VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS INSTITUTAS
INFORMATIKOS KATEDRA

Generatyviniai besivaržantys neuroniniai tinklai 3D modeliams generuoti iš 2D nuotraukų

Generative Adversarial Networks for 3D Model Reconstruction from 2D Images

Magistro baigiamasis darbas

Atliko:	Marius Bieliauskas	(parašas)
Darbo vadovas:	dr. Olga Kurasova	(parašas)
Recenzentas:	dr. Aistis Raudys	(parašas)

Vilnius – 2023

Turinys

Santrauka

Darbe atliekama literatūros apžvalga ir įgyvendinamas dirbtinio intelekto metodas, kuris, naudojant generatyvinius besivaržančius tinklus (angl. *Generative Adversarial Networks* arba GAN), galėtų iš vienos dvimatės nuotraukos sukonstruoti trimatę fotografuoto objekto reprezentaciją. Apžvelgiami tradiciniai trimačio skaitmeninimo metodai, bendra GAN teorija, įvairūs GAN metodų patobulinimai, esami darbai, atliekantys trimatį skaitmeninimą su GAN pagalba, bei 3D modelių išlyginimo ir kokybės gerinimo metodai. Galiausiai aprašomi atlikti analizuojamų metodų pritaikymo darbai ir gauti rezultatai.

Raktiniai žodžiai: Dirbtinis intelektas, GAN, fotogrametrija, 3D skaitmeninimas.

Summary

A literature review is performed and an artificial intelligence based method is implemented, which, using generative adversarial networks (GAN), would be able to construct three-dimensional representations of objects captured in a photograph. Traditional three-dimensional digitization methods, general GAN theory, various improvements to GAN methods, current work on three-dimensional digitization with the help of GAN, and methods for smoothing and improving the quality of 3D models are reviewed. Finally, the application and results of the analyzed methods are described.

Keywords: Artificial intelligence, GAN, photogrammetry, 3D digitization.

1. Įvadas

Magistro baigiamajam darbui pasirinkta tema ”Generatyviniai besivaržantys neuroniniai tinklai 3D modeliam generuoti iš 2D nuotraukų”. Darbo metu nagrinėjami dirbtinio intelekto metodai, konkrečiai generatyviniai besivaržantys neuroniniai tinklai, toliau GAN (angl. *Generative Adversarial Networks*) ir jų pritaikymas norint išgauti trimatę objektų, esančių vienoje ar keliose dvimatėse nuotraukose, reprezentaciją.

Dirbtinio intelekto ir tiksliau gilaus mokymo srityje daugiausiai sėkmės yra susilaukę diskriminuojantys (angl. discriminative) modeliai, kurie yra naudojami klasių priskyrimui prie duomenų, tačiau generatyviniai modeliai, kurie gali sukurti naują informaciją, ilgą laiką atsiliko savo kokybe. GAN tinklai buvo prasiveržimas generatyvinių modelių srityje, ypač užduotyse susijusiuose su vaizdine informacija. Nuo pirmojo GAN pasiūlymo 2014 metais Ian J. Goodfellow ir kt. parašytame straipsnyje tuo pačiu pavadinimu [GPM+14], ši metodologiją labai greitai tobulėjo ir privedė prie itin gerų rezultatų kuriant naują informaciją, kuri yra sunkiai atskiriama nuo realios. Vieni labiausiai stebinančių rezultatų gauti žmonių veidų generavimo srityje. 2017 metais Tero Karras ir kt. parašytame straipsnyje [KAL+18] gauti fotorealistiški neegzistuojančių žmonių veidai, be jokių ženklų, kad modelis tiesiog atsiminė ir atkartojo jau matytus veidus iš mokymo duomenų. GAN tobulėjimą šioje srityje galima pamatyti 4 paveikslėlyje.

GAN tinklai yra viena naujausių ir aktualiausių dirbtinio intelekto technologijų, dirbanti su vaizdine medžiaga. Vienas mažiau išnagrinėtų GAN pritaikymų yra fotogrametrijos ir trimačių modelių kūrimo (skaitmeninimo) srityje. 3D modeliavimo sujungimas su GAN technologijomis aprašomas JiaJun Wu ir kt. straipsnyje [WZX+16] bei V.V. Kniaz ir kt. straipsnyje [KRK19]. Šis sujungimas gali būti pritaikomas kultūrinio paveldo atkūrimui, pramogų ir ypač kompiuterinių žaidimų srityse, bei kituose srityse, kuriose svarbus objektų 3D skaitmeninimas. Jeigu 3D modeliavime galėtų būti matomas panašus tobulėjimas kaip ir kitose GAN srityse, šios technologijos ženkliai sumažintų dabartinius kaštus anksčiau minėtose 3D modelių panaudojimo srityse, kadangi šiuo metu geriausios 3D skaitmeninimo technologijos reikalauja brangių gylio matavimo ar lazerinio skenavimo įrankių, norint išgauti reikiamą kiekį informacijos.

Magistro baigiamojo darbo tikslas yra atlikti trimačių modelių skaitmeninimui naudojamų, generatyviniais besivaržančiais neuroniniais tinklais pagrįstų, metodų bei potencialių jų patobulinimų analizę. Siekiant darbo tikslo buvo atliekami šie uždaviniai:

1. Įgyvendinti veikiantį GAN tinklą, kuris galėtų iš dvimačių nuotraukų išgauti fotografuojamų objektų trimates reprezentacijas.

2. Modifikuoti literatūros analizės metu apžvelgtus dirbtinius neuroninius tinklus, kad gauti kiek įmanoma geresnius skaitmeninimo iš vienos nuotraukos rezultatus.

3. Pritaikyti automatinio išlyginimo (angl. smoothing) algoritmą, kuris galėtų patobulinti gautų 3D modelių raišką bei kokybę.

2. Literatūros apžvalga

Siekiant suprasti visas šiuo metu paskleistas mokslines žinias apie trimačių modelių skaitmeninimą, atlikta literatūros analizė, kurios metu buvo renkama informacija apie šiuo metu tam naudojamas technologijas ir metodus, jų efektyvumą ir galimas tobulinimo galimybes. Literatūros apžvalga apims aktualią fotogrametrijos ir trimačio skaitmeninimo teoriją, esamus metodus ir įgyvendinimus, jų pasiekimus ir trūkumus, bei GAN technologijas ir jų pritaikymą fotogrametrijos rezultatų tobulinimui. Taip pat apžvelgiami trimačių modelių išlyginimo (angl. smoothing) ir raiškos padidinimo (angl. upscaling) algoritmai, kurie galėtų būti pritaikyti galutinio skaitmeninto 3D modelio kokybei pagerinti.

