafikos aparatine įranga.

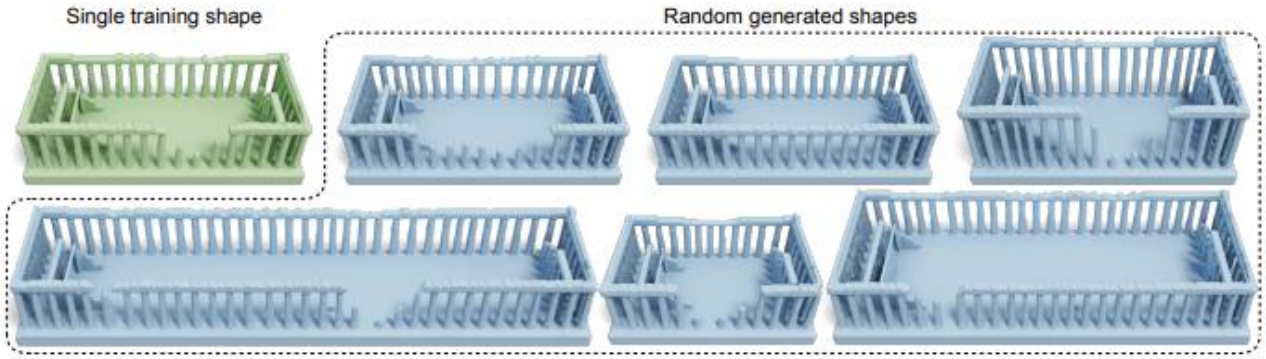
Trūkumai:

- **Sudėtingumas:** Darbas su tinkleliais reikalauja sudėtingų duomenų struktūrų ir algoritmų.
- **Nereguliari struktūra:** Sunku taikyti standartinius CNN[15], nes tinkleliai neturi reguliarios tinklelio struktūros.

[3] https://www.um.es/freefem/ff++/uploads/Main/torus.png

1.2.4. Neuroninis laukas (angl. Neural Field)

Neuroninis laukas - tai skaičiavimo modelis, sukurtas pagal smegenų struktūrą ir funkciją. Neuroninius laukus sudaro daug tarpusavyje sujungtų neuronų, kurių kiekvienas yra atsakingas už tam tikros įvesties duomenų savybės atvaizdavimą. Neuronai neuroniniame lauke gali bendrauti tarpusavyje ir daryti įtaką vienas kito aktyvumui, todėl susidaro sudėtingi aktyvumo modeliai, galintys atspindėti pagrindinę įvesties duomenų struktūrą.



1.5 pav. Neuroninio lauko pavyzdys[16]

1.3. Modelio generavimo tikslumo apskaičiavimo metodai

Vertinant generuojamų 3D modelių kokybę, svarbu naudoti tinkamas metrikas, kurios atspindėtų tiek geometrinį, tiek struktūrinį atitikimą[6] [17]. Šios metrikos leidžia objektyviai palyginti skirtingus modelius ir metodus.

1.3.1. Earth Mover‘s Distance

Earth Mover‘s Distance (EMD) - yra transportavimo uždavinio, kuriuo bandoma vieną aibę transformuoti į kitą, sprendimas[6]. Dviem vienodo dydžio poaibiams S1 ⊆ R3, S2 ⊆ R3, jų EMD apibrėžiamas taip[6]:

$$d_{EMD}(S1, S2) = \min_{\varphi: S1 \to S2} \sum_{x \in S1} ||x - \varphi(x)||_2 \quad (1)$$

kur φ yra bijekcija, x – tai, trijų dimensijų Euklidinėje erdvėje (R3) x = (x1,x2,x3) yra koordinates S1 rinkinyje, tuomet $\varphi(x) = (\varphi_1(x), \varphi_2(x), \varphi_3(x))$ yra atitinkantis taškas aibėje S2, kaip nustatyta pagal funkciją φ: S1 → S2. Tuomet $||x - \varphi(x)||_2$ yra apskaičiuojama taip[18]:

$$||x - \varphi(x)||_2 = \sqrt{(x_1 - \varphi_1(x))^2 + (x_2 - \varphi_2(x))^2 + (x_3 - \varphi_3(x))^2} \quad (2)$$

EMD yra jautri globaliems formos skirtumams ir gerai atspindi bendrą taškų debesų panašumą. Tačiau jos skaičiavimo sudėtingumas yra didelis – O(N3), kur N yra taškų skaičius, todėl praktiniam naudojimui dažnai reikia taikyti aproksimacijas arba sumažinti taškų skaičių.

1.3.2. Chamfer Distance

Chamfer Distance (CD) - dar viena dažnai naudojama 3D objektų generavimo metodų efektyvumo vertinimo metrika. Tai vidutinio atstumo tarp prognozuojamo 3D objekto viršūnių ir artimiausių atitinkamų tikrojo objekto viršūnių matas. Mažesnė CD reikšmė rodo didesnį artumą, o tai reiškia, kad prognozuojamas objektas yra tikslesnis ir artimesnis tikrajam objektui. Norint apskaičiuoti CD, iš pradžių apskaičiuojami atstumai tarp visų numatomų ir tikrųjų objektų atitinkamų viršūnių porų[7]. Tuomet esant P ir Q taškų debesims, formulė atrodytų taip[17]:

$$d_{CD}(P,Q) = \frac{1}{|P|}\sum_{x\in P}\min_{y\in Q}||x-y||_2^2 + \frac{1}{|Q|}\sum_{y\in Q}\min_{x\in P}||x-y||_2^2 \quad (3)$$

CD palyginti su EMD, efektyvesnis apskaičiuoti, kadangi CD skaičiavimo požiūriu paprastesnis, nes CD apskaičuoja vidutinį atstumą tarp kiekvieno taško, o EMD reikalauja daugiau skaičiavimų, nes reikia rasti optimalų transportavimo planą, kad būtų sumažintos vieno paskirstymo transformavimo į kitą sąnaudos[6]. Taip pat, tai yra vienas iš dažniausiai naudojamų metrikų apskaičiuojant objektų generavimo tikslumą.

1.3.3. 3D Voxel-wise Softmax (cross-entropy loss)

Nuostolių funkcija apibrėžiama kaip vokselio kryžminės entropijos suma[1]. Tegul kiekvieno vokselio (i, j, k) galutinė išvestis būna Bernulio skirstiniai [1 - p(i,j,k) , p(i,j,k)], kur priklausomybė nuo įvesties $X = \{X_t\}_{t\in\{1,\dots,T\}}$ praleidžiama, ir tegul atitinkamas pagrindinis tiesos užimtumas yra y(i,j,k) ∈ {0, 1}, tada[1]:

$$L(X,y) = \sum_{i,j,k} y_{(i,j,k)}\log(p_{(i,j,k)}) + (1-y_{(i,j,k)})\log(1-p_{(i,j,k)}) \quad (4)$$

1.3.4. Intersection-over-Union

Sankirta virš sąjungos (IoU) yra dažnai naudojama metrika 3D objektų generavimo metodų našumui vertinti. Tai yra prognozuojamo 3D objekto ir tikrojo objekto sutapimo matas, išreiškiamas procentais. Didesnė IoU reikšmė rodo geresnį sutapimą, o tai reiškia, kad prognozuojamas objektas yra tikslesnis ir artimesnis pagrindiniam objektui.

Norint apskaičiuoti IoU, pirmiausia apskaičiuojama prognozuojamo ir tikrojo objekto sankirta ir sąjunga[19]. Susikirtimas - tai tikrojo objekto dalis, kuri taip pat patenka į prognozuojamą objektą, o sąjunga - tai bendras prognozuojamo ir tikrojo objektų plotas. Tuomet IoU apskaičiuojamas pagal šią formulę[19]:

$$IoU(P, I) = \frac{(P \cap I)}{(P \cup I)} \tag{5}$$

Reikia paminėti, jog šis metodas paprastai nėra taikomas su taškų debesies generavimu, kadangi kuriami taškai neturi tūrio, tačiau yra galimybių tai pritaikyti papildomai apibrėžiant taškus, kad sudaryti minėta tūrį, arba taškus paversti vokseliais. Naudojant vokselius šį metodą yra paprasčiau pritaikyti konkretesne formule[1]:

$$IoU = \frac{\sum_{i,j,k}[I(p_{(i,j,k)} > t)I(y_{(i,j,k)})]}{\sum_{i,j,k}[I(p_{(i,j,k)} > t) + I(y_{(i,j,k)})]} \tag{6}$$

Kur p(i,j,k) nurodo spėjamo vokselio koordinatėse (i,j,k) reikšmę, o y yra originalaus objekto vokselis. Tuomet t prognozuojamos vokselio vertės slenkstis (angl. threshold). Jei tam tikram vokseliui galioja abi sąlygos, indikatoriaus funkcija I(·) yra lygi 1. Jei viena iš sąlygų arba abi sąlygos yra klaidingos, indikatoriaus funkcija lygi 0. Tada IoU formulė susumuoja šias indikatoriaus vertes visiems 3D erdvės vokseliams ir apskaičiuoja IoU metrikos susikirtimo ir susivienijimo reikšmes.

1.3.5. Mean Square Error

Mean Square error (MSE) - yra įprasta metrika, naudojama įvairiose srityse, įskaitant 3D modelių kūrimą, siekiant įvertinti sukurtų modelių tikslumą, palyginti su jų pagrindiniais arba etaloniniais modeliais. MSE kiekybiškai įvertina vidutinius kvadratinius skirtumus tarp atitinkamų duomenų taškų sukurtuose modeliuose, taip nustatant, kaip gerai sukurtas modelis aproksimuoja pagrindinę tiesą.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{7}$$

Generuojant 3D modelius, MSE ypač naudinga vertinant sukurto modelio geometrinį tikslumą. Panagrinėkime vidutinės kvadratinės paklaidos sąvoką 3D modelių kontekste

1.3.6. Hausdorff distance

Hausdorff atstumas - tai didžiausias nuokrypis tarp dviejų modelių, matuojantis, kaip toli viena nuo kitos yra dvi taškų aibės. Esant dviem netuščioms taškų aibėms $A=\{x_1,x_2,\ldots,x_n\}$ ir $B=\{y_1,y_2,\ldots,y_m\}$, Hausdorff atstumas tarp A ir B yra apibrėžtas taip[20]:

$$H(A,B) = \max\left(h,(A,B),h(B,A)\right) \tag{8}$$

Kur,

$$h(A,B) = \max_{x\in B}(\min_{y\in B}||x-y||) \tag{9}$$

$$h(B,A) = \max_{y\in B}(\min_{y\in A}||y-x||) \tag{10}$$

Hausdorff atstumas dažnai naudojamas modelių atpažinimui, formų atitikimui ir klaidų kontrolei.

1.4. Duomenų apdorojimo būdai (preprocessing)

Duomenų rinkinio paruošimas yra būtinas norint jį tinkamai naudoti. Naudojamas duomenų rinkinys „ShapeNetCore“ yra iš anksto paruoštas, tačiau gali būti atvejų, kai reikia atlikti papildomus veiksmus. Standartiškai duomenys normalizuojami, kad būtų suvienodinta modelių skalė[5]Pcentruojami ir rotuojami taip, kad jų priekinė, kairė ir dešinė pusės atitiktų viena kitą. Taip pat yra vertinga pašalinti duomenis, kurie netikslingai ar neišsamiai reprezentuoja kategorijas[10] 3D modelių kodavimas į formatą, kurį gali naudoti gilaus mokymosi algoritmas. Įprasti metodai yra vokselizacija (objektas vaizduojamas vokselių tinkleliu) arba taškų debesys (objektas vaizduojamas taškų rinkiniu). Duomenų rinkinys turi būti padalintas į mokymo, patvirtinimo ir testavimo rinkinius. Mokymo rinkinys naudojamas modeliui apmokyti, tvirtinimo rinkinys naudojamas hiperparametrams derinti, o testavimo rinkinys naudojamas galutiniam modelio veikimui vertinti naudojant nematytus duomenis.

1.5. Modeliai

1.5.1. Generative Adverserial Network (GAN)

Generatyviniai adversariniai tinklai (GAN) yra gilaus mokymosi modelių tipas, kuris išpopuliarėjo dėl gebėjimo generuoti aukštos kokybės sintetinius duomenis. GAN sudaro du neuroniniai tinklai - generatorius ir diskriminatorius, kurie konkuruoja tarpusavyje unikalioje į žaidimą panašioje sistemoje [21,22].

1.5.1.1. Komponentai ir mechanizmas

Generatorius (G):

Generatorius paima atsitiktinį triukšmo vektorių, paimtą iš anksto apibrėžto pasiskirstymo, pavyzdžiui, Gauso pasiskirstymo, ir paverčia jį sintetiniais duomenimis. Pavyzdžiui, generuojant vaizdus, generatorius sukuria vaizdus, kurie turi atrodyti tikri.

Diskriminatorius (D):

Diskriminatorius veikia kaip dvejetainis klasifikatorius, kuris bando atskirti tikrus duomenis (iš mokymo aibės) nuo netikrų duomenų (sukurtų generatoriaus). Jis išveda tikimybę, nurodančią, ar įvesties duomenys yra tikri, ar netikri.

1.5.1.2. Mokymo procesas

GAN mokymas apima konkurencinį procesą, kurio metu abu tinklai laikui bėgant tobulėja per min-max optimizavimo žaidimą:

Diskriminatoriaus tikslas - teisingai suklasifikuoti tikrus ir netikrus duomenis, maksimaliai padidinant savo gebėjimą juos atskirti. Diskriminatoriaus nuostolių funkcija yra tokia[21]:

$$L_G = -E_{x\sim pdata(x)}[\log D(x)] - E_{z\sim pz(z)}[\log(1 - D(G(z)))] \quad (11)$$

Čia x žymi realius duomenis, z - atsitiktinį triukšmą, $p$data - duomenų pasiskirstymą, o pz - latentinio kodo išankstinį pasiskirstymą.

Generatoriaus tikslas: Generatoriaus tikslas - apgauti diskriminatorių, kad šis jo suklastotus duomenis klasifikuotų kaip tikrus. Generatoriaus nuostolių funkcija yra tokia[21]:

$$L_G = -E_{z\sim pz(z)}[\log D(G(z))] \quad (12)$$

Generatorius stengiasi maksimizuoti tikimybę, kad diskriminatorius neteisingai pažymės netikrus duomenis kaip tikrus. Generatorius ir diskriminatorius mokomi pakaitomis. Generatorius tobulina savo sintetinius rezultatus, kad apgautų diskriminatorių, o diskriminatorius tobulina savo gebėjimą aptikti netikrus pavyzdžius.

1.5.1.3. Iššūkiai

Modos griūtis (angl. Mode Collapse):

GAN gali susidurti su modos griūtimi, kai generatorius sukuria ribotus variantus, apimančius tik kelis duomenų pasiskirstymo aspektus. Dėl to generuojamose imtyse trūksta įvairovės.

Mokymo nestabilumas: Priešiškas mokymo procesas gali būti nestabilus, todėl neretai nesugebama pasiekti konvergencijos. Šis nestabilumas atsiranda dėl subtilios pusiausvyros tarp generatoriaus ir diskriminatoriaus mokymo metu.

1.5.2. Variational Autoencoder (VAE)

Variaciniai autoenkoderiai (VAE) - tai generatyvinių modelių klasė, turinti didelę įtaką Neprižiūriamajam mokymuisi ir atstovavimo mokymuisi. Didesnę įtaką suteikė pristatytas VAE, kuriame derinami gilaus mokymosi ir Bajeso išvedimo principai, kad būtų generuojami nauji duomenys, panašūs į įvesties duomenis, pagal kuriuos jie yra apmokyti[22].

1.5.2.1. Sudedamosios dalys ir mechanizmas

Koduotojas (angl. Encoder): Koduotojas yra neuroninis tinklas, kuris įvesties duomenis suspaudžia į mažesnio matmens erdvę. Jis įvesties duomenis x atvaizduoja į latentinį kintamąjį z ir pateikia du rezultatus: latentinės erdvės pasiskirstymo vidurkį (μ) ir dispersiją (σ^2). Tai leidžia modeliui išmokti latentinių kintamųjų pasiskirstymą, o ne fiksuotas reikšmes.

Latentinė erdvė - tai vieta, kurioje yra suspaustas duomenų atvaizdavimas. Užuot kodavus duomenis į vieną latentinės erdvės tašką, VAE koduoja juos į pasiskirstymą, paprastai Gauso. Koduotojas generuoja šio pasiskirstymo parametrus (μ ir σ) kiekvienam įėjimui.