2.1. 3D skaitmeninimas ir fotogrametrija

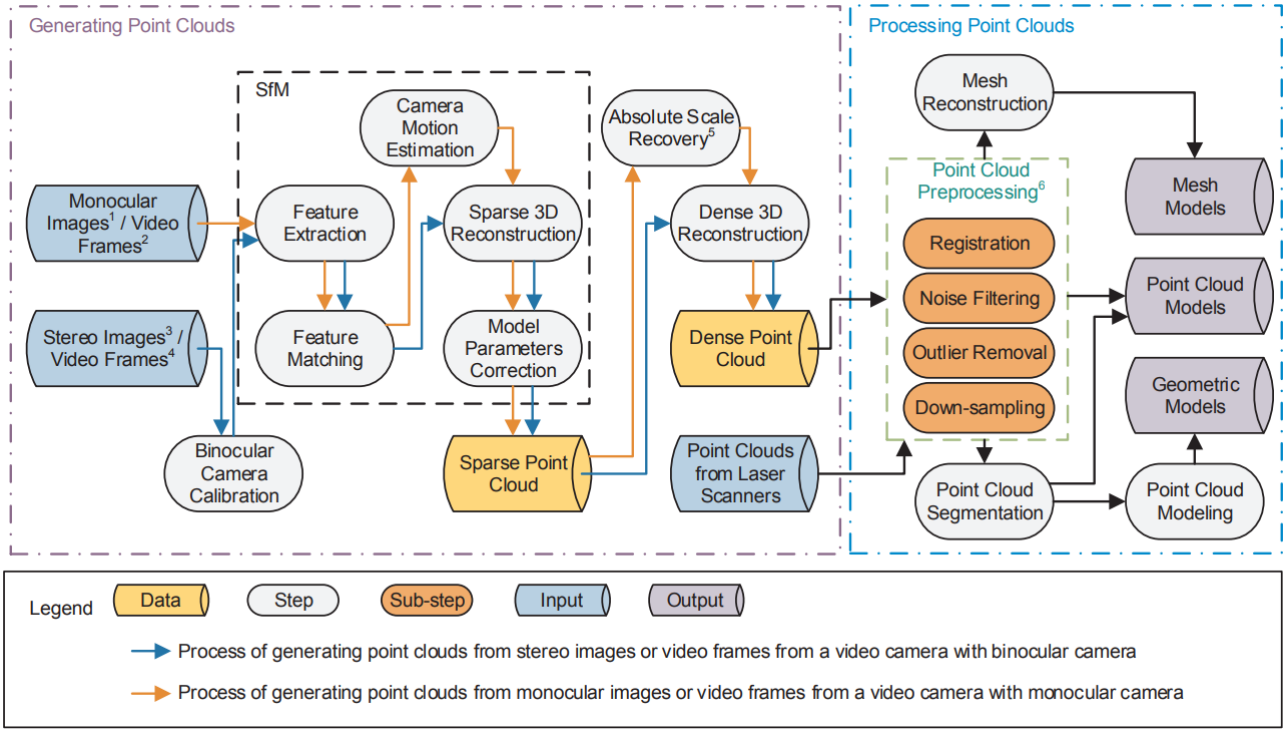
Vienas pagrindinių šio magistro baigiamojo darbo rezultatų turės būti suskaitmenintas trimatis modelis, sukurtas pagal realaus objekto atvaizdą. Trimatis modelis – tai skaitmeninis objekto atvaizdavimas sudarytas iš rinkinio (tinklo) sujungtų taškų arba tūrinių pikselių (vokselių), trimatėje erdvėje. Šie modeliai dažniausiai naudojami trimatei kompiuterinei grafikai kompiuteriniuose žaidimuose, filmuose, architektūroje, interjero dizaine, kultūrinio paveldo išsaugojimui bei analizei, ir kt.

3D modeliavimas nagrinėjamas jau ilgą laiką, nuo pat jo atsiradimo praeito amžiaus aštuntame dešimtmetyje. Savaime suprantama, kad per tą laikotarpį šia tema buvo pasiūlyta daugelis skirtingų metodų ir algoritmų. Modelių kūrimas pagal realius objektus, neskaitant rankinio atkartojimo, gali būti skaidomas į dvi dalis – fotogrametrinius metodus ir skenavimą (dažniausiai lazerinį skenavimą).

Fotogrametrija – tai mokslo šaka, tirianti objektų formos, matmenų ir (arba) padėties nustatymą iš fotografinių nuotraukų. Kai kalbama apie fotogrametriją, dažnai kalbama apie tradicinį iš oro arba iš orbitos gautų žemės paviršiaus nuotraukų tyrimą, tačiau šiam darbui pritaikoma artimojo nuotolio fotogrametrija, o tiksliau trimačių modelių išgavimas iš nuotraukų. Vienas standartinių fotogrametrijos metodų pritaikymo pavyzdys matomas Jūratės Sužiedelytės-Visockienės ir kt. 2015 metų straipsnyje [SBM+15] kuriame bandoma išgauti trimatę informaciją apie architektūros paveldo deformacijas. Straipsnyje minima, kad nors su lazeriniais skenavimais gaunami geresni trimačio skaitmeninimo rezultatai, dažnai lazerinio skenavimo galimybės nėra ir turi būti atliekama nuotraukų analizė.

Trimatis skenavimas – kitaip nei fotogrametrijoje, vietoje nuotraukų yra naudojami specializuoti prietaisai skirti atstumų nustatymui trimatėje erdvėje. Dažniausiai šiam tikslui naudojami lazeriais pagrįsti skaitytuvai, tačiau tam tikrose srityse naudojamos ir kitos technologijos, kaip kompiuterinė ar viršgarsinė tomografija.

2018 metais Zhiliang Ma ir Shilong Liu atliko tuo metu prieinamų trimačio skaitmeninimo (rekonstrukcijos) metodų palyginimą civilinės inžinerijos srityje [ML18]. Autoriai iš viso išanalizavo 95 skirtingus straipsnius, iš kurių 88, kaip įvestį naudojantys monofoninių ir stereofoninių kamerų vaizdus arba vaizdo įrašus bei lazerinių skaitytuvų išvestį, buvo aprašomi šioje apžvalgoje. Jų



1 pav. Trimačio skaitmeninimo žingsniai [ML18]. Pastabos: 1 – vaizdai gauti iš monofoninių kamerų, 2 – vaizdo įrašai gauti iš monofoninių kamerų, 3 – vaizdai gauti iš stereofoninių kamerų, 4 – vaizdo įrašai gauti iš stereofoninių kamerų.

parašytame straipsnyje galima rasti glaustą aprašymą apie bendrą skaitmeninimo procesą bei tam naudojamus algoritmus.

Autoriai dalina trimatį skaitmeninimą į du žingsnius: taškinių debesų generavimą ir jų apdorojimą. Pirmajame žingsnyje, taškiniai debesys yra generuojami iš vaizdinės medžiagos, bandant iš daugelio skirtingais kampais ir atstumais paimtų nuotraukų išgauti taškų, patalpintų ant viso atvaizduojamo objekto paviršiaus, koordinačių rinkinį. Šis žingsnis nėra pritaikomas naudojant lazerinius skaitytuvus, kadangi šių prietaisų išvestis, nuskenavus objektą, ir yra taškų debesis. Antrajame žingsnyje šis taškų rinkinys yra apdorojamas, tam kad sukurti norimą trimačio modelio išvestį. Apibendrintą objekto trimačio skaitmeninimo procesą galima matyti 1 paveikslėlyje.