Dekoduotojas (angl. Decoder): Dekoduotojas yra kitas neuroninis tinklas, kuris iš latentinio kintamojo z rekonstruoja įvesties duomenis. Jis atvaizduoja atrinktą latentinį kintamąjį atgal į duomenų erdvę, siekdamas gauti duomenis, kurie būtų kuo artimesni pradinei įvesties informacijai.

1.5.2.2. Mokymo procesas

VAE mokymas apima dviejų tikslų optimizavimą vienu metu:

Rekonstrukcijos nuostoliai: Rekonstrukcijos nuostoliai parodo, kaip gerai dekoderis gali atkurti įvesties duomenis iš latentinių kintamųjų. Tai yra apskaičiuojama naudojant nuostolių funkciją, pavyzdžiui, vidutinę kvadratinę paklaidą (MSE) arba dvejetainę įvesties duomenų ir rekonstruotų duomenų kryžminę entropiją.

KL divergencija (Kullback-Leibler): divergencija užtikrina, kad išmoktas latentinės erdvės pasiskirstymas būtų artimas iš anksto nustatytam išankstiniam pasiskirstymui (paprastai standartiniam Gauso pasiskirstymui). Šis reguliacijos narys neleidžia latentinei erdvei pernelyg prisitaikyti prie mokymo duomenų ir skatina latentinės erdvės sklandumą. Bendra VAE nuostolių funkcija yra tokia[22]:

$$L_{VAE} = E_{q\phi(z|x)}[\log_{p\theta}(x \mid z)] - D_{KL}(q_{\phi}(z \mid x) \,||\, p(z)) \tag{13}$$

kur $q\phi(z|x)$ yra koduotojas, $p\theta(x|z)$ yra dekoderis, o $p(z)$ yra išankstinis latentinių kintamųjų pasiskirstymas.

1.5.2.3. Iššūkiai

Užpakalinis žlugimas (angl. Posterior Collapse): Viena iš dažnų VAE problemų yra užpakalinis žlugimas, kai išmoktoje latentinėje erdvėje neužfiksuojama naudinga informacija, todėl dekoderis ignoruoja latentinius kintamuosius ir generuoja duomenis tik iš triukšmo. Taip gali atsitikti, kai mokymo procese dominuoja KL divergencijos narys.

Neryškūs rezultatai: Palyginti su kitais generatyviniais modeliais, tokiais kaip GAN, yra linkę duoti neryškius rezultatus. Taip yra todėl, kad rekonstrukcijos nuostoliai skatina modelį generuoti galimų išėjimų vidurkius, todėl vaizdai gali būti ne tokie ryškūs.

1.6. Difuzijos modelis

Difuzijos modeliai - tai generatyvinių modelių klasė, į kurią pastaruoju metu atkreiptas dėmesys dėl jų gebėjimo gauti aukštos kokybės duomenis taikant tikimybinį procesą. Šie modeliai grindžiami idėja, kad į duomenis iteratyviai pridedamas triukšmas, o paskui išmokstama šį procesą pakeisti, kad būtų generuojami nauji pavyzdžiai. Šie Difuziniai modeliai, parodė perspektyvumą įvairiose srityse, įskaitant vaizdų sintezę, teksto generavimą ir kt.[23].

1.6.1. Sudedamosios dalys ir mechanizmas

Pirmyn nukreipas difuzijos procesas (angl. Forward Diffusion Process): Vykstant tiesioginiam procesui, per keletą laiko žingsnių į duomenis palaipsniui įterpiamas triukšmas. Pradedant nuo pradinių duomenų $x0$, kiekviename vėlesniame žingsnyje pridedamas nedidelis Gauso triukšmo kiekis, taip sukuriant vis labiau triukšmingų duomenų seką $x1$,$x2$,...,xT, kur T yra bendras žingsnių skaičius. Šis procesas modeliuojamas kaip Markovo grandinė, o kiekvienas žingsnis apibrėžiamas dispersiniu grafiku βt. Šį procesą galima apibūdinti taip[23]:

$$q(x_t \mid x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{14}$$

1.7. Konvoliuciniai neuroniniai tinklai (CNN)

CNN yra plačiai naudojami vaizdų apdorojime ir gali būti pritaikyti 3D duomenims naudojant 3D konvoliucijas arba projekcijas[24]. Pavyzdžiui, 3D-R2N2[1] naudoja CNN kartu su RNN[25] 3D objektų rekonstrukcijai iš vieno ar kelių vaizdų

1.8. Tyrimai Lietuvoje ir pasaulyje

Šiame poskyryje apžvelgiama, kaip pasaulyje ir Lietuvoje vyksta vieno vaizdo 3D rekonstrukcijos tyrimai. 1.2 Lentelėje pateikiami keli darbai, kurių autoriai mėgina atkurti trimatę objekto formą iš ribotos 2D informacijos. Metodai varijuoja nuo pirminių, tokių kaip 3D-R2N2, kur vaizdiniai požymiai gaunami CNN sluoksniais ir apjungiami pasikartojančio neuroninio tinklo (RNN) būdu, iki pažangesnių metodikų, dirbančių su taškų debesimis ar vokseliais ar netgi giliai užkoduotomis implicitinėmis funkcijomis.

Skirtingi autoriai taiko įvairias kokybės vertinimo metrikas. Kai kurie, pvz., 3D-R2N2 ar Kulikajevas 2020, naudoja Intersection-over-Union (IoU), kuri ypač tinka voxel pagrindu rekonstruojamiems 3D modeliams. Kiti pasitelkia Chamfer Distance (CD) ar Earth Mover's Distance (EMD), aktualius taškų debesų ar paviršių atkūrimui, nes šios metrikos labiau pabrėžia paviršiaus atitikimą. Dar kiti, kaip Pixel2Mesh ar implicitinio paviršiaus tinklų kūrėjai, pateikia papildomus rodiklius, pavyzdžiui, F-score, rodantį teisingai atkurtų paviršiaus briaunų procentą.

1.2 lentelė atskleidžia, kad pirminiai vieno vaizdo rekonstrukcijos metodai prasidėjo nuo paprastos voxel išvesties, o tolesni darbai ėmė gilintis į detalesnį objekto atvaizdavimą per tinklelio struktūrą, taškų debesį ar net implicitines funkcijas. Matyti ir eksperimentų su papildomomis įvestimis, pavyzdžiui, gylio žemėlapiais ar normalių duomenimis, kurie padeda labiau „užpildyti" trimačio objekto paviršių trūkstamomis detalėmis. Taip pat svarbus faktorius – visos šios architektūros paprastai siekia geresnės geometrijos atkūrimo kokybės naudojant vieną vaizdą, tad sprendžiama pamatinė problema: iš vienos projekcijos suprasti visą objekto tūrį. Iš šių darbų labiausiai išsiskiria Genova darbas, kuris yra paminėtas numeriu 7. Čia buvo pasiektas F-score: 78.1%, kas yra įspūdinga, palyginus su kitais darbais. Šis projektas parodo, kad galbūt nereikia apsiriboti klasikinėmis objekto reprezentacijomis ir sudarė kelias dešimtis papildomų šablonų, kurie padeda modeliui nuspręsti į kokias zonas reikia atkreipti daugiau dėmesio, kai objektas yra rekonstruojamas.

1.**2 lentelė.** Tyrimai naudojantys „ShapeNetCore“ duomenų rinkinį

Nr	Autorius, Metai	Naudojami modeliai	Įvestis (iš vienos kameros)	Tikslumas	Pastabos
	Choy Christopher B. 2016[1]	CNN + RNN (3D-R2N2)	Vienas RGB vaizdas (taip pat gali kelis, bet veikia ir su 1)	IoU: 0.571	Vienas iš pirmutinių darbų, atkuriantis 3D vokselius iš vieno RGB vaizdo ar daugiau.
1	Wu Jiajun 2017[26]	CNN su tarpinėmis 2.5D reprezentacijomis (gylis, normalės)	Vienas RGB vaizdas	IoU: 0.57	Naudoja "Marr"[27] principus: iš vieno RGB vaizdo išvedamos tarpinės 2.5D projekcijos, paskui generuojamas 3D vokselių modelis.
2	Fan Haoqiang 2017[17]	Point Set Generation Network	Vienas RGB vaizdas	IoU: 0.64	Generuoja tiesioginį taškų debesį iš vieno RGB vaizdo.
3	Wang Nanyang 2018[2]	CNN → Graph CNN (Pixel2Mesh)	Vienas RGB vaizdas	F-score:59.72% CD:0.591 EMD:1.380	Atkuria tinklelius iš vieno RGB vaizdo
4	Groueix, Thibault 2018[28]	CNN + atlas parametrizacija	Vienas RGB vaizdas	CD:0.00511	Naudoja atlasų rinkinį tinklelio paviršiaus atkurimui iš vieno vaizdo.
5	Xu Qiangeng 2019[29]	CNN + Implicit Surface Net	Vienas RGB vaizdas	IoU:57.0% CD:0.01098 EDM:0.0284	DISN prognozuoja tiesioginę paviršiaus funkciją iš vieno vaizdo
6	Chen Zhiqin 2019[30]	VAE	Vienas RGB vaizdas	-	IM-Net prognozuoja, ar taškas priklauso objektui, iš vieno vaizdo atkuria išsamų 3D modelį.
7	Genova, Kyle 2019[31]	CNN + Occupancy Net	Vienas gylio vaizdas	IoU: 78.1% F-score: 78.1% Chamfer:0.76	Naudoja vaizdinę informaciją iš vienos kameros kampo. Naudojamas rankiniu būdų sudarytas šablonas kiekvienam objektui.
8	Audrius Kulikajevas 2020[10]	YOLOv3, Autoencoders	Generuojamas gilumo žemėlapis ir RGB vaizdas	IoU: Depth:26.46% RGB:41.27% RGB-D:60.20%	Objekto rekonstrukcija naudojant gilumo duomenis. Gilumo duomenys gali būti nestabilūs

Taipogi 1.3 lentelėje yra ir darbų, kurie naudoja skirtingas metodologijas, duomenų rinkinius bei vertinimo metrikas, todėl jie šiek tiek mažiau susiję su šio projekto tikslu, kadangi yra sunku lyginti. Šie tyrimai atspindi platesnį požiūrį į vieno vaizdo 3D rekonstrukciją, įtraukiant įvairias technologijas ir taikymo sritis. Pavyzdžiui, Nr 1 naudoja GAN metodą vokselizuotai objekto formai generuoti, pasiekdami IoU 0.477 ir F-score 0.594. Kita vertus, Nr 3 taiko Generative 3D Gaussian Splatting bei SDS (score distillation sampling) metodus, siekdami greitai generuoti tekstūruotas 3D formas su CLIP-similarity 0.738. Šie darbai demonstruoja, kaip įvairios architektūros ir technikos gali būti pritaikytos skirtingoms 3D rekonstrukcijos užduotims, nors jų rezultatai gali skirtis priklausomai nuo naudojamų metodų ir vertinimo kriterijų.

1.3 lentelė. Kitų metodikų tyrimai

Nr	Autorius, Metai	Naudojami modeliai	Įvestis	Tikslumas	Pastabos
1	Rundi Wu 2022[16]	GAN	Vokselizuota objekto forma	IoU: 0.477 F-score: 0.594 SSFID[4]: 0.074	3D formos generavimas. Ne efektyvi vokselių reprezentacija, nesubalansuotas išvesties kitimas
2	Angjoo Kanazawa 2018[32]	Deformuojamas 3D tinklas (vidutinis formos modelis + CNN prognozė)	Vienas RGB vaizdas	IoU: 0.46(lėktuvai), 0.64(automobiliai);	Modelis atkuria 3D formą, tekstūrą ir kamerą. Mokoma naudojant vieno vaizdo per instanciją priežiūrą. Eksperimentai su paukščiais (CUB) ir PASCAL objektais (lėktuvai, automobiliai).
3	Jiaxiang Tang 2023[33]	Generative 3D Gaussian Splatting, SDS (score distillation sampling), MSE	Vienas RGB vaizdas, tekstinė užklausa	CLIP-similarity: 0.738	Modelis naudoja 3D Gaussian Splatting techniką, siekdamas generuoti tekstūruotas 3D tinklų (mesh) formas per kelias minutes. Efektyviai optimizuojama per 2 etapų procesą.
4	Ruoshi Liu 2023[34]	Stable Diffusion, Score Jacobian Chaining (SJC), CLIP	Vienas RGB vaizdas	PSNR: 18.378; IoU: 0.5052	Zero-1-to-3 leidžia generuoti naujas objekto perspektyvas ir atkurti 3D formas. Naudojamas didelio masto išankstinis mokymas bei difuzijos modelių optimizacija.

1.2 ir 1.3 lentelėse atskleidžiama pagrindinė vieno vaizdo 3D rekonstrukcijos tyrimų dinamika Lietuvoje ir pasaulyje. Panašūs tyrimai, naudojantys ShapeNetCore duomenų rinkinį, demonstruoja pagrindines architektūras ir metodikas, siekiant tiksliai atkurti 3D objektus iš vienos 2D projekcijos. Kita vertus, įvairūs kitų metodikų tyrimai parodo, kad šis laukas yra plačiai tyrinėjamas su skirtingomis technologijomis ir taikymo kryptimis. Tai rodo, kad nors pagrindiniai iššūkiai – kaip smulkių detalių atkūrimas ir tikslumo pagerinimas – yra bendri, tyrėjai naudoja įvairias strategijas jų sprendimui. Tokiu būdu šis tyrimas remiasi esamais pasiekimais ir stengiasi integruoti pažangias metodikas, siekiant pagerinti 3D rekonstrukcijos tikslumą bei efektyvumą.

1.9. Naudojami Įrankiai ir metodai

Šiame skyriuje aptariama, kokias priemonės ir technologijos bus naudojamos, siekiant pagrindinio tikslo: iš vieno vaizdo atkurti trimačio objekto formą naudojant gilųjį mokymąsi. Visa duomenų kelionė – nuo papildomos informacijos generavimo iki galutinio modelio vertinimo – remsis keliais svarbiais komponentais: „Blender“[35] duomenų kūrimui, „PyTorch“[36] modelių mokymui, vokseline reprezentacija trimatiems duomenims ir „Intersection-over-Union“ (IoU) kaip pagrindine vertinimo metrika.

Idėja yra tokia: pirmiausia reikia tinkamų duomenų, kad modelis turėtų iš ko mokytis. Vėliau, su turimais duomenimis, apmokome neuroninį tinklą, kad iš vieno 2D vaizdo jis išmoktų atkurti

[4] Single Shape Fréchet Inception Distance

objekto trimatę formą. Galiausiai, gautą rezultatą patikriname ir įvertiname, ar mūsų modelis iš tiesų atkuria daiktą taip, kaip norėtume.

Visas procesas atrodytų taip:

1. Duomenų generavimas su „Blender“: paruošiami papildomi sintetiniai vaizdai.
2. Modelio mokymas su „PyTorch“: naudojamai GPU resursai ir lanksčią sistemą, kad neuroninis tinklas galėtų „pažinti“ 3D formą iš dvimačių vaizdų.
3. Reprezentacija vokseliais: trimačiai duomenys bus atvaizduojami vokselių forma, kad būtų lengviau modelį apmokyti – juk CNN tinklams taip paprasčiau skaičiuoti 3D erdvėje.
4. Vertinimas su IoU: gautą 3D rezultatą palyginame su tiesa (angl. ground truth) per IoU metriką, kuri parodys, kiek modelis tiksliai atkuria objekto tūrį.

1.9.1. „Blender“ generuoti duomenims

Blender yra atvirojo kodo 3D kūrimo paketas, apimantis kaulų kūrimą, animaciją, modeliavimą, atvaizdavimą, komponavimą ir judesio sekimą. Jis suteikia galingas scenarijų kūrimo galimybes naudojant Python, todėl yra tinkamas įrankis sintetiniams 3D duomenims generuoti.

Privalumai:

- Universalumas: „Blender“ siūlo platų 3D modelių kūrimo ir tvarkymo įrankių spektrą.
- Skriptavimo galimybės: Jo Python API leidžia automatizuoti modelio generavimą, įskaitant atsitiktines transformacijas, tekstūrų taikymą ir apšvietimo koregavimą.
- Bendruomenės palaikymas: „Blender“ turi didelę naudotojų bazę, išsamią dokumentaciją ir vadovėlius.

Trūkumai:

- Našumas: Labai didelės raiškos arba sudėtingoms scenoms kurti „Blender“ gali prireikti didelių skaičiavimo išteklių.
- Mokymosi kreivė: Norint išnaudoti pilną Blender potencialą, reikia laiko perprasti jo įrankius bei skriptavimo galimybes.

1.9.2. PyTorch modelio kūrimui

PyTorch yra atvirojo kodo gilaus mokymosi sistema, sukurta „Facebook“ dirbtinio intelekto tyrimų laboratorijoje. Ji žinoma dėl savo dinaminio skaičiavimo grafiko, kuris palengvina eksperimentavimą ir mokslinius tyrimus, ir sklandžios integracijos su GPU spartinimu naudojant CUDA.

Privalumai:

- Naudojimo paprastumas: „PyTorch“ siūlo intuityvų ir paprastą API, todėl idealiai tinka kurti ir testuoti neuroninius tinklus.
- GPU pagreitinimas: „PyTorch“ efektyviai išnaudoja GPU išteklius, todėl gerokai pagreitina modelių mokymą ir išvadų darymą.
- Bendruomenė ir ištekliai: Didelė bendruomenė ir išsami dokumentacija teikia pakankamą paramą kūrėjams ir tyrėjams.

Trūkumai:

- Parengtumas gamybai: Manoma, kad „PyTorch“, palyginti su „TensorFlow“, yra mažiau tinkama gamybai, nors naujausi pasiekimai, pavyzdžiui, „TorchServe“, tai sušvelnino.

PyTorch bus naudojamas gilaus mokymosi modeliams, skirtiems generuoti 3D objektus iš vieno vaizdo vaizdų, įgyvendinti ir mokyti. Jo GPU palaikymas užtikrina, kad mokymo procesas būtų efektyvus ir keičiamo dydžio.

1.9.3. Duomenų atvaizdavimas naudojant vokselius

Šiame projekte bus naudojama vokselinė reprezentacija. Vokseliai yra 3D analogija 2D pikseliams: tai vienetinių tūrių (kubiukų) matrica, nurodanti, ar tam tikra erdvės vieta priklauso objektui.

Privalumai:

- **Reguliari struktūra:** Lengva taikyti 3D konvoliucijas, kurios yra intuityvus sprendimas CNN tipo architektūroms.
- **Pilnas tūris:** Vokseliai atvaizduoja ne tik paviršių, bet ir visą objekto tūrį, kas gali būti naudinga tam tikriems uždaviniams.

Trūkumai:

- **Didelis atminties poreikis:** Padidinus rezoliuciją (pvz., nuo 32^3 iki 128^3 vokselių), atminties sąnaudos auga kubiškai.
- **Sudėtingi objektai:** Didelės raiškos vokseliai būtini smulkioms detalėms atkurti, o tai reikalauja daugiau skaičiavimo išteklių.

Nepaisant to, vokseliai yra patogi reprezentacija, deranti su CNN pagrįstomis architektūromis (kaip Pix2Vox[37]), nes suteikia aiškią 3D struktūrą, leidžiančią modeliams suprasti objekto formą erdvėje.

1.9.4. Sankirtos virš sąjungos (IoU) kaip vertinimo metrika

Ši metrika yra naturaliai pritaikyta skaičiuojant tikslumą dirbant su 3d objektais, kurie yra reprezentuojami vokseliais. Kadangi tai yra dažnai naudojama metrika, iš point cloud bus sudaroma vokselinė reprezentacija ir naudojantis šia metrika palyginama su realaus objekto vokseliais

Privalumai:

- Intuityvumas: IoU yra lengvai suprantamas ir interpretuojamas, todėl yra paprastas modelio našumo matas.
- Standartinė metrika: ji plačiai naudojama šioje srityje, todėl yra patikimas etalonas skirtingiems modeliams ir metodams palyginti.
- IoU natūraliai pritaikytas vokselių vaizdavimui, todėl turėtų būti mažiau sunkumų jį pritaikyti.

Trūkumai:

- Nepaiso smulkesnių detalių: IoU nesuteikia tiesioginio informacijos apie paviršiaus lygumą, briaunų aštrumą ar smulkius geometrijos niuansus. Tačiau bendram tikslumui vertinti ji puikiai tinka.

1.10. Apibendrinimas

Šioje analizėje buvo aptartas vieno vaizdo pagrindu vykdomas 3D objektų generavimas, apžvelgiant naudojamus duomenų rinkinius, modelių architektūras, vertinimo metrikas bei kylančius techninius ir praktinius iššūkius. Pagrindinis šios srities tikslas – transformuoti ribotą 2D informaciją į tikslius 3D modelius, išlaikant originalią objekto geometriją ir struktūros detalumą. Nors giliojo mokymosi metodai jau pasiekė reikšmingų laimėjimų, ši sritis vis dar susiduria su iššūkiais, kurie reikalauja tolesnio tyrimų bei inovacijų.

- **Duomenų svarba ir paruošimas:** „ShapeNetCore“ duomenų rinkinys, aptartas šioje analizėje, yra plačiausiai naudojamas šioje tyrimų srityje dėl savo įvairovės ir universalumo. Šio rinkinio turimi 3D modeliai leidžia modelius mokyti taip, kad jie galėtų generalizuoti įvairias objektų formas bei struktūras. Vis dėlto, tinkamas duomenų paruošimas, įskaitant normalizaciją, centravimą ir nereikalingų pavyzdžių pašalinimą, išlieka esminis veiksnys siekiant aukšto modelių tikslumo. Nepaisant šio rinkinio standartizavimo, vis dar egzistuoja iššūkių, susijusių su trūkumais bei duomenų apdorojimo resursais.
- **Modelių architektūros ir reprezentacijos:** Aptarti modeliai atskleidė skirtingas 3D objektų reprezentacijas, tokias kaip taškų debesys, vokseliai, tinkleliai ir neuroniniai laukai. Taškų debesys pasižymi lankstumu ir gebėjimu tiksliai atkurti sudėtingas formas, tačiau jų netvarkinga struktūra apsunkina apdorojimą. Vokseliai, savo ruožtu, leidžia paprastai reprezentuoti tūrinę informaciją, tačiau jų naudojimas yra ribojamas dėl didelių atminties sąnaudų. Tinkleliai suteikia itin tikslų paviršiaus atvaizdavimą, tačiau dėl savo struktūros apdorojimas reikalauja sudėtingesnių algoritmų. Galiausiai, neuroniniai laukai atveria naujų galimybių, leidžiančių generuoti aukštos kokybės paviršių netgi iš ribotos įvesties, tačiau jų praktinė integracija vis dar reikalauja optimizacijos.
- **Vertinimo metrikos ir iššūkiai:** Analizėje išryškinta vertinimo metrikų įvairovė parodė, kad kiekviena jų akcentuoja skirtingus 3D modelių aspektus. Chamfer Distance ir Earth Mover’s Distance yra populiarios taškų debesų atitikimui vertinti, o Intersection-over-Union (IoU) ypač naudinga vokselių pagrindu veikiančioms architektūroms. Mean Square Error bei kitos metrikos leidžia analizuoti geometrinį tikslumą ir struktūrinius atitikimus. Tinkamas šių metrikų derinimas yra būtinas norint objektyviai įvertinti skirtingų modelių kokybę. Nepaisant pažangos, iššūkiai, tokie kaip smulkių detalių atstatymas ar efektyvus didelės apimties duomenų apdorojimas, išlieka aktualūs.

2. 3D objekto formos rekonstrukcijos metodai ir reikalavimai

Šiame skyriuje pateikiami funkciniai ir nefunkciniai reikalavimai, skirti pagerinti 3D objektų generavimo procesą, paremtą giliuoju mokymusi ir Pix2Vox[37] tipo metodikomis. Pagrindinis vertinimo matas – Intersection-over-Union (IoU), pabrėžiantis atkuriamo tūrio atitiktį tikrajam objektui. Reikalavimai sutelkti į stabilumą, efektyvumą, interpretuojamumą ir galimybes integruoti pažangesnes strategijas.

2.1. Funkciniai reikalavimai

1. **Vieningas duomenų mastelis ir vienodi formatai:** Įvesties vaizdai ir jų atitinkami 3D duomenys turi būti apdorojami taip, kad kiekvienas objektas turėtų standartizuotą mastelį ir orientaciją. Tokiu būdu modelis gauna nuoseklias sąlygas, neatsižvelgiant į objekto pradinę padėtį ar dydį.
2. **Vokselinė reprezentacija ir 3D konvoliucijos:** Sugeneruojami objektai turi būti atvaizduojami vokselių tinkleliu, suderinamu su 3D konvoliucijų sluoksniais. Tai padeda algoritmui atpažinti tūrinę formą ir semantinius požymius, reikalingus tiksliam objekto rekonstravimui iš vieno vaizdo.
3. **Pagrindinė vertinimo metrika – IoU:** Vertinant generuojamo 3D modelio tikslumą, būtina naudoti Intersection-over-Union metriką kaip pagrindinį rodiklį. Ši metrika leidžia aiškiai įvertinti, kokia dalis generuoto tūrio sutampa su tikruoju objekto tūriu, pasiūlydama intuityvų ir pramonėje pripažintą standartą.
4. **Rezultatų registravimas ir analizės priemonės:** Visos treniravimo ir vertinimo metu gautos IoU reikšmės, modelio svoriai, hiperparametrų konfigūracijos bei generuoti 3D modeliai turi būti išsaugomi. Ši informacija svarbi rezultatų palyginimui, modelio tobulinimui bei kokybės užtikrinimui ateityje.

2.2. Nefunkciniai reikalavimai

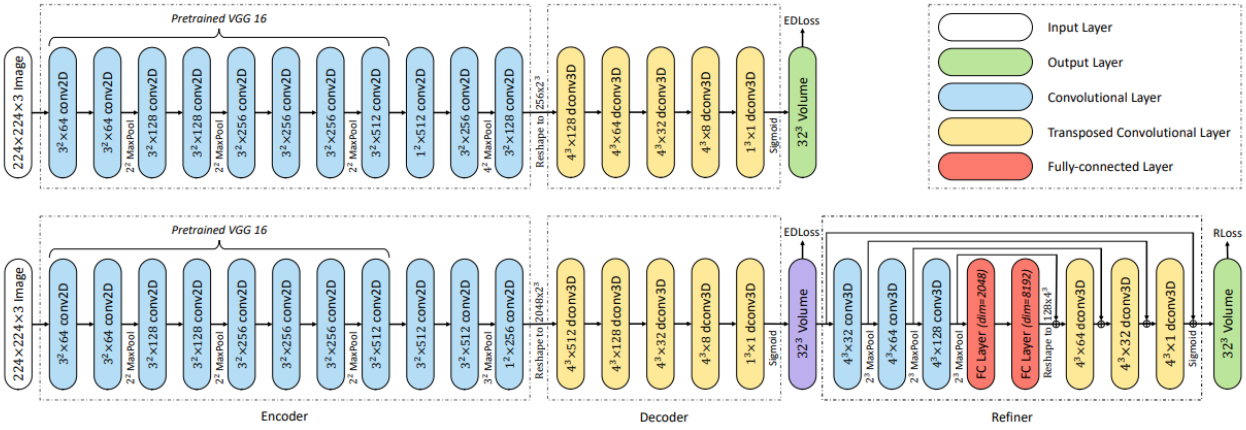
1. **Tikslumo slenkstis (≥ 67% IoU):** Sukurto 3D modelio IoU turi pasiekti bent 67%. Šis slenkstis nurodo priimtiną minimalią kokybę, užtikrinančią, kad generuojamas objektas bus tinkamai atkurtas tūrio lygmenyje.
2. **Išteklių efektyvumas:** Mokymo ir išvadų darymo procesams naudotina GPU akceleracija bei efektyvus duomenų srautų valdymas, siekiant sumažinti skaičiavimo laiko ir energijos sąnaudas. Toks išteklių naudojimo optimizavimas gerina bendrą sistemos tvarumą ir ekonomiškumą.
3. **Modelio stabilumas ir patikimumas:** Modelio treniravimas turi būti stabilus, išvengiant ekstremalių rezultatų svyravimų tarp epochų. Stabilumas didina pasitikėjimą sistemos sprendiniais, leidžia greičiau pasiekti konvergenciją ir sumažina per didelės pertreniravimo riziką.

2.3. Naudojamas modelis

Naudojamas modelis remiasi Pix2Vox[37] architektūra, efektyviai ir lanksčiai rekonstruoja 3D objektus iš dvimačių vaizdų. Svarbiausia savybė – gebėjimas integruoti informaciją iš vieno ar kelių vaizdų, pasitelkiant kontekstinį suvokimą ir adaptatyvius susiliejimo mechanizmus. Taip

pasiekiamas didesnis tikslumas bei kokybė, ypač kai naudojamas daugiau nei vienas įvesties vaizdas.

Pix2Vox architektūra išskaidoma į keturis pagrindinius modulius: koduotoją (encoder), dekoduotoją (decoder), kontekstinio susiliejimo (context-aware fusion) modulį ir rafinerį (refiner). Kiekvienas modulis atlieka savitą funkciją, užtikrindamas sklandų perėjimą nuo dvimačio vaizdo požymių prie trimatės vokselinės reprezentacijos.



2.1 pav. Modelio architektūros atvaizdavimas[37]

Reikia pabrėžti, jog 2.1 paveikslėlyje yra atvaizduojami du modeliai: Pix2Vox-F ir Pix2Vox-A. Pix2Vox-F yra atvaizduojamas kaip viršutinė paveikslėlio dalis. Šis modelis yra smulkesnis nei jo tariamas brolis ir turi privalumų ir trūkumų. Vienas iš gerųjų aspektų yra, tai kad jis yra gerokai mažesnis, ir reikia suteikti mažesnį kiekį laiko apmokymui. Tačiau dėl to nukenčia ir modelio tikslumas 3-4%.

Tuomet paveikslėlio apatinėje dalyje yra atvaizduojamas Pix2Vox-A, kur atsiranda „Refiner“ dalis. Kaip ir pavadinimas sako, ši modelio dalis bando tikslinti rezultatus. Nors kaip ir anksčiau minėta, kad modelis išgauna papildomus kelis procentus tikslumo, tai drastiškai padidina apmokymo laiką šiek tiek daugiau nei du kartus.

Toliau yra aprašoma modelio architektūra detaliau:

Koduotojas (Encoder): Koduotojas naudoja iš anksto apmokytą VGG16 tinklą, kuris sugeba išgauti semantiškai turtingus požymius iš 224x224 dydžio RGB vaizdų. Iš pradžių generuojami 512 matmens požymių žemėlapiai, kurie vėliau apdorojami papildomais konvoliuciniais sluoksniais. Šis etapas užtikrina, kad vaizdinė informacija būtų efektyviai suspausta ir paruošta trimačiam atkūrimui. Požymių išskyrimo procesas leidžia modelio vidinėms reprezentacijoms įsisavinti objekto formos, tekstūros ir apšvietimo sąlygų ypatumus.

Dekoduotojas (Decoder): Dekoduotojas atlieka atvirkštinį procesą – konvertuoja suspaustą dvimatę požymių reprezentaciją į trimatį vokselių tinklelį, apibrėžiantį objekto tūrį. Tam pasitelkiamos kelios trimačios dekonvoliucijos, laipsniškai didinančios erdvinę raišką. Rezultatas – grubus 3D objekto modelis, kuriame matyti bazinė forma ir apytikrės struktūros. Šis žingsnis leidžia pereiti nuo abstrakčių vaizdinių požymių prie aiškios, vokseliais apibrėžiamos erdvinės formos.

Kontekstinis susiliejimas (Context-aware Fusion): Kai naudojami keli vaizdai (pvz., objekto nuotraukos iš skirtingų kampų), kontekstinio susiliejimo modulis adaptatyviai sujungia keliuose vaizduose sukurtus 3D tūrius į bendrą reprezentaciją. Taikomas mechanizmas įvertina kiekvieno grubaus tūrio svarbą, parenka vertingiausią informaciją ir sukuria sujungtą, kokybiškesnį 3D modelį. Tokia strategija padeda išnaudoti papildomas perspektyvas, gautas iš skirtingų vaizdų, ir padidinti atkūrimo patikimumą bei tikslumą.

Rafineris (Refiner): Sujungtam 3D modeliui rafineris suteikia galutinį blizgesį. Šis modulis, pasitelkdamas U-formos tinklą su skip-jungtimis, tobulina detales, taiso nedidelius netikslumus ir pašalina triukšmą, atsiradusį ankstesniuose žingsniuose. Rafineris išlaiko reikšmingą vietinę struktūrą ir padeda atkurti smulkesnes geometrijos ypatybes, todėl gautas rezultatas yra vizualiai ir struktūriškai išbaigtesnis.