2.1.1. Taškinių debesų generavimo algoritmai

Ma ir Liu savo apžvalgoje [ML18] rado, kad taškinių debesų generavimo žingsnis susideda iš septynių mažesnių, iš eilės atliekamų žingsnių: ypatybių išgavimo (angl. feature extraction), ypatybių prilyginimo (angl. feature matching), kameros judesio įvertinimo (angl. camera motion estimation), retos 3D rekonstrukcijos (angl. sparse 3D reconstruction), modelio parametrų koregavimo (angl. model parameters correction), absoliutaus (tikrojo) mastelio atkūrimo (angl. absolute scale recovery) ir galiausiai tankios 3D rekonstrukcijos (angl. dense 3D reconstruction). Pirmieji penki šio proceso žingsniai dar bendrai vadinami struktūra iš judesio, arba SfM (angl. Structure from Motion).

Ypatybių išgavimo tikslas yra surinkti ypatybių taškus, kurie atspindi pradinę skaitmeninamo objekto struktūrą, iš vaizdinės medžiagos. Šiame žingsnyje yra naudojami ypatybių taškų detek-

toriai, kurie randa jų poziciją fotografuojamoje ar filmuojamoje scenoje, ir taškų aprašytojai (angl. descriptors), kurie aprašo taškų savybes. Autorių apžvelgtuose straipsniuose rasti tokie detektoriaus metodai ir jų veikimo principai:

- SIFT (angl. Scale Invariant Feature Trasform) detektorius [Low04] – šiame metode iš pradinių nuotraukų yra sugeneruojami įvairaus mastelio vaizdų rinkiniai – mastelio erdvės (angl. scale-space), pritaikant konvoliuciją su Gauso funkcija, kuri matoma formulėje (1), x ir y formulėje atitinka pikselių koordinates, o σ atitinka gauso suliejimo (angl. Gaussian blur) koeficientą. Tada juos vieną iš kito atimant gaunami Gauso skirtumo vaizdai (angl. difference-of-Gausian images), kuriuose kiekvienas pikselis yra lyginamas su kaimyniniais pikseliais ir ieškoma ekstremumo taškų, kurie vėliau tampa ypatybių taškais.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{1}$$

- SURF (angl. Speeded-Up Robust Features) detektorius [BET+08] – vėlgi generuojama mastelio erdvė tačiau vietoje originalių nuotraukų yra naudojami jų vientisi vaizdai (angl. integral images), kurie kiekvienoje vaizdo koordinatėje laiko sumą visų pikselių verčių, esančių stačiakampyje į viršų ir į kairę nuo tos koordinatės. Tai leidžia vietoje Gauso funkcijos naudoti kvadratinius filtrus, kadangi pikselių verčių sumavimas tam tikroje vaizdo dalyje reikalauja žinoti tiktai kampų vertes ir yra skaičiuojamas daug greičiau. Taip pat vietoje Gauso skirtumo yra skaičiuojamas Hesiano determinantas kiekvienam vaizdui mastelio erdvėje ir pagal jį randami ypatybių taškai.

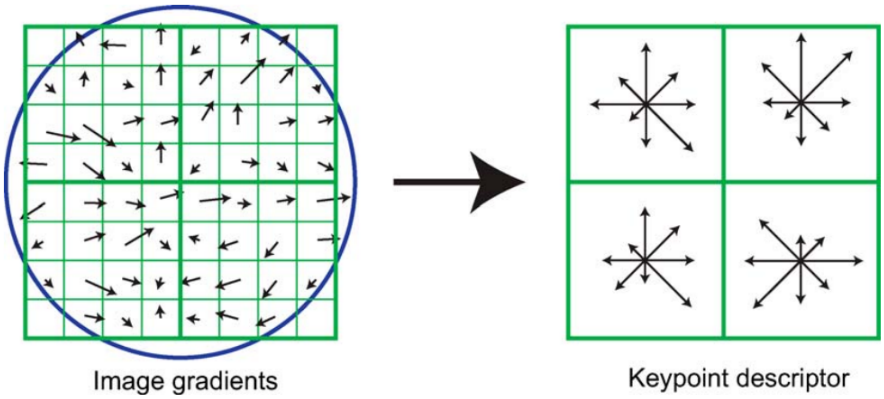
- ASIFT (angl. affine-SIFT [MY09] – tai SIFT metodo papildymas, kuriame prie mastelio erdvės yra pridedami sugeneruoti vaizdai, imituojančiais kameros pasukimą.

- Harris kampų detektorius [HS88] – tai paprastesnis algoritmas, kuris bando rasti nuotraukose esančių objektų kampus (kraštus) stebint staigius pikselių verčių pasikeitimus.

- FAST (angl. Features from Accelerated Segment Test) [RD06] – kampų radimo metodas, kuris greitesnis už Harris algoritmą. Jis kiekvienam pikseliui analizuoja 16 kaimyninių pikselių, išsidėsčiusių apskritimu nuo jo, ir jeigu tam tikra jų dalis peržengia nusistatytą šviesumo slenkstį šis pikselis yra laikomas kampo dalimi.

Deskriptoriai:

- SIFT aprašytojas [Low04] – apart ypatybių aptikimo, SIFT metodas taip pat susideda ir iš ypatybių aprašymo žingsnio. Jame vektoriai aprašantys ypatybių taškus yra generuojami remiantis vaizdo taškų, esančių aplink ypatybės tašką, gradiento dydžiais ir kryptimis. Vizualų to pavaizdavimą galima matyti 2 paveikslėlyje.

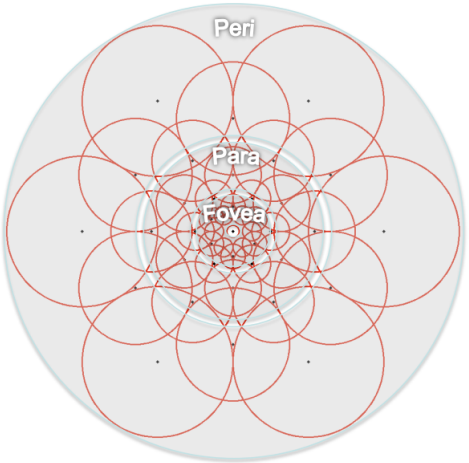


2 pav. Ypatybę aprašančio vektoriaus sukūrimas sujungiant aplink ypatybės tašką esančių pikselių gradientus SIFT metode [Low04].

- SURF aprašytojas [BET+08] – ypatybių taškai aparašomi sumuojant pikselių reikšmes mažuose 4×4 kvadratėliuose tam tikru spinduliu aplink šiuos taškus. Prieš sumuojant, kvadratėliams yra pritaikomi Haar bangelių filtrai horizontalia (x) ir vertikalia (y) kryptimis (formulė (2), kur t yra x arba y ašies reikšmės, o $\frac{1}{2}$ laikomas filtro vidurio tašku).

$$\varphi(t) = \begin{cases} 1 & \text{Jei, } 0 \leq t < \frac{1}{2} \\ 0 & \text{Jei, } \frac{1}{2} \leq t \leq 1 \end{cases} \tag{2}$$

- FREAK (angl. Fast REtinA Keypoint) [AOV12] – šis metodas renka apskritime aplink ypatybės tašką esančius pikselius žmogaus akies tinklaine pagrįstu metodu, t.y. imamos mažesnės grupės esant arčiau ypatybių taško ir didesnės grupės esant toliau. Taip taškams esantiems arčiau ypatybės taško yra suteikiama didesnė vertė (svoris) nei taškams esantiems toliau. Šio metodo rezultate gautas aprašas yra dvejetainė eilutė apskaičiuota atliekant vieno bito Gauso skirtumo operacijas tarp rinkinių porų. Kaip gaunami taškų rinkiniai galima pamatyti 3 paveikslėlyje.