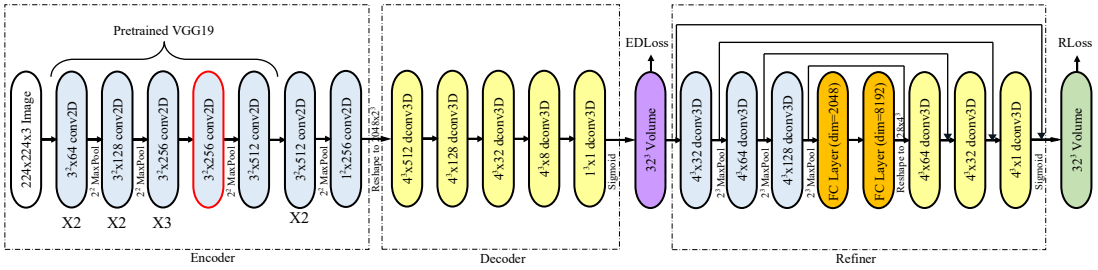
Modelio privalumai:

- **Darbas su vienu ir keliais vaizdais:** Pix2Vox suprojektuotas taip, kad galėtų atkurti objekto 3D formą net ir iš vieno vaizdo, tačiau rezultatai gerėja, jei pateikiama daugiau skirtingais kampais darytų atvaizdų.
- **Kontekstinis suvokimas:** Adaptyvus susiliejimo modulis padeda integruoti informaciją iš įvairių vaizdo šaltinių, išrenkant tinkamiausius požymius galutiniam modeliui. Tai sustiprina galutinės formos tikslumą ir detalumą.
- **Efektyvus skaičiavimo resursų panaudojimas:** Architektūra orientuota į paralelinį atvaizdų apdorojimą. Dėl to modelio mokymo ir išvadų darymo procesai vyksta sparčiau, palyginti su kitais metodais, kurie naudoja sudėtingesnes laikinio apdorojimo (RNN) schemas.

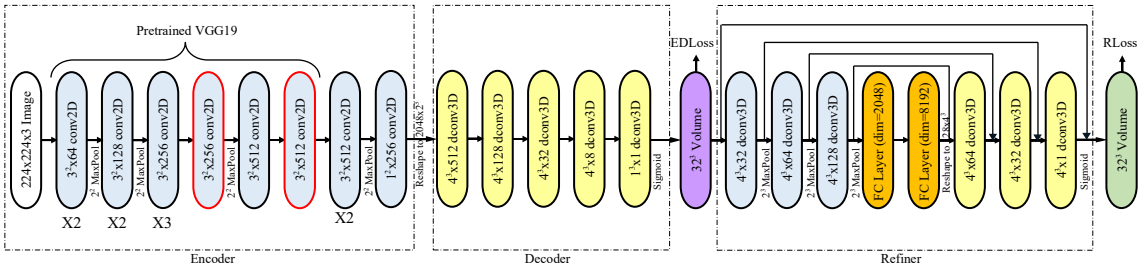
Bendras Pix2Vox architektūros sprendimas pabrėžia aiškiai atskirtus, tačiau tarpusavyje glaudžiai sąveikaujančius modulius, suteikiančius sistemai galimybę efektyviai iššifruoti vizualią informaciją, atkurti objekto 3D struktūrą ir užtikrinti aukštos kokybės rezultatą.

2.4. Architectūrinės modifikacijos

Atliktos kelios architektūrinės modifikacijos, siekiant išsiaiškinti, ar skirtingo gylio ir sandaros iš anksto apmokyti konvoliuciniai tinklai gali pagerinti Pix2Vox sistemos rekonstrukcijos tikslumą, todėl buvo atlikti trys atskiri eksperimentai: du naudojant modifikuotą VGG 19 ir vienas – ResNet 18. Visuose bandymuose dekoderio, „merger“ bei „refiner“ dalys išliko nepakitusios; pakeitimai palietė tik koduotojo bloką, nes būtent jis formuoja svarbiausius vaizdo bruožus, kuriuos vėliau naudoja trimačio tūrio atkūrimo grandinės.

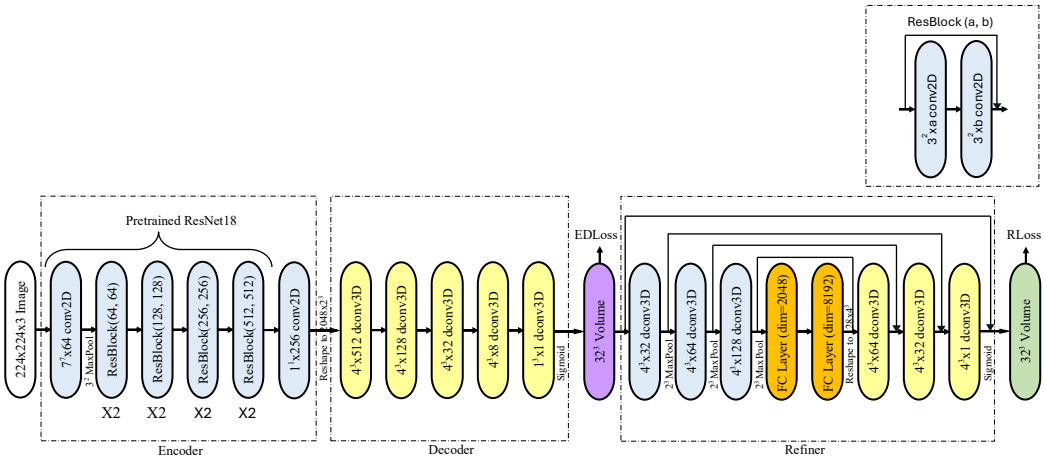


2.2 pav. Pix2Vox-VGG19-1



2.3 pav. Pix2Vox-VGG19-2

Pirmosiose dviejose architektūrose, pavaizduotuose 2.2 pav ir 2.3 pav, remiantis klasikinės VGG 19 architektūros logika, prie bazinio VGG 16 karkaso buvo pridėti papildomi konvoliuciniai sluoksniai, atitinkantys 32 × 256 ir 32 × 512 filtrus. Pix2Vox-VGG19-1 architektūroje įterptas vienas naujas 32 × 256 sluoksnis, esantis prieš maxpool opearaciją. Pix2Vox-VGG19-2 architektūroje, kartu prie 3^2 x 256 sluoksnio, yra pridėtas paplidomas vienas sluoksnis 3^2 x 512 po maxpool operacijos. Schemosee šios naujos konvoliucijos pažymėtos raudonu rėmeliu, kad vizualiai būtų akcentuota jų vieta lyginant su pradiniu sprendiniu.



2.4 pav. Pix2Vox-ResNet18

Trečioje architektūroje, pavaizduota 2.4 paveikslėlyje, VGG pagrindą visiškai pakeitė ResNet 18. Koduotojo brėžinyje nenurodytos visos atskiros konvoliucijos, nes pati ResNet koncepcija grindžiama likučių blokais (angl. residual blocks), kuriuose du 3 × 3 filtrai sujungti trumpomis jungtimis. Šiuo atveju buvo pasitelktas pilnai ImageNet duomenų rinkinyje apmokytas tinklas, pašalinant tik globalų vidurkinimo sluoksnį ir galutinį pilnai sujungtą klasifikatorių. Rezultatas – 512 kanalų 7 × 7 bruožų žemėlapiai, kurie, 1 × 1 konvoliucijos pagalba, sumažinami iki 256 kanalų ir dvišale interpoliacija išplečiami iki 8 × 8, kad atitiktų dekoderio reikalaujamą įvestį.

Taigi visos trys modifikacijos – vieno sluoksnio VGG 19, dviejų sluoksnių VGG 19 ir ResNet 18 koduotojai – sudaro palankias sąlygas palyginti, kaip modelio gylis bei likučių struktūros veikia trimačio tūrio atkūrimo kokybę. Schemose raudonai pažymėtos VGG dalys aiškiai išskiria pridėtus sluoksnius, o ResNet brėžinys akcentuoja visišką koduotojo keitimą. Kadangi dekoderio ir refinerio komponentai liko identiški, bet koks našumo pokytis tiesiogiai siejamas su koduotojo pakeitimais ir leidžia objektyviai įvertinti, ar gilesni, modernesni pagrindai pateisina didesnius skaičiavimo kaštus lyginant su paprastesniu VGG 16 sprendimu.

2.5. Nuostolių funkcijos

Siekiant kiekybiškai įvertinti rekonstrukcijos kokybę ir užtikrinti, kad tinklas mokymosi metu sistemingai artėtų prie tikslo objekto tūrio, visame Pix2Vox architektūros mokyme taikomas vokselinis binarinės kryžminės entropijos nuostolis. Kiekvienam vokseliui i skaičiuojama tikimybė pi, jog ta gardelė yra užpildyta, o gti žymi atitinkamą pagrindinės tiesos užpildymo reikšmę (0 arba 1). Bendra rekonstrukcijos paklaida apibrėžiama kaip vidutinė visų N vokselių entropija[37]:

$$L = \frac{1}{N}\sum_{i=1}^{N}[gt_i \log(p_i) + (1 - gt_i)\log(1 - p_i)] \tag{15}$$

kur mažesnė L reikšmė rodo didesnį tikrosios ir rekonstruotos geometrijos sutapimą.

Pradiniame tinklo etape, kai aktyvus tik koduotojas ir dekoderis, ši metrika žymima „EDLoss“. Realizacijoje jai papildomai pritaikomas konstantinis svoris (dauginama iš 10), kad klaidos signalas būtų pakankamai stiprus ir dominuotų ankstyvoje optimizacijos fazėje. Taip skatinama dekoderį kuo efektyviau generuoti stambų objekto tūrio išsidėstymą iš kiekvieno atskiro vaizdo matymo, nes ši dalis turi gražinti visų vėlesnių modulio paklaidų įtaką.

Kai po kontekstinės suliejimo dalies įjungiam rafinerio pakopa, pradeda veikti papildomas nuostolis „RLoss“. Jis apskaičiuojamas identiška formule, bet taikomas tarp rafinerio pateikiamo patobulinto tūrio ir pagrindinės tiesos. Paprastai „RLoss“ dažniausiai mažesnis už „EDLoss“, nes į šį etapą patenka jau sujungtas, preliminariai teisingas objektas. Nors abiejų nuostolių matematinė struktūra sutampa, jų funkcinė paskirtis skiriasi: „EDLoss“ griežtai tvarko globalią formą, o „RLoss“ subtiliai koreguoja lokalius neatitikimus – ypač tuos, kurių nepavyko išspręsti suliejimo modulyje, pavyzdžiui, plonas kojas, kraštines ar smulkias ertmes.

Kadangi „EDLoss“ ir „RLoss“ yra sumuojami, bendras optimizacijos procesas tampa dvigubo masto: pirma uždaroma stambi klaida tarp neapdoroto dekoderio volumetrinės išvesties ir realybės, o vėliau, rafineris kartu su „RLoss“ fokusuojasi į aukštesnio dažnio detales. Šis dviejų etapų mokymas leidžia architektūrai pasiekti aukštesnį „Intersection over Union“ rodiklį nei ankstesni rekonstrukcijos metodai.

2.6. Testavimo planas

Kuriant ir tobulinant 3D modelių generavimo sprendimus, itin svarbu atlikti nuodugnų testavimą, siekiant užtikrinti, kad pasiekti rezultatai būtų ne tik tikslūs, bet ir praktiškai pritaikomi. Testavimo planas gali apimti įvairius matavimo ir analizės metodus, kurie padeda geriau suprasti sistemos stiprybes, trūkumus bei galimybes tobulėti. Šiam tikslui gali būti suformuota išsami strategija, aprėpianti nuo paprasčiausio veikimo greičio matavimo iki detalių eksperimentų, skirtų patikrinti, kaip modelis reaguoja į skirtingus duomenų šaltinius, kokybės reikalavimus ar techninių išteklių apribojimus.

2.6.1. Veiklos testavimas

Veiklos testavimas (angl. performance testing) orientuojasi į kiekybinius rodiklius, kurie rodo, ar sprendimas gali būti efektyviai naudojamas realiose situacijose. Šioje dalyje fokusuojamasi į kelis esminius aspektus:

- **Mokymo trukmė ir išteklių naudojimas:** Vertinama, kiek laiko reikia modeliui apmokyti naudojant tam tikrą duomenų rinkinį. Matuojamas laikas, skirtas vienai epochai, bei bendra apmokymo proceso trukmė. Be to, stebimas CPU, GPU ir atminties naudojimas, siekiant įvertinti, ar sprendimas gali būti apmokytas naudojant įprastą aparatinę įrangą, ar reikia itin galingų resursų. Tokio tipo informacija padeda nustatyti, ar modelis bus tinkamas didelėms organizacijoms ar tyrimų laboratorijoms, kur trumpos mokymo trukmės ir ekonomiškas energijos vartojimas tampa svarbiais kriterijais.
- **Išvadų darymo (inference) greitis:** Nagrinėjama, kiek laiko reikia sugeneruoti 3D modelį iš pateikto vaizdo. Greitesnis išvadų darymo procesas leidžia modelį naudoti interaktyviose sistemose, pavyzdžiui, virtualioje realybėje, robotikoje, papildytos realybės programose ar gamybos linijose, kur greita reakcija ir rezultatų pateikimas per kelias sekundes arba net realiuoju laiku yra itin svarbūs.
- **Duomenų kiekio keitimas:** Tikrinama, kaip sprendimas reaguoja sumažinus duomenų kiekį. Pvz., galima patikrinti, ar mokymo trukmė bei išvadų darymo laikas auga linijiniu, kvadratiniu ar kubiniu mastu didėjant duomenims. Tai padeda numatyti, kokiose situacijose modelis išlieka praktiškas, o kada verta ieškoti alternatyvių sprendimų ar modelio optimizavimo strategijų.

2.6.2. Sistemos testavimas ir patvirtinimas

Sistemos testavimas (angl. system testing) apima visą 3D modelio generavimo ciklą – nuo pradinių duomenų apdorojimo etapų iki galutinės 3D formos generavimo ir jos kokybės vertinimo. Šiame kontekste aktualu patikrinti:

- **Duomenų paruošimo tinkamumas:** Įsitikinama, kad duomenų normalizacijos ir apdorojimo procedūros neklaidina modelio, o priešingai – pagerina rezultatų kokybę. Galima atlikti testus, kuriuose duomenys apdorojami skirtingais metodais (pvz., skirtingi normalizavimo ar duomenų papildymo būdai), kad būtų nustatytas optimalus variantas.
- **Modelio architektūros stabilumas:** Tikrinama, ar modelio architektūra nepraranda stabilumo keičiant hiperparametrus, pvz., mokymosi spartą ar partijos (batch) dydį. Atkreipiamas dėmesys, ar nėra požymių, kad modelis linkęs į treniravimo nestabilumą, pvz., kai klaidos funkcija nustoja mažėti arba pradeda svyruoti.
- **Tikslumo vertinimas:** IoU metrika, kaip pagrindinis kokybinis rodiklis, padeda objektyviai įvertinti, kiek sugeneruotas 3D modelis sutampa su atskaitos formomis. IoU reikšmių stebėjimas su skirtingais objektais, kamerų pozicijomis ar apšvietimo sąlygomis leidžia įsitikinti, kad modelis yra pakankamai lankstus ir geba gerai generalizuoti.

2.6.3. Modelio efektyvumo testavimas

Modelio efektyvumo testavimas orientuojasi į kokybinį rezultatų vertinimą įvairiose situacijose. Šio tipo testuose gali būti tiriama:

- **Objektų įvairovė:** Naudojami įvairių kategorijų objektai (pvz., lėktuvai, automobiliai, kėdės, suolai, monitoriai, lempos ir kt.) padeda suprasti, ar modelis sugeba gerai rekonstruoti tiek paprastas, tiek sudėtingas struktūras. Jei modelis puikiai tvarkosi su elementariais objektais, bet nesiseka su smulkiai detalizuotomis formomis, tai rodo kryptį, kurioje būtina tobulinti modelį ar papildyti duomenų rinkinį.
- **Ilgalaikis testavimas ir regresijos testai:** Atliekant testavimą ne vieną kartą, o reguliariai (pvz., pridedant naujų duomenų, keičiant modelio architektūrą ar hiperparametrus), galima stebėti, ar sprendimas ilgainiui gerėja. Regresijos testai padeda nustatyti, ar įvedus tam tikrą patobulinimą nebuvo netyčia pablogintas anksčiau buvęs rezultatas.

3. Eksperimentiniai 3D objekto formos rekonstrukcijos rezultatai ir įvertinimai

Toliau aprašyti visi rezultatai, yra išgauti nemodifikuojant jokių originalių Pix2Vox hiperparametrų. Mokymui visuomet duodamos 224x224 nuotraukos. Nuotraukos yra apdorojamos po 64 (batch size), kur galiausiai sudaro rekonstruotą objektą, kuris yra 32x32x32 vokselių tinklas. Apmokymas trunka 250 epochų. Naudojamas „Adam“ optimizacija. Mokymo greitis yra nustatytas į 0.001 koduotojui, decoderiui ir rafineriui. Mokymo greitis yra sumažinamas per pusę, pasaiekus 150 epochą.

Pateiktoje 3.1 lentelėje aprašomi 3D objektų generavimo modelio veikimo preliminarūs rezultatai įvairiose objektų kategorijose, palyginant juos su bazinio modelio (3D-R2N2) rezultatais. Vertinimui naudojama IoU metrika, kuri yra standartinis būdas įvertinti vokselių pagrindu generuotų 3D modelių tikslumą.

3.1 lentelė. Preliminarūs modelio rezultatai

Kategorija	**Pavyzdžių skaičius**	**Bazinė linija**	**t=0.20**	**t=0.30**	**t=0.40**	**t=0.50**
Lėktuvas	404	0.5130	0.6404	0.6618	0.6631	0.6481
Suolas	181	0.4210	0.5506	0.5729	0.5769	0.5637
Komoda	157	0.7160	0.7757	0.7828	0.7851	0.7841
Automobilis	749	0.7980	0.8290	0.8449	0.8518	0.8512
Kėdė	677	0.4660	0.5537	0.5618	0.5559	0.5364
Ekranas	109	0.4680	0.5166	0.5193	0.5128	0.4976
Lempa	231	0.3810	0.4619	0.4585	0.4404	0.4094
Garsiakalbis	161	0.6620	0.7000	0.6998	0.6927	0.6756
Šautuvas	237	0.5440	0.5880	0.6141	0.6188	0.6059
Sofa	317	0.6280	0.7103	0.7225	0.7232	0.7093
Stalas	850	0.5130	0.5863	0.5992	0.6005	0.5908
Telefonas	105	0.6610	0.7269	0.7401	0.7438	0.7394
Laivas	193	0.5130	0.5982	0.6153	0.6154	0.5992
Bendras Vidurkis			**0.6420**	**0.6549**	**0.6548**	**0.6427**

Lentelėje pateikiama dabartinio modelio veikimo analizė, lyginant rezultatus su baziniu modeliu (3D-R2N2) įvairiose objektų kategorijose. Tikslumo matui naudojama Intersection-over-Union (IoU) metrika, kuri rodo, kiek generuoto modelio tūris sutampa su realiu objektu. Kategorijos apima jau minėtus 13 objektų, tarp kurių pavyzdžiui yra automobiliai ar kėdės. Tiriamasis modelis nuosekliai lenkia bazinį variantą visuose slenksčio lygiuose, parodydamas reikšmingą pažangą.

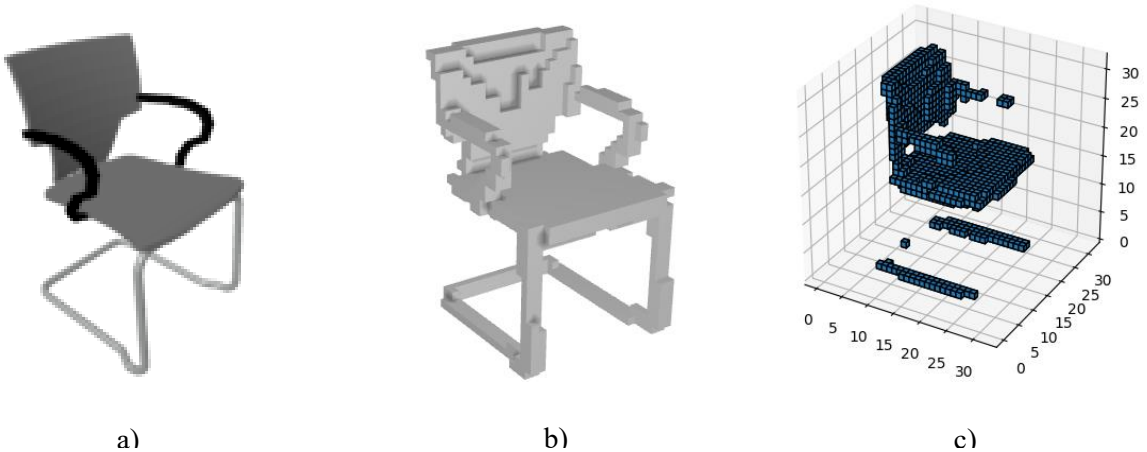
Analizė rodo, kad bendras tikslumas didėja nuo žemiausio slenksčio (t=0,20), aukščiausią reikšmę pasiekdamas ties t=0,30 (vidutinis IoU – 0,6549). Aukštesni slenksčiai, kaip t=0,50, nors ir padeda pašalinti triukšmą, praranda dalį smulkių detalių, todėl tikslumas ima mažėti. Tai rodo, kad optimalus slenkstis daugeliui kategorijų yra tarp t=0,30 ir t=0,40, kur pasiekiamas geriausias balansas tarp detalumo ir triukšmo pašalinimo.

Kai kurios kategorijos, tokios kaip automobiliai, komodos ir telefonai, demonstruoja aukščiausius IoU rezultatus, tikėtina dėl jų paprastos struktūros ir mažesnės variacijos. Tuo tarpu sudėtingesni objektai, kaip lempos ar kėdės, kurių formos yra plonesnės ir labiau fragmentuotos, turi žemesnius rezultatus. Šie duomenys rodo, kad modelis gerai generalizuoja paprastesnėse kategorijose, tačiau

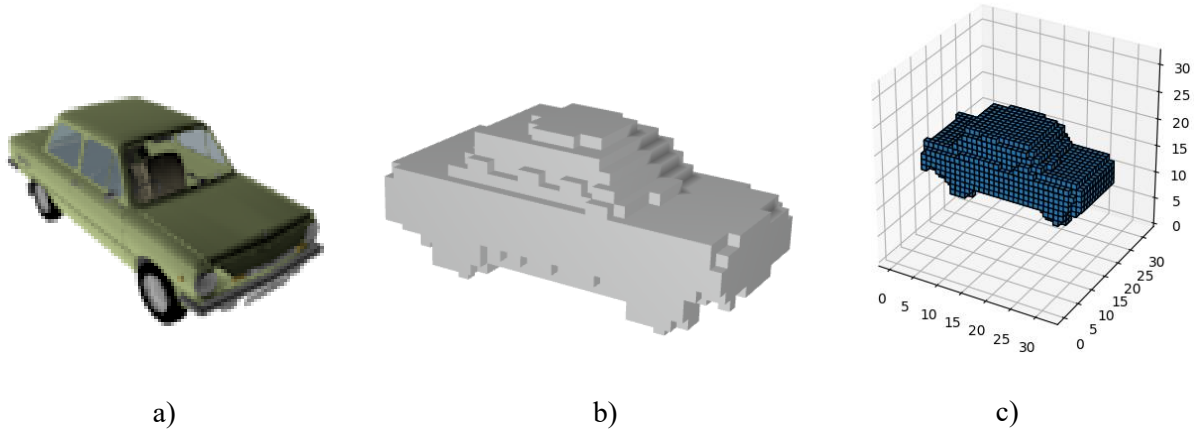
sudėtingesnėms reikalingas tolesnis tobulinimas, pavyzdžiui, papildomų duomenų ar geresnės architektūros integracija.

Bendrai, rezultatai pabrėžia dabartinio modelio efektyvumą ir galimybes tobulėti. Jis ne tik viršija bazinio modelio pasiekimus, bet ir parodo svarbius įžvalgesnius skirtumus tarp skirtingų kategorijų. Tai rodo, kad pasirinktos metrikos bei metodai yra tinkami, tačiau sudėtingesniems objektams reikia papildomų sprendimų, kad būtų pasiekti aukštesni tikslumo lygiai.

Paveiksluose 6.1 ir 6.2 pateikiami modelio veikimo rezultatų pavyzdžiai, atskleidžiantys, kuriose situacijose modelis pasiekia gerus rezultatus ir kuriose susiduria su sunkumais. Šie pavyzdžiai parodo, kaip skirtingos objektų formos ir detalės lemia modelio tikslumą generuojant 3D struktūras.



3.1 pav. Rezultatai 1: a) įvestis b) tikroji vokselinė reikšmė c) modelio sukurtas 3D objektas



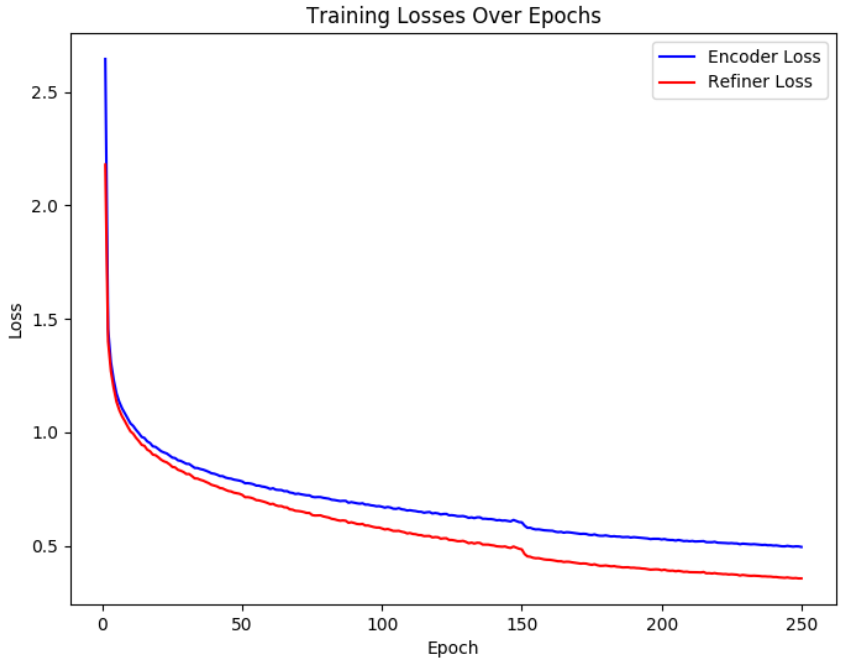
3.2 pav. Rezultatai 2: a) įvestis b) tikroji vokselinė reikšmė c) modelio sukurtas 3D objektas

Vienas iš ryškiausių iššūkių matomas 3.1 paveikslėlyje. Kai objektai turi daug smulkių detalių, modelis nesugeba jų tiksliai atkurti, todėl prarandama reikšminga objekto forma. Šią problemą ypač gerai iliustruoja kėdės atvaizdavimo pavyzdys. Kėdės dažnai turi plonas kojas, subtilias rankų atramas arba sudėtingas geometrines formas, kurios sunkiai atpažįstamos. Dėl to modelis šiose vietose nepakankamai užpildo vokselius arba apskritai jų neatkuria, kas lemia neišbaigtus

rezultatus. Šie rezultatai buvo pristatyti konferencijose “Data Analysis Methods for Software Systems”, kurio sertifikatas yra pateiktas prieduose 1 ir “IVUS”, kuris pateiktas prieduose 2.

Tuo tarpu automobiliai yra viena iš kategorijų, kur modelis pasiekia žymiai geresnių rezultatų, kaip matyti 3.2 paveikslėlyje. Automobilių formos yra paprastesnės, mažiau formų įvairumų, o pagrindiniai jų struktūros elementai yra stambesni ir lengviau atpažįstami. Modelis geba atkurti automobilio korpusą su gana aukštu tikslumu, nors smulkesnės detalės, kaip šoninio vaizdo veidrodėliai, lempos ar vidinės dalys, paprastai nėra atkuriamos. Tai galima paaiškinti tuo, kad modelis dėmesio skiria objekto bendrai formai.

Šie rezultatai rodo, kad modelio tikslumas stipriai priklauso nuo objekto formos sudėtingumo ir detalumo lygio. Smulkios struktūros ir didelė formų įvairovė kelia modelio gebėjimams rimtų iššūkių, tačiau paprastesni objektai su stambesnėmis detalėmis yra generuojami žymiai tiksliau. Tai pabrėžia poreikį toliau tobulinti modelį, siekiant geresnės smulkių detalių rekonstrukcijos ir platesnio pritaikymo įvairiose objektų kategorijose.



3.3 pav. Modelio apmokymo nuostoliai

3.3 paveikslėlyje pavaizduoti mokymo nuostoliai rodo, kad modelis per epochas veikia gerai, palaipsniui mažėja, bet nepasiekia plokščiakalnio. Tai reiškia, kad mokant per daugiau epochų ilgainiui bus pasiekta geresnių rezultatų.

3.2 lentelė. Atšildyto koduotojo rezultatai

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6392	0.6656	0.6754	0.6710
Suolas	181	0.4210	0.5630	0.5835	0.5896	0.5818
Komoda	157	0.7160	0.7937	0.7979	0.7939	0.7816
Automobilis	749	0.7980	0.8357	0.8498	0.8565	0.8568
Kėdė	677	0.4660	0.5590	0.5673	0.5643	0.5490
Ekranas	109	0.4680	0.5359	0.5365	0.5269	0.5074

Lempa	231	0.3810	0.4597	0.4584	0.4488	0.4286
Garsiakalbis	161	0.6620	0.7046	0.7037	0.6966	0.6789
Šautuvas	237	0.5440	0.5952	0.6197	0.6270	0.6166
Sofa	317	0.6280	0.7227	0.7312	0.7306	0.7156
Stalas	850	0.5130	0.5923	0.6030	0.6054	0.5986
Telefonas	105	0.6610	0.7352	0.7443	0.7441	0.7356
Laivas	193	0.5130	0.6029	0.6188	0.6209	0.6082
Bendras vidurkis			**0.6484**	**0.6605**	**0.6620**	**0.6526**

3.2 lentelėje pateikti rezultatai rodo modelio našumą, kai atšildyti kodavimo sluoksniai leidžia tinklui papildomai prisitaikyti prie mokymo duomenų savybių. Palyginti su baziniu scenarijumi, kai koduotojas lieka neužšaldytas, matyti nedidelis, bet nuoseklus IoU našumo pagerėjimas, ypač esant vidutiniam ir aukštesniam slenksčiui (t=0,30-0,40). Vėlesniems bandymams atliekamas mokymo duomenų apkarpymas. Testavimo ir patvirtinimo rinkiniai paliekami tokie patys, kad būtų galima įvertinti, kaip gerai modelis apibendrina tuos pačius testavimo ir patvirtinimo duomenis.

3.1. Duomenų šalinimas

Norint stebėti, kaip kinta objektų rekonstrukcijos tikslumas, priklausant nuo duomenų kiekio, naudojamas duomenų kiekis apmokymui yra dalimis mažinimas, kad pamatyti duomenų įtaką galutiniam rezultatui.

3.3 lentelė. Rezultatai su 80 % mokymo duomenims

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6441	0.6630	0.6662	0.6567
Suolas	181	0.4210	0.5596	0.5792	0.5824	0.5722
Komoda	157	0.7160	0.7832	0.7887	0.7883	0.7819
Automobilis	749	0.7980	0.8334	0.8473	0.8539	0.8552
Kėdė	677	0.4660	0.5560	0.5614	0.5562	0.5392
Ekranas	109	0.4680	0.5277	0.5286	0.5218	0.5052
Lempa	231	0.3810	0.4583	0.4546	0.4415	0.4173
Garsiakalbis	161	0.6620	0.7094	0.7099	0.7039	0.6905
Šautuvas	237	0.5440	0.5858	0.6077	0.6144	0.6045
Sofa	317	0.6280	0.7101	0.7190	0.7198	0.7100
Stalas	850	0.5130	0.5885	0.5984	0.6000	0.5934
Telefonas	105	0.6610	0.7272	0.7253	0.7206	0.7116
Laivas	193	0.5130	0.5994	0.6145	0.6169	0.6074
Bendras vidurkis			**0.6449**	**0.6551**	**0.6555**	**0.6462**

Sumažinus mokymo duomenis iki 80 %, šiek tiek sumažėjo objektų rekonstrukcijos tikslumas, kaip parodyta 3.3 lentelėje. Tačiau tikslumo sumažėjimas nėra labai didelis. Blogiausiu atveju tikslumas sumažėjo apie 1 %, o tai rodo, kad didinant duomenų kiekį tikslumas gali šiek tiek padidėti. Nors vidurkis ir sumažėja, kai kurių objektų, pavyzdžiui, lėktuvo ar automobilio, pokyčiai yra daug mažesni - jų sumažėjimas siekia apie 0,2 %. Tikėtina, kad taip atsitiko dėl to, kad šie objektai buvo apmokyti naudojant gerokai didesnę duomenų dalį nei kiti objektai.