3 pav. Pikselių rinkimas aplink ypatybės tašką FREAK metode. Visi apskritimai yra imlūs laukai (angl. receptive fields) pikselių, pagal kurios bus skaičiuojama ypatybės tašką aprašanti dvejetainė eilutė [AOV12].

- MSD (angl. Multi-Scale Descriptor) [SC13] – aprašymo generavimas šiame metode pagrindžiamas pikselių, esančių aplink ypatybės tašką, šviesumo (angl. intensity) gradientais, keliuose skirtinguose masteliuose.

Ypatybių prilyginimo algoritmai bando rasti sutapimus tarp gautų ypatybių taškų lyginant nuotraukų poras. Tam naudojami apytikslių arčiausių kaimynų algoritmai, arba ANN (angl. Aproximate Nearest Neighbours), kurie skaičiuoja Euklido atstumus tarp ypatybių taškų aprašų. Kad panaikinti klaidingus atitikimus ir išlaikyti objekto geometrinę informaciją, yra naudojami RANSAC [FB81] arba ORSA [RGL+14] algoritmai, imantys nuotraukų poras ir pažingsniui skaičiuojantys scenos epipolinę geometrinę informaciją (Epipolinė geometrija – tai stereo matymo geometrija, kai dvi kameros mato tą pačią trimatę sceną iš dviejų skirtingų padėčių).

Kameros judesio įvertinimo tikslas yra kiekvienai nuotraukai surasti ją paveikiančius kameros parametrus. Šie parametrai gali būti vidiniai, kaip kameros objektyvo židinio nuotolis ir radialinis iškraipymas arba tai gali būti išoriniai parametrai, kaip kameros pasukimas, pajudinimas ar pakreipimas. Kai turima nuotraukų pora, dvi matricos gali nusakyti jų epipolinės geometrijos informaciją – esminė matrica (angl. essential matrix), kuri laiko tiktai išorinius parametrus, ir pagrindinė matrica (angl. fundamental matrix), kuri laiko ir vidinius ir išorinius parametrus. Jų radimui atitinkamai yra naudojami du, penkių taškų [Nis04] ir aštuonių taškų [Har97], algoritmai, kurie pagal savo pavadinimus paima arba penkis arba aštuonis prilygintus taškus nuotraukose ir pagal juos skaičiuoja vieną ar kitą matricą. Kadangi esminė matrica nelaiko vidinių kameros savybių, ji ir penkių taškų algoritmas naudojami tiktai tada, kai kameros vidinės savybės jau yra žinomos ir nereikia švaistyti laiko jų ieškant.

Sekantis žingsnis, reta 3D rekonstrukcija, siekia apskaičiuoti taškų padėtis trimatėje erdvėje ir gauti fotografuojamą sceną ar objektą atitinkantį taškų debesį, naudojant praeituose žingsniuose gautus ypatybių taškų tarpusavio atitikimus ir kiekvienos nuotraukos kameros parametrus. Tam naudojamas triangulacijos algoritmas ([LCS+16], [YCH+13], [KL17]).

Turint retą scenos taškinį debesį yra atliekamas kameros parametrų koregavimas. Šiame žingsnyje bandoma ištaisyti ir sulyginti visų nuotraukų parametrus. Tam yra naudojamas ryšulio reguliavimo algoritmas (angl. Bundle adjustment), pagrįstas netiesinių mažiausiųjų kvadratų metodu [TMH+00].

Prieš atliekant galutinio taškų debesies išgavimo žingsnį yra randamas absoliutusis (tikrasis) objekto mastelis, sulyginant realias koordinates ir atstumus tarp įvairių taškų su gautais praeituose žingsniuose. Tam gali būti naudojamas geografinės registracijos (angl. geo-registration arba geo-referencing) algoritmas [ZHK+14] kuris apskaičiuoja nuotraukų pasukimo ir perkėlimo matricas tarp turimo taškų debesies bei tikrosios koordinačių sistemos ir jomis transformuoja taškų debesį. Taip pat galima mastelio suvienodinimą atlikti rankiniu būdu, pamatuojant atstumus tarp vienos ar kelių porų taškų ant realaus objekto, ir pagal tai išsivedant visą 3D modelio mastelį. Tačiau naudojant tokį metodą labai lengvai gali atsirasti klaidų, taip pat jis ganėtinai lėtesnis.

Galiausiai atliekama tanki 3D rekonstrukcija, papildant turimą retą taškų debesį su informacija iš nuotraukų su pataisytais kameros vidiniais ir išoriniais parametrais. Tam yra naudojami PMVS [FP10] arba SGM [Hir05] metodai, bendrai paėmus, autoriai rado, kad SMG algoritmas greitesnis,

o PMVS yra tikslesnis ir stabilesnis.

Ma ir Liu straipsnyje taip pat išskiriami žingsnių pokyčiai stereofoninėms kameroms (galima pamatyti 1 paveikslėlyje) ir vaizdo įrašams, tačiau šiam darbui tai nėra aktualu.

2.1.2. Taškinių debesų apdorojimo algoritmai

Dažniausiai taškiniai debesys gauti iš aukščiau aprašytų žingsnių ar iš lazerinių skaitytuvu neatitinka modelių pritaikymo reikalavimų, kaip nereikalingų elementų įsiterpimas iš scenos fono. Tad norint atitikti šiuos reikalavimus turimi taškiniai debesys yra apdorojami atliekant dar keturis žingsnius: taškinio debesies išankstinis apdorojimas (angl. preprocessing), tinklelio rekonstrukcija (angl. mesh reconstruction), taškinio debesies segmentacija ir galiausiai jo modeliavimas. Tam naudojamus algoritmus galima rasti 1 lentelėje.

1 lentelė. Ma ir Liu straipsnyje [ML18] apžvelgti taškinių debesų apdorojimo metodai ir algoritmai.

Žingsnis	Požingsniai	Algoritmai ir metodai
Taškinio debesies išankstinis apdorojimas	registracija	ICP (angl. Iterative closest point) [CSS+02]
	triukšmo filtravimas	Rankinis taškų pašalinimas
	išskirčių pašalinimas	RANSAC [FB81]
	raiškos sumažinimas (angl. downsampling)	Taškų atstumo strategija (angl. Point spacing strategy) [SKK15]
Tinklelio rekonstrukcija	-	PSR (angl. Poisson surface reconstruction) [Hop08]
Taškinio debesies segmentacija	-	Regiono auginimas (angl. Region growing) K-vidurkių klasterizavimas (angl. K-means clustering) Vokseliais (angl. voxel) pagrįsti algoritmai Hough transformacija RANSAC [FB81]
Taškinio debesies modeliavimas	-	Objektų matmenų išgavimas [PHL+17]

2.2. Generatyviniai besivaržantys tinklai

Generatyviniai besivaržantys neuroniniai tinklai, toliau GAN (angl. Generative Adversarial Networks), buvo pirmą kartą pasiūlyti 2014 metais Ian J. Goodfellow ir kt. straipsnyje tuo pačiu pavadinimu [GPM+14]. Naudojant šį metodą vienu metu yra mokomi du dirbtinių neuroninių tinklų modeliai, kurie tarpusavyje konkuruoja. Pirmasis modelis yra generatyvinis ir bando pagal turimus mokymo duomenis sukurti tokią išvestį, kuri apgautų kitą, diskriminuojantį modelį, kuris gavęs generatyvinio modelio rezultatus, turi atskirti, ar jie priklauso realiems duomenims, ar jie yra sugeneruoti. Taip gaunamas modelių tarpusavio nulinės sumos žaidimas, kuriame vieno modelio sėkmė yra kito modelio nesėkmė, t.y. jeigu generatyvinio modelio rezultatai pagerėja ir jis dažniau apgauna diskriminuojantį modelį, tai reiškia, kad diskriminuojančio modelio rezultatai proporcingai suprastėja (ir atvirkščiai), tad bendrai abiejų modelių rezultatų suma išlieka nulinė.

Paprasčiausias šio metodo pritaikymas gaunamas kai abu tinklai yra daugiasluoksniai perceptronai. Jie gali būti išreiškiami kaip $G(z;\theta_g)$ ir $D(x;\theta_d)$. G – tai generatyvinio modelio funkcija,

kuri iš atsitiktinio triukšmo vektoriaus z generuoja naują informaciją naudojant parametrus θ_{g}. D – tai diskriminuojančio modelio funkcija išvedanti vieną skaitinę reikšmę, kuri nusako tikimybę, kad duomenų rinkinio elementas x yra dalis originalių duomenų, o ne G modelio sugeneruotų duomenų, naudojant parametrus θ_{d}.

Modelių apmokymo metu yra atliekamas savotiškas minimax žaidimas, kuris naudojamas ir kituose nulinės sumos žaidimuose kaip šachmatai. Jo eigoje modelis D yra mokomas, kad maksimizuoti tikimybę teisingai nuspėti ar duomenys yra realūs ar sugeneruoti, o tuo pačiu modelis G yra mokomas minimizuoti tą pačią tikimybę. Tai išreiškiama su šia netekties funkcija (angl. loss function), kurioje $V(D, G)$ yra žaidimo vertės funkcija (angl. value funtion):

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathrm{x}}[logD(x)] + \mathbb{E}_{\mathrm{z}}[log(1 - D(G(z)))] \tag{3}$$

Autoriai rado, kad praktikoje šis žaidimas turi būti įgyvendinamas pažingsniui, kadangi vienas ar kitas modelis gali tapti permokytas. Tam siūloma atlikti kelis D apmokymo žingsnius prieš atliekant vieną G mokymo žingsnį. Kita problema yra prasti gradientai G mokymo pradžioje, kadangi D labai lengvai atskiria sugeneruotus duomenis, kad to išvengti galima vietoje $log(1 - D(G(z)))$ minimizavimo, maksimizuoti $D(G(z))$, taip išreiškiamas tas pats tikslas, tačiau gaunami daug reikšmingesni gradientai.

Šio žaidimo globalus optimumas gaunamas, kai diskriminuojančio modelio tikimybė pasirinkti sugeneruotus duomenis ir pasirinkti originalius duomenis tampa vienoda. Tai reiškia, kad algoritmas nebesugeba atskirti sugeneruotų duomenų nuo realių, t.y tikimybių pasiskirstymai lygūs – $p_{\mathrm{g}} = p_{\mathrm{data}}$ ir $D(x) = \frac{1}{2}$.

GAN tinklai buvo pritaikyti daugelyje skirtingų sričių, tačiau daugiausiai sekmės susilaukė vaizdinės medžiagos apdorojimo ir kūrimo srityje – [ZXL+17], [IZZ+17] ir [LTH+17]. Po pirmojo Goodfellow GAN paskelbimo taip pat buvo pasiūlyta daugelis patobulinimų. Keli jų aprašyti 2016 metų T. Salimans ir kt. straipsnyje [SGZ+16]. Jame yra išsikeliama GAN konvergavimo problema. Šie tinklai paprastai naudoja gradientinio nusileidimo metodus, ieškančius žemos netekties funkcijos reikšmės, tačiau šie metodai gali nesugebėti konverguoti (suartėti). Tai atsitinka kai vienas iš modelių pradeda dominuoti ir abiejų modelių rezultatai nustoja tobulėti.

Salimans straipsnyje [SGZ+16] yra pasiūlomi penki konvergavimo skatinimo būdai:

1. Ypatybių prilyginimas (angl. Feature matching) – bandoma išspręsti GAN nestabilumą ir sustabdyti generatyvinio modelio persimokymą (angl. overfitting) duodant jam naują tikslą. Vietoje tiesioginio diskriminatoriaus išvesties tobulinimo, generatorius bando sukurti duomenis, kurie atitinka tam tikras realių duomenų statistikas ar ypatybes, o diskriminatorius naudojamas tiktai tam, kad pateikti statistikas, kurias atrodo aktualu prilyginti.

2. Mažų partijų skirstymas (angl. minibatch discrimination) – viena dažniausiai pasitaikančių generatoriaus problemų yra generatyvinio modelio užstrigimas prie vieno taško, kuriame jis sukuria vis tą pačią išvestį. Tai atsitinka, nes diskriminuojantis (skirstantis) tinklas vertina kiekvieną duomenų elementą atskirai ir neduoda generatoriui jokios priežasties didinti išvesties įvairumo. T.Salimans ir kt. siūlo tai spręsti pritaikant mažų partijų diskriminaciją –

diskriminatoriui yra duodamos mažos partijos duomenų egzempliorių, kurie yra vertinami kartu. Modelio išvestis vis tiek privalo išlikti viena skaitinė reikšmė, tad vertinimo metu yra klasifikuojamas vienas duomuo, tačiau kiti duomenys yra naudojami kaip pagalbinė informacija.

3. Istorinis vidurkis – prie GAN modelio netekties funkcijos atitinkamai pridedami praeituose žingsniuose naudotų parametrų θ_g ir θ_d vidurkiai. Taip bandoma išvengti gradiento orbitų (ciklų) aplink optimalų tašką.

4. Vienpusis žymių išlyginimas (angl. one-sided label smoothing) – naudojant šią metodiką, standartiniai klasifikavimo taikiniai, tikimybės 0 arba 1, yra pakeičiami labiau išlygintomis reikšmėmis kaip 0,1 ir 0,9. Taip yra padidinamas diskriminatoriaus atsparumas generatoriaus apgaudinėjimo bandymams ir padeda išvengti generatoriaus dominavimo.