3.4 lentelė. Rezultatai su 50 % mokymo duomenims

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6207	0.6448	0.6534	0.6476

Suolas	181	0.4210	0.5358	0.5550	0.5590	0.5511
Komoda	157	0.7160	0.7752	0.7811	0.7812	0.7764
Automobilis	749	0.7980	0.8254	0.8395	0.8469	0.8484
Kėdė	677	0.4660	0.5288	0.5341	0.5296	0.5151
Ekranas	109	0.4680	0.4935	0.4921	0.4842	0.4701
Lempa	231	0.3810	0.4311	0.4260	0.4112	0.3872
Garsiakalbis	161	0.6620	0.6909	0.6910	0.6854	0.6723
Šautuvas	237	0.5440	0.5737	0.5982	0.6069	0.6020
Sofa	317	0.6280	0.6897	0.7004	0.7043	0.7018
Stalas	850	0.5130	0.5631	0.5722	0.5737	0.5678
Telefonas	105	0.6610	0.7183	0.7256	0.7239	0.7193
Laivas	193	0.5130	0.5839	0.6011	0.6042	0.5944
Bendras vidurkis			**0.6249**	**0.6359**	**0.6374**	**0.6302**

Kaip parodyta 3.4 lentelėje, sumažinus naudojamų mokymo duomenų kiekį iki 50 %, rekonstrukcijos tikslumas sumažėja dar labiau - jis svyruoja apie 3 %, palyginti su 100 % mokymo duomenų. Modelis gerai veikia, kai pusė duomenų nenaudojama.

3.5 lentelė. Rezultatai su 20 % mokymo duomenims

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.5819	0.6036	0.6036	0.6066
Suolas	181	0.4210	0.4853	0.4933	0.4902	0.4758
Komoda	157	0.7160	0.7207	0.7150	0.7030	0.6831
Automobilis	749	0.7980	0.8147	0.8270	0.8325	0.8325
Kėdė	677	0.4660	0.4872	0.4854	0.4759	0.4561
Ekranas	109	0.4680	0.4511	0.4449	0.4344	0.4149
Lempa	231	0.3810	0.3818	0.3712	0.3541	0.3283
Garsiakalbis	161	0.6620	0.6590	0.6516	0.6385	0.6117
Šautuvas	237	0.5440	0.5423	0.5658	0.5721	0.5648
Sofa	317	0.6280	0.6596	0.6624	0.6574	0.6411
Stalas	850	0.5130	0.5222	0.5272	0.5259	0.5165
Telefonas	105	0.6610	0.6944	0.6962	0.6935	0.6820
Laivas	193	0.5130	0.5517	0.5597	0.5551	0.5410
Bendras vidurkis			**0.5904**	**0.5962**	**0.5936**	**0.5817**

Pašalinus 80 % duomenų, tikslumas šiek tiek labiau sumažėjo, kaip parodyta 3.5 lentelėje. Dabar tikslumas vidutiniškai sumažėjo apie 6-7 %. Šiuo metu objektai, kurie turi daugiau duomenų, taip pat pradėjo rodyti, kad jiems pradeda trūkti duomenų. Šie rezultatai rodo, kad didelės mokymo duomenų dalies pašalinimas turi įtakos rekonstrukcijos tikslumui, nors ji ir mažesnė, nei tikėtasi. Tikėtina, kad taip yra dėl modelio architektūros, kurioje naudojamas iš anksto apmokytas VGG16, todėl tikslumas išlieka gana aukštas. Kitas dalykas yra tai, kad nėra matomas, joks tiesioginis sąryšis su visu duomenų kiekiu ir rekonstrukcijos tikslumu. Tikėtasi, kad objektui turint didesnį kieki duomenų, tas objektas būtų paveiktas mažesniu tikslumo nukritimu. Atkreipus dėmesį į du objektus, kaip “Lėktuvas” ir “Ekranas” galima pastebėti, jog esant lėktuvo objektų didesniam kiekiui, rekonstrukcijos kiekis per šiuos eksperimentus sumažėjo beveik 7 %, kur ekrano rekonstrukcijos tikslumas nukrito 8.5 %. Tuomet yra objektai „Kėdė“ ir „Stalas“, kurie turi kelis kart daugiau duomenų, negu dauguma kitų objektų. Rekonstrukcijos tikslumai kėdei nukrito apie 8 %, kur stalams nukrito taipogi, apie 8 %. Tikslumo nukritimas šiek tiek varijuoja tarp objektų, tačiau nepastebėta, jokių išskirtinumų. Galima daryti prielaidą, kad didesnis tikslumo nukritimas

įvyksta su modeliui sunkiau apdorojamais objektais. Prie to prisideda objektų formos sudėtingumas bei skirtingų variacijų kiekiai. Kaip pavyzdys, objektas „Automobilis“ prarado vos 2.4 %. Tai rodytų, jog būtų vertinga paeksperimentuoti su pačio modelio architektūra. Todėl toliau atliekami testai yra modifikacijos atliktos pačiai architektūrai

3.2. Modelio architektūros modifikacijos

Siekiant, išgauti geresnius rezultatus, telkiamas dėmesys, į pačio modelio koduotojo architektūrą. Rezultatai yra aprašyti atlikus kelias architektūros perturbacijas, bei pilnai keičiant modeli. Pirmiausiai yra bandoma pridėti papildomus sluoksnius prie VGG16, kad arčiau atitiktų VGG19 architektūrą. Tuomet, vėliau yra keičiamas koduotojas, pritaikant ResNet18 architektūrą

3.6 lentelė. Pix2Vox-VGG19-1 rezultatai

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6401	0.6678	0.6790	0.6759
Suolas	181	0.4210	0.5615	0.5801	0.5833	0.5736
Komoda	157	0.7160	0.7827	0.7895	0.7895	0.7848
Automobilis	749	0.7980	0.8371	0.8508	0.8568	0.8563
Kėdė	677	0.4660	0.5577	0.5680	0.5671	0.5531
Ekranas	109	0.4680	0.5348	0.5363	0.5296	0.5134
Lempa	231	0.3810	0.4606	0.4601	0.4510	0.4319
Garsiakalbis	161	0.6620	0.7062	0.7099	0.7083	0.6974
Šautuvas	237	0.5440	0.5850	0.6148	0.6263	0.6208
Sofa	317	0.6280	0.7148	0.7232	0.7238	0.7149
Stalas	850	0.5130	0.5940	0.6064	0.6090	0.6021
Telefonas	105	0.6610	0.7292	0.7429	0.7472	0.7444
Laivas	193	0.5130	0.5956	0.6143	0.6202	0.6131
Bendras vidurkis			**0.6469**	**0.6604**	**0.6633**	**0.6557**

3.6 lentelėje pavaizduoti pirmos architektūros modifikacijos rezultatai, kai prie bazinio VGG16 pridėtas vienas papildomas konvoliucinis sluoksnis, parodė tik simbolinį vidutinio IoU padidėjimą. Pokytis praktiškai neviršijo kelių dešimtųjų, kas rodo, kad vienas papildomas sluoksnis nesukūrė pakankamai naujos galios, o parametrų skaičiaus augimas virto papildomu mokesčiu.

3.7 lentelė. Pix2Vox-VGG19-2 rezultatai

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6314	0.6556	0.6629	0.6572
Suolas	181	0.4210	0.5611	0.5815	0.5870	0.5788
Komoda	157	0.7160	0.7886	0.7963	0.7952	0.7820
Automobilis	749	0.7980	0.8342	0.8479	0.8543	0.8550
Kėdė	677	0.4660	0.5551	0.5644	0.5600	0.5435
Ekranas	109	0.4680	0.5278	0.5264	0.5142	0.4905
Lempa	231	0.3810	0.4609	0.4619	0.4525	0.4318
Garsiakalbis	161	0.6620	0.7034	0.7057	0.7011	0.6827
Šautuvas	237	0.5440	0.5897	0.6148	0.6211	0.6131
Sofa	317	0.6280	0.7106	0.7194	0.7185	0.7028
Stalas	850	0.5130	0.5859	0.5995	0.6035	0.5981
Telefonas	105	0.6610	0.7354	0.7401	0.7351	0.7239
Laivas	193	0.5130	0.5934	0.6105	0.6127	0.5999
Bendras vidurkis			**0.6436**	**0.6564**	**0.6577**	**0.6481**

3.7 lentelėje parodyti rezultatai, kai padidinus gylį ir taip priartinus koduotoją prie VGG19, rekonstrukcijos tikslumas pradeda kristi. Nors ir tikslumo praradimas nėra ženklus, ši tendencija leidžia teigti, kad vien gylio didinimas sekant VGG19 architektūrą, nebesuteikia reikšmingos naudos, nes architektūra išlieka hierarchiškai sekli ir linkusi dubliuoti jau turimus filtrus, neužtikrindama naujos semantinės informacijos.

3.8 lentelė. Pix2Vox-ResNet18 rezultatai

Kategorija	Pavyzdžių skaičius	Bazinė linija	t=0.20	t=0.30	t=0.40	t=0.50
Lėktuvas	404	0.5130	0.6464	0.6723	0.6838	0.6828
Suolas	181	0.4210	0.5811	0.5999	0.6073	0.6055
Komoda	157	0.7160	0.8107	0.8147	0.8146	0.8107
Automobilis	749	0.7980	0.8395	0.8524	0.8585	0.8597
Kėdė	677	0.4660	0.5714	0.5793	0.5786	0.5692
Ekranas	109	0.4680	0.5372	0.5386	0.5339	0.5225
Lempa	231	0.3810	0.4604	0.4620	0.4560	0.4409
Garsiakalbis	161	0.6620	0.7135	0.7128	0.7084	0.6990
Šautuvas	237	0.5440	0.5884	0.6189	0.6321	0.6328
Sofa	317	0.6280	0.7303	0.7369	0.7367	0.7292
Stalas	850	0.5130	0.6061	0.6144	0.6162	0.6110
Telefonas	105	0.6610	0.7448	0.7505	0.7531	0.7496
Laivas	193	0.5130	0.6077	0.6248	0.6315	0.6292
Bendras vidurkis			**0.6567**	**0.6682**	**0.6713**	**0.6666**

Kur kas reikšmingesnį pokytį pavyko pasiekti visiškai pakeitus VGG architektūrą į ResNet18, kurių rezultatai pavaizduoti 3.8 lentelėje. likučių blokai (angl. residual blocks), leido modeliui išmokti gilesnes, be gradiento gesimo, funkcijas, todėl bendras vidutinis IoU ties t = 0,30 pakilo iki 0,6682, o ties t = 0,40 – iki 0,6713, tai yra maždaug 1.6 % daugiau nei preliminarių rezultatų konfigūracijoje. Be to, kanalo redukcija nuo 512 iki 256 suderino koduotojo išvestį su dekoderio įvestimi, sumažindama parametrų infliaciją ir pagerindama signalo–triukšmo santykį.

3.9 lentelė. vidutinių rekonstrukcijos tikslumų rezultatai

Eksperimentas	t=0.20	t=0.30	t=0.40	t=0.50
Preliminarus	0.6420	0.6549	0.6548	0.6427
Atšildymas	0.6484	0.6605	0.6620	0.6526
80% duomenų	0.6449	0.6551	0.6555	0.6462
50% duomenų	0.6249	0.6359	0.6374	0.6302
20% duomenų	0.5904	0.5962	0.5936	0.5817
Pix2Vox-VGG19-1	0.6469	0.6604	0.6633	0.6557
Pix2Vox-VGG19-2	0.6436	0.6564	0.6577	0.6481
Pix2Vox-ResNet18	**0.6567**	**0.6682**	**0.6713**	**0.6666**

3.9 lentelėje yra surinkti visi bendri vidurkiai iš ankstesnių bandymų. Čia galima patogiau pamatyti rekonstrukcijos tikslumo pakitimus. Juodai paryškinti geriausiai gauti rezultatai, kas kaip jau minėta, būtų koduotojo architektūros pakeitimas į ResNet18. Tai parodo, jog galima išgauti aukštesnius tikslumo rezultatus optimizuojant modelio architektūrą. Šiuo atveju buvo atlikti bandymai tik su sluoksnių pakeitimais, tačiau tikriausiai galima pastumti šiuos procentus priekyn ir atliekant perturbacijas su hiperparametrais.

3.10 lentelė. Palyginamieji rezultatai pagal IoU metrika, tarp skirtingų modelių

Kategorija	3D-R2N2	MarrNet	Haoqiang Fan model	DISN	Extended YOLOv3	Pix2Vox-A	Pix2Vox-Resnet18
Lėktuvas	0.513	-	0.601	0.617	-	**0.684**	0.6838
Suolas	0.421	-	0.550	0.542	-	**0.616**	0.6073
Komoda	0.716	-	0.771	0.531	-	0.792	**0.8146**
Automobilis	0.798	-	0.831	0.770	-	0.854	**0.8585**
Kėdė	0.466	-	0.544	0.549	-	0.567	**0.5786**
Ekranas	0.468	-	0.552	**0.577**	-	0.537	0.5339
Lempa	0.381	-	**0.462**	0.397	-	0.443	0.4560
Garsiakalbis	0.662	-	**0.737**	0.559	-	0.714	0.7084
Šautuvas	0.544	-	0.604	**0.680**	-	0.615	0.6321
Sofa	0.628	-	0.708	0.671	-	0.709	**0.7367**
Stalas	0.513	-	0.606	0.489	-	0.601	**0.6162**
Telefonas	0.661	-	0.749	0.736	-	**0.776**	0.7531
Laivas	0.513	-	0.611	0.602	-	0.594	**0.6315**
Bendras vidurkis	0.571	0.57	0.640	0.594	0.602	0.661	**0.6713**

3.10 lentelėje yra pavaizduoti rezultatai iš kelių kitų sukurtų modelių. Šie modeliai taipogi naudojosi ShapeNet duomenų rinkiniu, todėl yra lengviau palyginti rezultatus. Tačiau, kaip lentelėje yra matoma, ne visi moksliniai straipsniai publikavo pilnus, kiekvienam objektui skirtus, rezultatus. Visi rezultatai yra pateikti pagal IoU metriką, kur didžiausias skaitmuo reiškia geresnį rezultatą. Paryškinti skaitmenys, nurodo, jog tai yra geriausias rezultatas skirtam objektui.