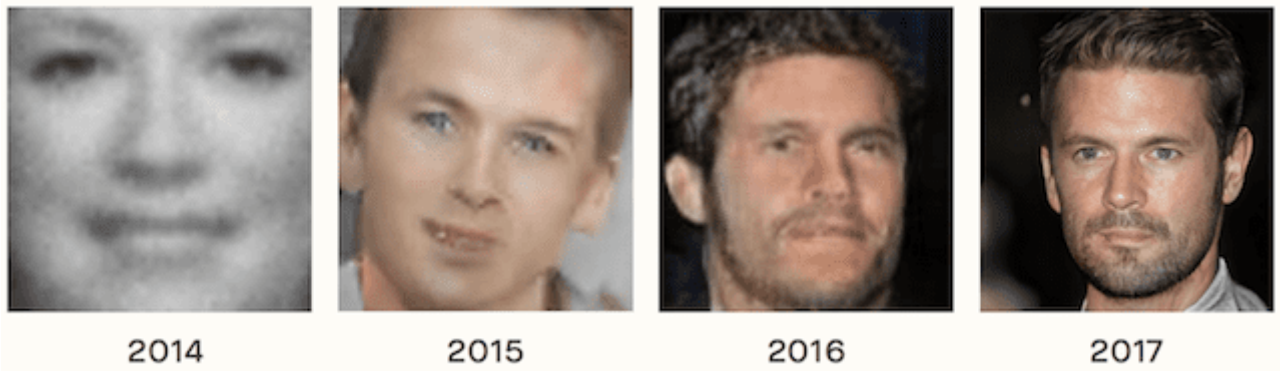
5. Virtualus partijų normalizavimas – partijų normalizacija (angl. batch normalization) yra pritaikoma daugelyje giliojo mokymosi įgyvendinimų, mokymo stabilizavimui ir pagreitinimui. Jos metu modelio išvestys mokantis su atskiromis duomenų partijomis yra standartizuojamos taip, kad jų vidurkis būtų lygus 0, o standartinis nuokrypis 1. Tačiau tai reiškia, kad išvestys pagal vieną duomenų egzempliorių tampa priklausomos nuo kitų egzempliorių. Kad to išvengti, bet vis tiek išnaudoti normalizacijos naudą, autoriai pasiūlė virtualų partijų normalizavimą. Šiuo metodu kiekviena įvestis yra normalizuojama pagal pavyzdinę partiją duomenų, kuri yra nustatoma prieš pradedant mokymo procesą.

Sekančią partiją patobulinimų 2018 metais pasiūlė T. Kerras ir kt. [KAL+18]. Nors jų parašytame straipsnyje siūlomi keli metodai modelių konkurencijos pagerinimui, pagrindinis pasiekimas buvo progresyvaus GAN auginimo metodas. Pagal jį, abu modeliai pradžioje yra mokomi su mažesnės rezoliucijos duomenimis, o mokymo eigoje ši rezoliucija yra progresyviai didinama. Tai ne tik sutaupo laiko (autoriai rašo, kad tai 2-6 kartus pagreitino modelių mokymą), bet ir padidina GAN stabilumą ir leidžia pasiekti daug geresnius rezultatus.

Šis laipsniškas rezoliucijos didinimas leidžia modeliui pirmiausiai atrasti bendrąsias duomenų savybes ir tada pamažu perkelti savo dėmesį link smulkesnių detalių, o ne bandyti viską išmokti vienu metu. Generatyvinis ir diskriminuojantis modeliai yra auginami vienodai ir visi sluoksniai yra prieinami viso mokymosi metu. Kai nauji sluoksniai yra pridedami, jie įvedami pamažu – naujam, padvigubintos rezoliucijos sluoksniui, yra pridedamas svoris, išreiškiamas kaip rezoliucijos daugiklis nuo 0,5 iki 1, pagal kurį sluoksnio rezoliucija yra sumažinama perpus ir mokymo metu tiesiškai didinama iki pilnos. Taip yra bandoma išvengti staigių pokyčių mokymo procese, kurie galėtų neigiamai jį paveikti.

Taip pat tame pačiame straipsnyje buvo pasiūlytas objektyviai įvertinamas modelio kokybės rodiklis vadinamas supjaustytu Wasserstein atstumu arba SWD (angl. Sliced Wasserstein Distance). Šis rodiklis nurodo duomenų tarpusavio skirtumą, tad vertinant generatyvinius modelius tikslas yra gauti kuo mažesnę šio rodiklio reikšmę lyginant sugeneruotus duomenis ir realius duomenis. Tai leidžia daug lengviau įvertinti gautus generatyvinius modelius, kadangi nereikia pasikliauti žmogiškuoju vertinimu, kuris yra nepastovus ir sunkiai įgyvendinamas dideliu mastu.

Visi šie pasiūlymai privedė prie labai greito GAN tinklų tobulėjimo. Akivaizdžiausias to pavyzdys buvo veidų nuotraukų generavimo srityje, ką galima matyti 4 paveikslėlyje. Tačiau tas pats progresas gali būti pritaikomas ir kitose GAN taikymuose.



4 pav. GAN tinklų tobulėjimas veidų vaizdų kūrimo srityje [BAC+18]

2.3. 3D GAN

Trimačių modelių skaitmeninimas, pritaikant GAN algoritmus fotogrametrijos mokslui, nėra nagrinėjamas pirmą kartą. Keli metodai buvo iškelti kasdienių objektų, ar kultūrinio paveldo skaitmeninimui, turint ribotą kiekį informacijos. Darbo įkvėpimas ir pagrindiniai literatūros apžvalgos šaltiniai buvo 2016 metais JiaJun Wu ir kt. parašytas straipsnis apie objektų trimatį skaitmeninimą iš vienos dvimatės nuotraukos [WZX+16] bei 2019 metais V.V. Kniaz ir kt. parašytas straipsnis apie bendresnį GAN pritaikymą architektūrinio paveldo srityje [KRK19].

2.3.1. Modelio skaitmeninimas absoliučioje pozicijoje

Dažniausias 3D skaiteminimo GAN tinklais tipas yra modelio skaitmeninimas absoliučioje pozicijoje. Gavus nuotrauką generuojantis tinklas visus modelius bando kurti tokiu pačiu pasukimu koordinačių erdvėje. Taip gaunamas daug lengviau panaudojmas ir vertinamas modelis, tačiau gali kilti išukių su skirtingais nuotraukų kampais, tad yra pravartu turėti mokymo duomenis su nuotraukomis atvaizduojančiomis modelius iš daugelio skirtingų kampų.

JiaJun Wu ir kt. straipsnyje aprašomas toks trimačių objektų generavimas pritaikant tūrinės konvoliucijos neuroninius tinklus (angl. volumetric convolution) ir GAN. Pagrindinis autorių tikslas buvo sukurti metodą generuojantį 3D modelius, kurie yra panašūs į jau turimus realių objektų trimačius modelius tokiu pačiu principu, kaip ir anksčiau minėti dvimačiai GAN metodai kūrė naujus vaizdus pagal realias nuotraukas. Tačiau metodas taip pat buvo pritaikytas ir modelių generavimui atsižvelgiant į jų dvimačius atvaizdus. Tai padaryta jį sujungiant su variaciniu automatiniu koduotoju arba VAE (angl. Variational AutoEncoder) [KW13], [LSL+15].

Siūlomas 3D modelių generatorius paima 200-ų matmenų triukšmo vektorių, pripildytą atsitiktinėmis reikšmėmis iš intervalo [0, 1] ir atvaizduoja jį į $64 \times 64 \times 64$ kubą. Tam naudojamas konvoliucinis neuroninis tinklas su penkiais $4 \times 4 \times 4$ konvoliuciniais sluoksniais tiek generatyviniame, tiek diskriminuojančiame tinkle.

Kuriant modelius iš dvimačių nuotraukų pakeičiamas tiktai pradinis triukšmo vektorius. Vietoje atsitiktinių reikšmių yra naudojamos reikšmės aprašančios nuotraukos savybes. Šios reikšmės gaunamos naudojant variacinį automatinį koduotoją (VAE) – tai dirbtinis neuroninis tinklas, kuris paverčia jam duotą įvesties kintamąjį į jį aprašantį daugiamatį statistinį reikšmių išdėstymą (vektorių).

Straipsnyje naudojamas VAE tinklas sudarytas iš penkių erdvinių konvoliucinių sluoksnių, kurių dydžiai {11, 5, 5, 5, 8}, o žingsnių dydžiai (angl. Stride), nustatantys kaip algoritmas keliauja per paduodamus vaizdus, atitinkamai – {4, 2, 2, 2, 1}. Neskaitant partijų normalizacijos ir ReLU sluoksnių, kurie pritaikomi visuose metodo neuroniuose tinkluose.

Netekties funkcija šiam metodui susideda iš trijų dalių: rekonstrukcijos netektis, t.y. Euklido atstumo tarp sugeneruoto 3D modelio ir žinomo teisingo modelio – L_{recon}, jau minėtos GAN netekties – $L_{\text{3D-GAN}}$, ir Kullback-Leiber nuokrypio (angl. KL divergence) – L_{KL}. KL nuokrypis – tai skirtumas tarp dviejų tikimybių pasiskirtymų, šiuo atveju bandoma nenutolti nuo variacinio pasiskirstymo gauto su pradiniu vektoriumi $p(z)$, kuris šiame darbe buvo išreikšas kaip daugiavariacinis Gauso pasiskirtymas su nuliniais vidurkiais ir vienetine dispersija. Funkcija matoma formulėje (4) (α_1 ir α_2 KL nuokrypio ir rekonstrukcijos netekties svoriai):

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}} \tag{4}$$

Metodo įvertinimui buvo naudojamas IKEA baldų trimačių modelių rinkinys, kuris yra gana sudėtingas, kadangi nuotraukos yra paimtos natūralioje erdvėje, todėl dažnai dalis objekto būna uždengta kitų objektų [LPT13]. Pritaikius GAN tinklus, gautas iki 13 % vidutinis patobulėjimas anksčiau geriausiems gautiems dirbtinio intelekto rezultatams (žr. 2 lentelę ir 5 pav.).

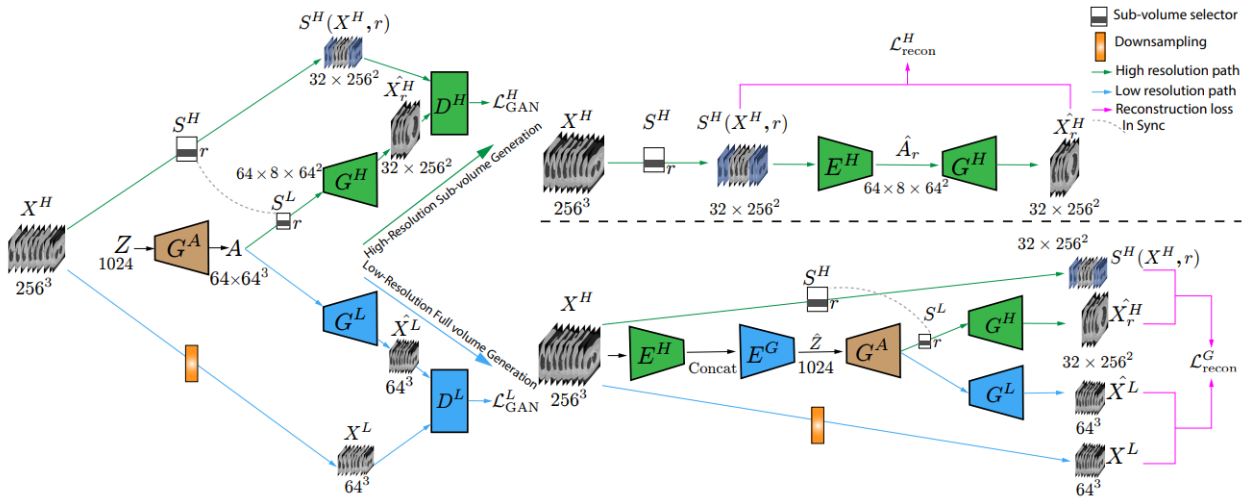
2 lentelė. Wu ir kt. straipsnyje [WZX+16] gauti 3D modeliavimo rezultatai (Tikslumas procentais).

Metodas	Lova	Knygų lentyna	Kėdė	Darbo stalas	Sofa	Stalas	Vidurkis
AlexNet-fc8 [GFR+16]	29,5	17,3	20,4	19,7	38,8	16,0	23,6
AlexNet-conv4 [GFR+16]	38,2	26,6	31,4	26,6	69,3	19,1	35,2
T-L Network [GFR+16]	56,3	30,2	32,9	25,8	71,7	23,3	40,0
3D-VAE-GAN (geriausias rezultatas)	63,2	46,3	47,2	40,7	79,8	42,3	53,1

5 pav. Wu ir kt. straipsnyje [WZX+16] gauti 3D modeliavimo rezultatai.

Vienas naujausių tokio tipo 3D GAN metodų yra 2022 metų Li Sun ir kt. straipsnyje pasiūlytas HA-GAN [SCX+22]. Autoriai rašo, kad esminis iššūkis 3D modelių kūrimui su GAN tinkais yra

tam reikalingas atminties kiekis. Dirbtinių neuroninių tinklų, generuojančių $256 \times 256 \times 256$ ir didesnės raiškos objektus, gradientai nebetelpa net į modernių vaizdo plokščių atmintį. Tad norint kovoti su šia problema, Sun ir kt. siūlo padalinti GAN tinklą į dvi šakas – viena šaka yra apmokoma su žemos raiškos $64 \times 64 \times 64$ objektais, o kita šaka apmokoma su atsitiktinai pasirinktomis aukštos raiškos modelių dalinėmis iškarpomis, kurios sudaro $\frac{1}{8}$ viso modelio dydžio ($32 \times 256 \times 256$). Abi šakos yra sujungtos bendru, trečiu, generatyviniu tinklu, kuris leidžia panaudoti bendrą objekto informaciją generuojant aukštos raiškos 3D modelius. Bendrą modelio diagramą galima pamatyti 6 paveikslėlyje.



6 pav. Kairėje – HA-GAN architektūra be koduotojo, mokymas yra padalinamas į dvi šakas, vienoje tinklas apmokomas su pilnais mažos raiškos duomenimis, o kitoje modelis apmokomas su daliniais aukštos raiškos duomenimis. Dešinėje – Hierarchinis koduotojas apmokomas su dvejais skirtingais rekonstrukcijos nuostoliais, pritaikytais abiejų tipų duomenims [SCX+22].