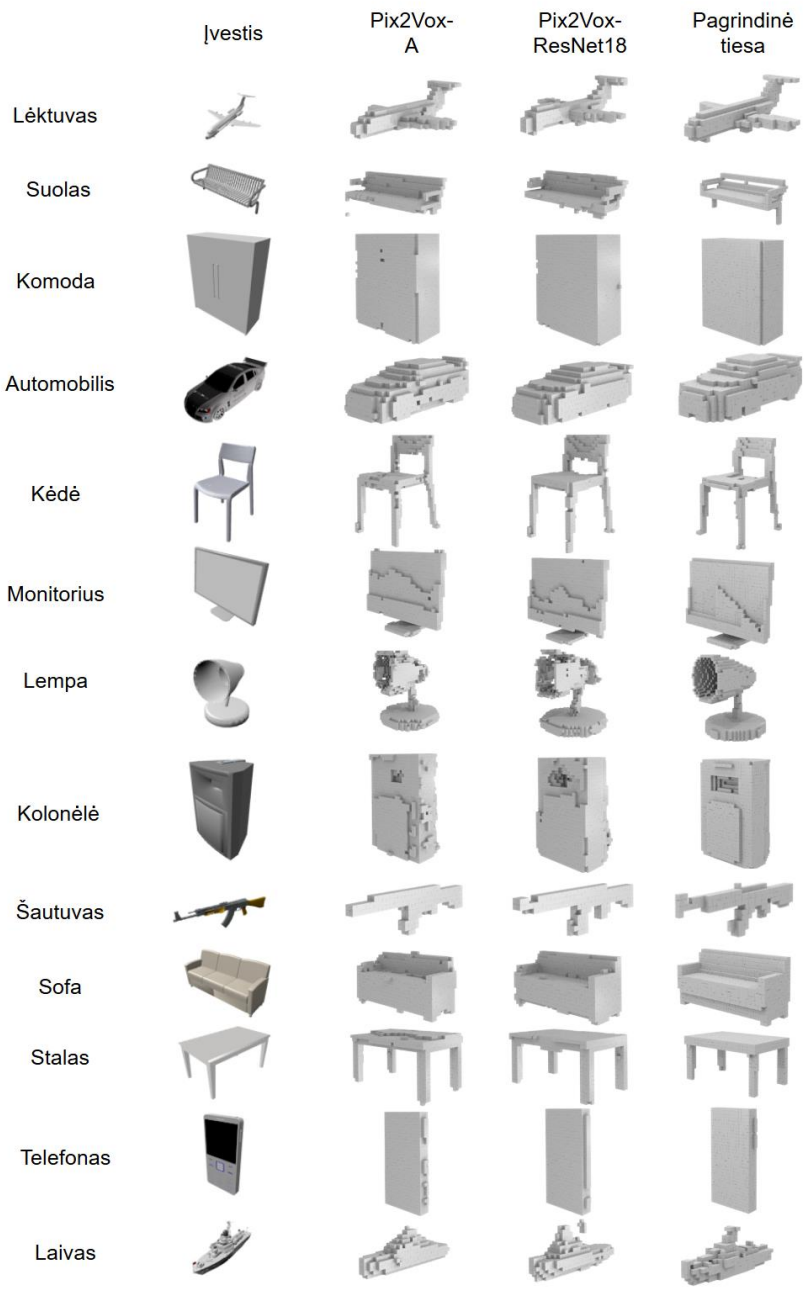
Gauti eksperimentiniai rezultatai aiškiai rodo, kad modelio tikslumas priklauso nuo turimų duomenų kiekio ir pasirinktų architektūrinių sprendimų. Kai treniravimui palikome visą pradinį rinkinį, vidutinis IoU pasiekė aukštą lygį ir tapo atspirties tašku visiems tolesniems bandymams. Mažindami duomenų kiekį pastebėjome nuoseklų, bet ne dramatišką, smukimą: mokant su pusės dydžio rinkiniu vidutinis tikslumas sumažėjo maždaug trimis procentais, o su vos penktadaliu duomenų krito dar keturiais. Tai rodo, kad iš anksto apmokytas koduotojas vis dar geba išlaikyti nemažą dalį vizualios informacijos, tačiau sudėtingesnės, smulkesnių detalių reikalaujančios kategorijos, tokios kaip lempos ar kėdės, patiria didesnį nuosmukį nei paprastesni objektai, pavyzdžiui automobiliai ar komodos. Vis dėlto, nebuvo pastebėta, kad duomenų kiekis, būtų tiesiogiai susijęs, su tikslumo pokyčiu. Pavyzdžiui, kai duomenų kiekis buvo sumažintas iki 20 %, objektas „Stalas“ prarado 7.5 % tikslumo nuo originalios vertės, o objektas „Automobilis“ prarado apytiksliai 2 %. Duomenų rinkinyje, stalų yra apie 1000 daugiau nei automobilių. Tai reiškia, kad duomenų kiekis turi mažesnį kiekį įtakos, kai jų kokybė yra prastesnė. Paprastai, objektas yra įsivaizduojamas, kaip daiktas su keturiomis kojomis, kurios pritvirtintos prie pagrindinės lentos, skirtos padėti daiktams. Čia ir kyla problema, jog stalo dizainų stilių yra daug, kur skirtingų stalų gali siekti šimtais. Kita vertus, automobiliai išlieka panašūs, kur visi turi keturis ratus ir atitinkamą karkasą. Šie atvejai yra pastebimai visame ShapeNet duomenų rinkinyje, kur objektų variacija yra plati. Būtent dėl tokių duomenų rinkinio požymių yra manoma, jog didžioji dalis tikslumo buvo prarasta, dėl duomenų variacijos. Tačiau, negalima to apibūdinti, kaip blogu rinkiniu. Didelė variacija užtikrina, kad modeliai nepersimokys. Nors ir tikslumo vidurkis išgaunamas mažesnis, tai

duoda modeliui geresnę galimybę rekonstruoti skirtingus objektus ir įgauti geresnį supratimą objektų savybėms.

Architektūrinių modifikacijų rezultatai parodė, jog vien tik didinant VGG sluoksnių skaičių nepavyksta pagerinti rekonstrukcijos kokybės, papildomi filtrai padidino parametrų skaičių, tačiau nesuteikė naujos, semantiškai reikšmingos informacijos. Kita vertus, perėjimas prie ResNet18 suteikė šiek tiek geresnius rezultatus. Likučių jungtys leido išlaikyti stabilų gradientą, o kanalų sumažinimas geriau suderino koduotojo išvestį su dekoderio įvestimi. Šis derinys pakėlė vidutinį IoU tikslumą maždaug 1.5 %. Kadangi tai buvo vienintelis testas su ResNet architektūra, yra verta išbandyti ir su naujesnėmis ResNet architektūromis. Taipogi, galima ir pasinaudoti architektūros dalimi, kur suteikus mažesnį parametrų kiekį, galima būtų išgauti rezultatą greičiau ar net ir su aukštesniu rezultatu. Tuomet smulkiau aprėpta šio projekto dalis yra hiperparametrai. Modelio apmokymas yra ilgai trunkantis procesas ir šiuo atveju buvo skirta 250 epochų. Daryti eksperimentus su hiperparametrais gali suteikti įžvalgos į modelio greitinimą ar aukštesnio tikslumo išgavimą. Šiame projekte tuo nebuvo užsiimta, kadangi keičiant modelio konfigūraciją, tektų praleisti didelius kiekius laiko arba skaičiavimo resursų. Vis dėlto, tai yra zona, kur gali suteikti naudingus rezultatus.

3.3. Rekonstruoti atsitiktinai parinkti skirtingi objektai

Projekto metu buvo suformuota vaizdų seka, kurioje užfiksuoti modelių rekonstrukcijos rezultatai. Siekiant užtikrinti nešališką vertinimą, pavyzdžiai buvo parinkti atsitiktinai, todėl tyrime nedalyvavo iš anksto atrinkti, itin gerai atpažįstami atvejai. Tačiau, kad išvengti spalvų dokumente, yra nuspręsta nedėti jokių objektų su spalvomis, todėl tenka atsitiktinai rinkti objektą iki tol, kol jame nebus spalvų. Kiekvienas sugeneruotas vaizdas atspindi vieno objekto rekonstrukciją, kuri lyginama su pradine įvestimi ir turima pagrindinės tiesos apimtimi. Tai suteikia galimybę vizualiai patikrinti, kokiu mastu trimačio vaizdo atkūrimas atitinka realų daikto tūrį. Šie rezultatai yra pavaizduoti 3.4 paveikslėlyje.



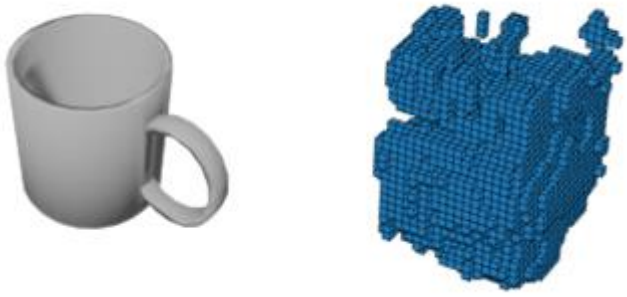
3.4 pav. Visų skirtingų modelių rekonstrukcijų pavyzdžiai

Nuotraukose pateikti keli paprasti objektai, tokie kaip telefonas ar komoda, kurių forma yra stati ir simetriška. Dėl aiškių kontūrų šie daiktai buvo rekonstruoti palyginti tiksliai, o net ir smulkesni briaunų netikslumai nepaveikė bendro tūrio suvokimo. Vis dėlto šiame pačiame rinkinyje pastebėtas objektas „Suolas“ aiškiai išsiskiria. Jo rekonstrukcija pameta keletą svarbių detalių, tačiau pagrindinė problema, kad įvesties vaizdas neatitinka pagrindinės tiesos. Šis neatitikimas leidžia daryti prielaidą, kad duomenų rinkinyje egzistuoja objektų, kuriuos modelis apdoroja ypač netiksliai. Tokia situacija rodo galimą treniravimo duomenų šališkumą, nes modelis galėjo dažniau susidurti su suolams nebūdingomis formomis arba per mažai matė tinkamų jų pavyzdžių. Atsitiktinai nustatytas nesėkmingas pavyzdys kelia klausimą dėl viso duomenų rinkinio kokybės. Jei rinkinyje yra daugiau panašių atvejų, nei buvo matyta šiame patikrinime, jie gali daryti didelę įtaką galutiniam modeliui. Tokie iškraipymai gali klaidinti mokymosi algoritmą, skatinti per stiprų prisitaikymą prie neteisingų formų ir taip bloginti bendrą rekonstrukcijos tikslumą. Kadangi objektų

pašalinimas ar pakeitimas tvarkingesniais pavyzdžiais sumažintų bendrus netikslumus, būtų tikslinga patikrinti naudojamus duomenis. Papildomas tyrimas leistų nuspręsti, ar būtina pašalinti probleminius objektus iš apmokymo proceso. Jei paaiškėtų, kad klaidų mastas didesnis nei tikėtasi, vertėtų suformuoti griežtesnius filtravimo kriterijus arba papildomai sunorminti duomenis. Tokia praktika sumažintų nepageidaujamą įtaką mokymosi eigai ir užtikrintų, kad rekonstrukcijos rezultatai geriau atspindėtų realų objektų tūrį. Galutinis tikslas išlieka tas pats – pasiekti, kad modelis kuo tiksliau atkurs įvairių formų daiktus nepriklausomai nuo jų kategorijos ir sudėtingumo.

3.4. Nematytas Objektas

Tiriant modelio gebėjimą rekonstruoti visai naujus vaizdus, buvo atliktas bandymas su anksčiau nematytu daiktu. Buvo tikėtasi, kad neuroninis tinklas nepasieks tokių pačių rezultatų, kokie gauti treniravimo ir validacijos rinkiniuose, nes treniruotės metu šio objekto kategorija nebuvo pateikta. Hipotezė pasitvirtino, o rezultatai pateikiami 3.5 paveikslėlyje. Eksperimentui pasirinktas puodelis, nes tai plačiai paplitęs kasdienis daiktas, turintis aiškiai atpažįstamą formą ir rankenėlę, kuri palanki rekonstruojant trimatę geometriją.



3.5 pav. Modeliui nematytas objektas „Puodelis“

Puodelio vaizdas buvo naudojamas atliekant kelis bandymus, kurių tikslas buvo rasti, kokios įvesties savybės lemia kokybiškiausią rekonstrukciją. Paaiškėjo, kad reikšmingiausi veiksniai yra pradinio paveikslėlio skiriamoji geba ir pati objektą užimanti paveikslėlio dalis. Jeigu vaizdo dydis arba mastelis neatitiko to, prie kurio tinklas buvo pritaikytas treniruotės metu, kokybė pastebimai prastėjo. Pavyzdžiui, sumažinus nuotrauką daugiau nei du kartus, atkuriamos formos tapo dar labiau netikslios. Panaši neigiama įtaka pasireiškė ir tuomet, kai puodelis užėmė didesnį paveikslėlio dalį, nes modelis nebegalėjo patikimai išskirti jokių bruožų. Bandymų rezultatai parodė, kad modelis vis dar sugeba sunkiai išgauti bendrą siluetą, vis dėlto trūksta kokybės. Gauti rezultatai leidžia spręsti, kad modelio generalizavimo gebėjimai išlieka riboti. Jei tinklas anksčiau nematė specifinės objekto kategorijos, jis remiasi bendromis formų užuominomis, todėl negali tiksliai atkurti elementų, kurie treniravimo rinkinyje buvo reti arba visai nepasitaikė. Puodelio atveju rankenėlė pasirodė esanti sudėtingas segmentas, kuris visai nebuvo atkuriamas. Be to, įvesties mastelio klaidos sustiprino rekonstrukcijos netikslumus, nes tinklas treniruojamas su pastoviu, normalizuotu vaizdų dydžiu. Taigi tyrimas parodė, jog neuroninis tinklas negali laisvai adaptuotis naujiems objektams ir vaizdo mastelio pokyčiams, kol nėra pateiktas pakankamai įvairus treniravimo duomenų rinkinys ar taikomos papildomos normalizavimo procedūros.

3.5. Realaus pasaulio objektas

Kadangi numatoma pritaikyti trimatės rekonstrukcijos technologiją praktinėse situacijose, buvo nuspręsta išmėginti tinklo veikimą naudojant ne sintetinį, o realiai nufotografuotą daiktą. Palyginimui pasirinkta kėdė, nes tai dažnas namų aplinkos objektas, lengvai fotografuojamas ir turintis konstrukciją, su kuria tinklas jau susipažino mokymo metu. Eksperimentas atliktas paprasta kamera, be papildomo apšvietimo valdymo ar scenos paruošimo. Nuotrauka buvo sumažinta iki tinklo priimamo 224 × 224 pikselių dydžio, pašalintas fonas.

3.6 pav. Modeliui matytas objektas kėdė, tačiau nuotrauką gauta nufotografavus, tikrą objektą

Kaip pavaizduota 3.6 paveikslėlyje, rezultatas rodo, kad rekonstruotas objektas išlaiko atpažįstamą kėdės siluetą: matyti keturios kojos, sėdimoji dalis ir viena pagrindinė atlošo plokštuma. Vis dėlto atkūrimas nėra identiškas fotografuotam daiktui. Originali kėdė turi kelias siauras nugaros atramas, kurios nuotraukoje aiškiai matomos, tačiau tinklas jų neatkūrė. Vietoje jų sugeneruotas vientisas atlošas, panašesnis į kito stiliaus kėdę, tikriausiai dažniausiai pasitaikančią treniravimo rinkinyje. Tokia neatitiktis rodo dvi galimas problemas. Pirmoji problema, kad modelis galėjo persimokyti, todėl gavęs panašų kontūrą priskiria jį labiausiai matytam šablonui ir atkuria ne individualų, o tipišką rinkinio pavyzdį. Antroji problema, kad realaus daikto paviršius turi medienos raštą, blizgesio pakitimus ir šešėlius, kuriuos sukuria nelygus apšvietimas. Šios savybės keičia ryškumo bei spalvų pasiskirstymą, todėl neuroninis tinklas gali sunkiau nustatyti tikslias briaunas ir formas. Tai būtų duomenų rinkinio trūkumas, kadangi duomenys yra sintetiniai ir neturi daug realybės bruožų. Papildomi bandymai parodytų, kiek stipriai tokiems skirtumams paveikūs skirtingi objekto paskirtys. Galima daryti prielaidą, kad modelis geriau susitvarkys su daiktais, turinčiais lygius, be rašto paviršius, pavyzdžiui, plastikines dėžes ar metalinius vamzdžius. Tokiais atvejais tekstūrų įtakos beveik nėra, tad sintetiniai ir realūs pavyzdžiai tampa panašesni. Vis dėlto tam, kad kėdės, stalai ir kiti mediniai arba audiniu dengti daiktai būtų atkuriami patikimiau, būtina praplėsti duomenų rinkinį realiomis nuotraukomis. Vienas praktiškas žingsnis būtų įtraukti papildomus duomenų paruošimo metodus, pavyzdžiui, tekstūros sumažinimą, ryškumo bei kontrasto variacijas ir fonų įvairovę. Tokie pakeitimai sumažintų atotrūkį tarp skaitmeninių ir fotografuotų vaizdų, todėl tinklas prisitaikytų prie realaus pasaulio triukšmo ir apšvietimo pokyčių. Eksperimento rezultatai leidžia teigti, kad vien sintetinių vaizdų nepakanka gerai bendrinei rekonstrukcijai. Nors bazinė forma atkurta teisingai, praktinėje užduotyje reikėtų aukštesnės detalumo kokybės, ypač jei galutinis modelis bus naudojamas dizaino ar gamybos procesuose. Todėl siekiant pritaikyti šią

technologiją realiems objektams, rekomenduojama papildyti mokymosi rinkinį realiomis fotografijomis, atlikti duomenų normalizaciją ir pritaikyti technikas, mažinančias persimokymo riziką.

3.6. Rekonstrukcijos sparta

Eksperimentas buvo atliktas siekiant nustatyti, ar rekonstrukcijos sparta kinta įtraukus skirtingus mokymo ir duomenų paruošimo pakeitimus. Rezultatai pateikti 3.11 lentelėje, kurioje užfiksuota, kiek laiko truko atkurti visus validacijos rinkinio objektus naudojant aštuonis skirtingus modelius. Visi bandymai buvo vykdomi su tomis pačiomis validacijos nuotraukomis, kurios buvo taikomos pradinio apmokymo metu. Pilnas trukmės stulpelis rodo bendrą laiką sekundėmis, sunaudotą visų objektų rekonstrukcijai. Vidutinio laiko stulpelis nurodo, kiek sekundžių vidutiniškai reikėjo vienam objektui. Greta pateikti trumpiausi ir ilgiausi vieno objekto rekonstrukcijos laikai.

3.11 lentelė. Objektų rekonstrukcijos spartos rezultatai

Eksperimentas	Pilnas laikas (s)	Vidutinis Laikas (s)	Trumpiausias laikas (s)	Ilgiausias laikas (s)
Preliminarus	43.47	0.010	0.008	0.024
Atšildymas	50.93	0.012	0.008	0.022
80% duomenų	41.05	0.009	0.008	0.019
50% duomenų	40.94	0.009	0.008	0.046
20% duomenų	41.14	0.009	0.008	0.043
Pix2Vox-VGG19-1	42.67	0.010	0.008	0.024
Pix2Vox-VGG19-2	42.76	0.010	0.008	0.073
Pix2Vox-ResNet18	38.42	0.009	0.007	0.021

Duomenys parodė kelias svarbias tendencijas. Pirmiausia, atšildytas modelis, kuriame buvo leista koduotojo sluoksniams toliau mokytis, rekonstrukcijai skyrė beveik aštuoniomis sekundėmis daugiau nei preliminarus modelis. Šis skirtumas rodo, kad išplėstas mokymas gali padidinti skaičiavimų sudėtingumą, todėl rekonstrukcija tampa lėtesnė. Kita vertus, skirtingų treniravimo duomenų kiekių nuo 80 % iki 20 % poveikis spartai buvo minimalus. Visų trijų sumažinto rinkinio bandymų bendras laikas išliko apie 41 sekundę. Tai reiškia, jog duomenų kiekis nesudaro įtakos modelio atkūrimo greičiui. Vertinant VGG architektūros versijas galima pasakyti, kad jų rezultatai yra labai artimi. Pirmoji versija rekonstravo per kiek daugiau nei 42 sekundes, o antroji, per beveik tą patį laiką. Galiausiai išsiskyrė Pix2Vox-ResNet18 modelio rezultatas. Jam prireikė mažiau nei 39 sekundžių, o vidutinis vieno objekto laikas liko toks pats kaip ir VGG bandymuose. Tai leidžia teigti, kad ResNet architektūra gali būti ne tik tikslesnė, kaip parodyta ankstesniuose tikslumo bandymuose, bet ir spartesnė atliekant rekonstrukcijas.

Apibendrinant galima teigti, kad rekonstrukcijos sparta iš esmės priklauso nuo tinklo architektūros ir specifinių treniravimo strategijų. Koduotojo atšildymas padidino bendrą laiką, o duomenų kiekio sumažinimas reikšmingos įtakos neturėjo. ResNet18 pasirodė greičiausias, todėl ši architektūra galėtų būti naudingesnė realaus laiko taikymuose, kuriuose rekonstrukcija turi būti atliekama kaip įmanoma greičiau.

3.7. Skaičiavimo sudėtingumai ir ištekliai

Visi mokymai ir bandymai buvo atlikti „NVIDIA GeForce GTX 1080 Ti“ vaizdo plokštėje, kurios atmintis siekia 11 GB. Toks aparatinis pagrindas leidžia vertinti pokyčius praktiškai vienodomis sąlygomis, todėl pasikeitimai matomi tik dėl architektūros ar duomenų rinkinio ypatybių. Duomenų rinkinio apkarpymas tiesiškai sumažino trukmę. Kai mokymas buvo vykdomas su 80 %, 50 % ir 20 % pradinės imties, epochos trukmė sumažėjo atitinkamai iki 92 %, 71 % ir 24 % viso laiko. Esant mažesniam duomenų kiekiui, daromi validacijos testai, santykinai sudaro didesnę dalį mokymo laiko, kadangi jo dydis buvo išlaikytas toks pats. Visoje sekoje GPU atminties poreikis beveik nekito.

3.12 lentelė. Kiekvieno naudoto modelio parametrų kiekiai.

Modelis	Parametrų kiekis
Pix2Vox	117,780,083
Pix2Vox-VGG-1	114,244,467
Pix2Vox-VGG-2	114,835,059
Pix2Vox-ResNet18	117,195,891

Pagal 3.12 lentelėje pateiktus duomenis, matosi parametrų kiekiai, kuriuos modelis turėjo. Didesnis skaičius reiškia didesnį modelio sudėtingumą, kuris daro įtakos apmokymo spartai bei išgautam tikslumui. Tačiau ne visuomet reiškia, kad pasiekus didesnį skaičių, kad gautas rezultatas taps labiau tenkinamas. Šiuo atveju, Pix2Vox-ResNet18 modifikacija turi didžiausią kieki parametrų, tačiau sugebėjo išgauti aukštesnį tikslumą rekonstrukcijoje, bei tai atliko greičiau.

3.8. Diskusija

Eksperimentai rodo, kad „Pix2Vox“ pagrįstu modeliu galima gauti tinkamas 3D rekonstrukcijas įvairiomis sąlygomis, tačiau taip pat išryškėja, kokią įtaką našumui turi duomenų ir modelio pasirinkimas. Apmokytas pagal pakankamai didelį sintetinių duomenų rinkinį, tinklas atliko tūrines rekonstrukcijas, kurių bendra forma buvo tenkinamai suderinta su pagrindinėmis tiesomis – patikimai užfiksuota stambi objektų geometrija. Naudojamų duomenų kiekis sumažino išgaunama rekonstrukcijos tikslumą, tačiau buvo tikėtasi, jog tikslumas nukris gerokai daugiau. Modeliui mokantis su 20 % duomenų kiekiu, nukrito apie 7 % IoU tikslumas Modelio architektūra atliko pastebimą, bet palyginti nedidelį vaidmenį rekonstrukcijos našumui. Originalioje Pix2Vox sistemoje naudojamas VGG pagrįstas koduotojas, o eksperimentai su VGG19 pagrindu davė preastesnius rezultatus. Pakeitus koduotoją „ResNet18“ variantu, bendras tikslumas šiek tiek pagerėjo (maždaug 1–2 % IoU), be to, tai turėjo įtakos mokymo dinamikai. Tai parodo priešingą rezultatą, kur atliktas perėjimas nuo VGG prie ResNet-152 pagerino IoU tik ~0,3 %, kai buvo naudojami vaizdai iš 20 krypčių, ir netgi buvo prastesnis už VGG, kai naudojamas vienas vaizdas[38]. Tai gali reikšti, jog pridedant vis daugiau sluoksnių, rezultatas tik prastėja.Vieno vaizdo aplinkoje ResNet18 ir VGG19 modeliai veikė panašiai, nė vienas iš jų neturėjo didelio pranašumo nė vienoje konkrečioje objektų kategorijoje, o tai rodo, kad pagrindo pasirinkimas nėra dominuojantis veiksnys, jei jis yra pakankamai išraiškingas. Vis dėlto, yra pastebima, kad ResNet18 modelis parodė praktinį pranašumą: jis šiek tiek greičiau atliko skaičiavimus, greičiau nei VGG pagrįsti modeliai, greičiausiai dėl efektyvesnės ResNet architektūros. Šis rezultatas rodo, kad ResNet pagrindu sukurtas Pix2Vox gali būti ne tik toks pat tikslus kaip VGG variantas, bet ir skaičiavimo požiūriu lengvesnis - tai patraukli diegimo ypatybė. Taipogi, Pix2Vox dizainas literatūroje jau laikomas efektyviu (pranešama, kad jis yra iki 24 kartų greitesnis už senesnį 3D-

R2N2 metodą)[37]. Vienas iš eksperimentų metu nustatytas iššūkis – modelio apibendrinimas duomenims, su nematytais objektais. Vertinant realaus pasaulio objektų vaizdus (kurie skiriasi nuo švarių sintetinių vaizdų, naudotų mokymui), rekonstrukcijos kokybė akivaizdžiai suprastėjo. Modelis nesugeba kokybiškai rekonstruoti nematytų objektų ir realaus pasaulio objektų naudojantis viena 2D nuotrauka.Kitas apribojimas susijęs su smulkių detalių ir tekstūruotų paviršių atkūrimu. Vokselinis išvesties vaizdavimas savaime riboja detalumo lygį, kurį galima pastebėti. Dažnai yra pametami objektų bruožai, kaip kojos ar atramos ant kėdžių ar objektų apvalumai. Tačiau, šia bėdą galima būtų ištaisyti didinant rekonstruojamo tinklelio dydį, kur šiame projekte naudojama buvo 323, galima didinti iki 643. Tiktais yra problema, kad didinant tinklelio dydį, modelio apmokymo laikas kyla eksponentiškai. Visame projekte, buvo atsižvelgiama tik į rezultatus gautus iš vieno 2D paveikslėlio. Ateityje galima stebėti pokyčius su daugiau kadrų. Natūraliai, modeliui gaunant daugiau informacijos, turėtų būti išgaunamas rezultatas, tačiau įmanoma susidurti su sunkumais, kur objektas nėra pilnai suvokiamas, kas gali lydėti į prastesnius rezultatus. Galiausiai yra daug sričių, kur galima plėsti tobulinimą: duomenų kokybės užtikrinimas, pridėti daugiau duomenų, hiperparametrų keitimai, architektūros sluoksnių keitimai, pridėti skirtingų skiriamųjų gebų paveikslėlių apmokymui, skirti apmokymui tikrojo pasaulio nuotraukų, didinti vokselių tinklelio dydį. Taip pat galima keisti rekonstruojamų objektų reprezentaciją iš vokselių į taškų debesį, kas galėtų išgauti smulkesnes detales.

Išvados

1. Atlikus aukštos kokybės duomenų rinkimo ir apdorojimo užduotį, buvo sėkmingai paruoštas išsamus „ShapeNetCore“ duomenų rinkinys. Duomenų rinkinio įvairovė užtikrino, kad modelis galėtų apdoroti įvairias objektų kategorijas bei formas. Tačiau apribotas klaidinančiais objektais, kurie gali būti priskirti ne tai kategorijai.
2. Pasirinkta Pix2Vox architektūra buvo sėkmingai pritaikyta generuoti 3D objektus iš vieno vaizdo. Naudota architektūra parodė tenkinamus rezultatus, integruojant pažangius koduotojo, dekoduotojo, kontekstinio susiliejimo ir rafinerio modulius, kurie užtikrino stabilų modelio mokymą ir gerą tikslumą.
3. Buvo atlikti keli architektūrinai naudojamo modelio pokyčiai, kur buvo sekama pagal VGG-19 modelio architektūra pridedant papildomus sluoksnius ir pilnai perrašant su ResNet18 architektūra. ResNet18 architektūros pritaikymas gražino aukštesnį tikslumo ir spartos rezultatą.
4. Modelio tikslumas buvo išsamiai vertintas naudojant Intersection-over-Union metrika. Rezultatai parodė, kad pasirinktas vertinimo metodas leidžia objektyviai įvertinti modelio našumą ir lyginti jį su kitais literatūroje minimais sprendimais. Tokie rezultatai kaip 0,8518 IoU automobilio kategorijoje parodė reikšmingą pažangą, palyginti su baziniu modeliu.
5. Rezultatai parodė, kad modelis efektyviai generalizuoja paprastesnės objektų formas, tokias kaip automobiliai, tačiau susiduria su iššūkiais atkuriant sudėtingas struktūras, pavyzdžiui, kėdes. Šie rezultatai yra svarbi įžvalga tolimesniam modelio tobulinimui, siekiant geresnio smulkių detalių atkūrimo. Ataskaita dokumentuoja ne tik modelio privalumus, bet ir galimas tobulinimo kryptis, užtikrinant mokslinį sprendimo pagrįstumą.

Literatūros sąrašas

1. CHOY, C.B. ir kt. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. In . 2016. p. 628–644. .
2. WANG, N. ir kt. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In . 2018. .
3. LIN, C.-H. ir kt. Magic3D: High-Resolution Text-to-3D Content Creation. In . 2022. .
4. FURUKAWA, Y. - PONCE, J. Accurate, Dense, and Robust Multiview Stereopsis. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* . 2010. Vol. 32, no. 8, p. 1362–1376. .
5. CHANG, A.X. ir kt. ShapeNet: An Information-Rich 3D Model Repository. In . 2015. .
6. ACHLIOPTAS, P. ir kt. Learning Representations and Generative Models for 3D Point Clouds. In . 2017. .
7. QI, C.R. ir kt. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In . 2016. .
8. QI, C.R. ir kt. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In . 2017. .
9. WANG, Y. ir kt. Dynamic Graph CNN for Learning on Point Clouds. In . 2018. .
10. KULIKAJEVAS, A. ir kt. 3D Object Reconstruction from Imperfect Depth Data Using Extended YOLOv3 Network. In *Sensors* . 2020. Vol. 20, no. 7, p. 2025. .
11. RIEGLER, G. ir kt. OctNet: Learning Deep 3D Representations at High Resolutions. In . 2016. .
12. GRAHAM, B. ir kt. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In . 2017. .
13. MITRA, N.J. ir kt. Structure-aware shape processing. In *ACM SIGGRAPH 2014 Courses* . New York, NY, USA: ACM, 2014. p. 1–21. .
14. KIPF, T.N. - WELLING, M. Semi-Supervised Classification with Graph Convolutional Networks. In . 2016. .
15. LECUN, Y. ir kt. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE* . 1998. Vol. 86, no. 11, p. 2278–2324. .
16. WU, R. - ZHENG, C. Learning to Generate 3D Shapes from a Single Example. In . 2022. .
17. FAN, H. ir kt. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In . 2016. .
18. THE ENGINEERING TOOLBOX (2013) Distance between 3D Points. In *https://www.engineeringtoolbox.com/distance-relationship-between-two-points-d_1854.html* . 2013. .
19. REZATOFIGHI, H. ir kt. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. In . 2019. .
20. ZHANG, D. ir kt. An efficient approach to directly compute the exact Hausdorff distance for 3D point sets. In *Integrated Computer-Aided Engineering* . 2017. Vol. 24, no. 3, p. 261–277. .
21. GOODFELLOW, I.J. ir kt. Generative Adversarial Networks. In . 2014. .
22. KINGMA, D.P. - WELLING, M. Auto-Encoding Variational Bayes. In . 2013. .
23. HO, J. ir kt. Denoising Diffusion Probabilistic Models. In . 2020. .

24. SHI, B. ir kt. DeepPano: Deep Panoramic Representation for 3-D Shape Recognition. In *IEEE Signal Processing Letters* . 2015. Vol. 22, no. 12, p. 2339–2343. .
25. ELMAN, J.L. Finding Structure in Time. In *Cognitive Science* . 1990. Vol. 14, no. 2, p. 179–211. .
26. WU, J. ir kt. MarrNet: 3D Shape Reconstruction via 2.5D Sketches. In . 2017. .
27. MARR, D. *Vision*. . [s.l.]: The MIT Press, 2010. ISBN 9780262289610.
28. GROUEIX, T. ir kt. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In . 2018. .
29. XU, Q. ir kt. DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction. In . 2019. .
30. CHEN, Z. - ZHANG, H. Learning Implicit Fields for Generative Shape Modeling. In [interaktyvus]. 2018. Prieiga per internetą: <http://arxiv.org/abs/1812.02822>.
31. GENOVA, K. ir kt. Local Deep Implicit Functions for 3D Shape. In . 2019. .
32. KANAZAWA, A. ir kt. Learning Category-Specific Mesh Reconstruction from Image Collections. In . 2018. .
33. TANG, J. ir kt. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. In . 2023. .
34. LIU, R. ir kt. Zero-1-to-3: Zero-shot One Image to 3D Object. In . 2023. .
35. BLENDER FOUNDATION Blender. . .[s.l.]: Blender Foundation. https://www.blender.org/, 2024. .
36. PASZKE, A. ir kt. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In . 2019. .
37. XIE, H. ir kt. Pix2Vox: Context-aware 3D Reconstruction from Single and Multi-view Images. In [interaktyvus]. 2019. Prieiga per internetą: <http://arxiv.org/abs/1901.11153>.
38. THANGKTRAN - C_MAZIK Pix2Vox-ResNet. In *https://github.com/thangktran/Pix2Vox-ResNet* . 2022. .

Priedai

1 **DAMSS 2024 Konferencija**

DAMSS
DATA ANALYSIS METHODS FOR SOFTWARE SYSTEMS

CERTIFICATE OF PARTICIPATION

This is to certify that

Algirdas Pocius

Kaunas University of Technology

has attended the 15th conference

Data Analysis Methods for Software Systems

held in Druskininkai, Lithuania, November 28 - 30, 2024
and has presented the paper

Voxel-Based 3D Object Generation from Single Images Using an Enhanced Deep Learning Architecture

Program Committee

Dr. Jolita Bernatavičienė

Druskininkai, November 29, 2024

2 IVUS 2025 Konferencija