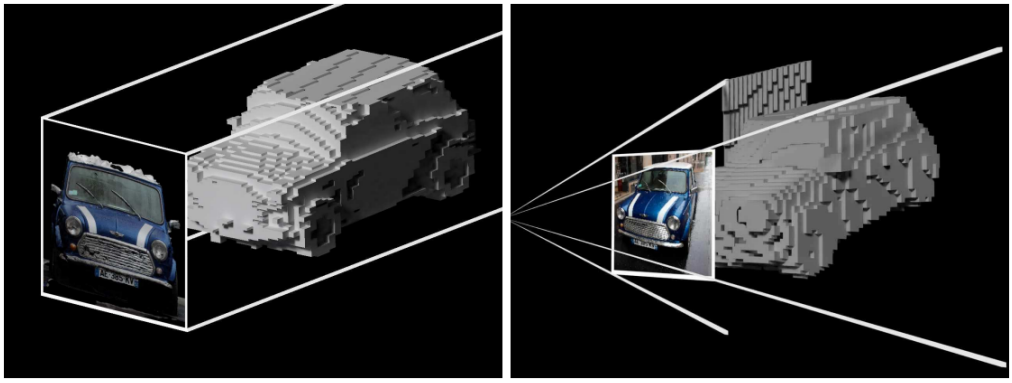
Sun ir kt. straipsnyje [SCX+22] minimi ir kiti metodai, panašiai dalinantys trimačius modelius į atskiras dalis, tačiau juose generuojamo modelio dalys yra kuriamos atskirai, tad pastebimi artefaktai (klaidos) tarp sugeneruotų dalių sujungimų. HA-GAN metode tuo tarpu, kadangi gradientai yra reikalingi tiktai mokymo metu, tinklų struktūra yra pakoreguojama tarp mokymo ir generavimų etapų taip, kad būtų kuriami pilni aukštos raiškos modeliai.

Taip pat metode yra pridedamas tokiu pačiu principu apmokomas koduotojas (angl. encoder), tačiau kitaip nei Wu ir kt. straipsnyje, koduotojo mokymui naudoti 3D modeliai ir jų dalys, siekiant stabilizuoti bendra GAN metodo mokymosi procesą. Norint pritaikyti HA-GAN rezultatus prie šio magistro baigiamojo darbo tikslo, autorių siūlomas koduotojas turėtų būti pakeistas kitu, dvimates nuotraukas apibūdinančiu koduotoju.

2.3.2. Su nuotrauka sulygintų modelių generavimas

V. V. Kniaz ir kt. savo 2018 straipsnyje [KKR18] pateikė šiek tiek kitokią modelių skaitmeninimo GAN tinklais strategiją. Jų siūlomas Z-GAN metodas paremtas vokselinio modelio pjūviais, sulygintais su nuotraukos (kameros) parametrais.

Vokselis – tai tūrinis pikselis, atvaizduojantis informaciją tam tikroje trimatės erdvės koordi-



7 pav. Palyginimas tarp vokselinio (kairėje) ir frukselinio (dešinėje) modelio [KKR18].

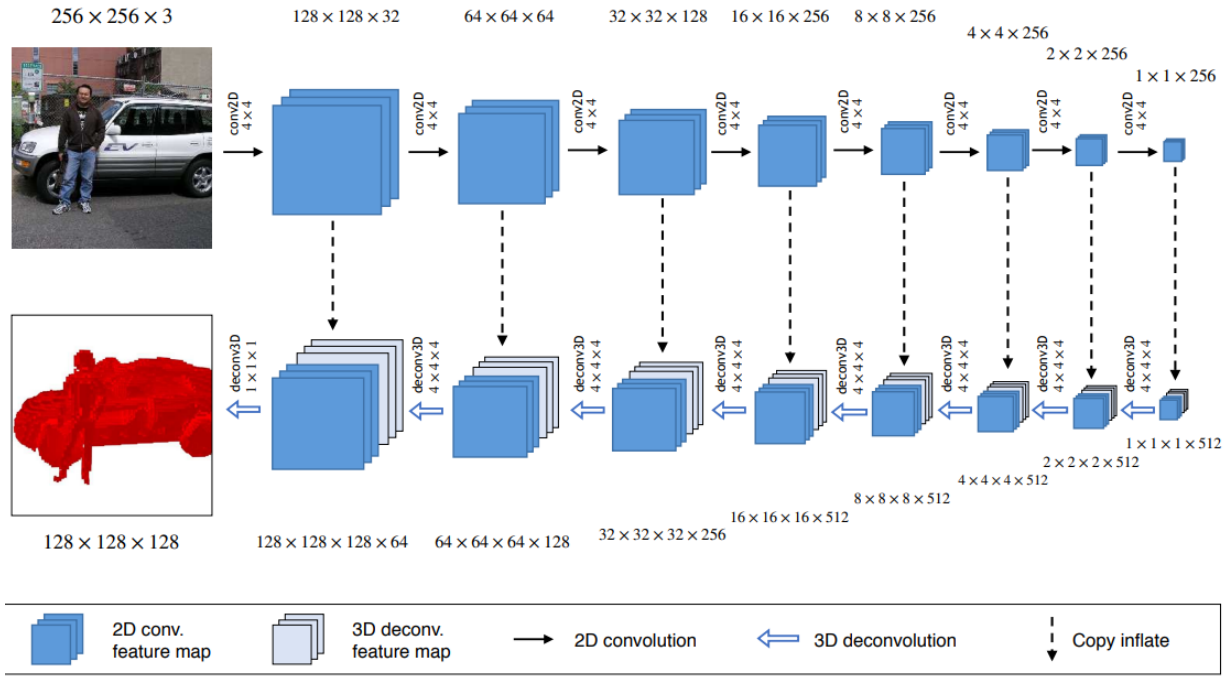
natėje. Vokselinio modelio pjūvis – tai vienetinio gylio 3D modelio dalis gauta supjausčius modelį pagal vieną ašį. Autorių hipotezė buvo, kad atsižvelgus į pjūvių sutapimus su tam tikromis fotografuojamo objekto silueto dalimis galima patobulinti 3D skaitmeninimo GAN tinklais rezultatus. Tam reikia sulygiuoti turimas nuotraukas su viena iš mokymosi 3D modelių ašių. Kad šis lygiavimas būtų tikslus, fotografuojama erdvė yra įsivaizduojama kaip nupjauta piramidė (angl. frustum), kurioje labiau nuo kameros pozicijos atitolę vokselių suoklsniai yra praporcingai didesni, taip labiau atitinkant kameros veikimą (žr. 7 pav.). Tokį nupjautos piramidės pavaidavimo sujungimą su vokseliniu modeliu autoriai vadina frukseliniu modeliu (angl. fruxel model).

Pagrindinė frukselinės modelio reprezentacijos mintis yra sutapatinti nuotraukose turimus kontūrus su modelio pjūviais ir taip generatorių išmokyti kurti modelius kurie labiau atitinka fotografuojamus objektus. Tokį modelį galima aprašyti naudojant keturių parametrų aibę: $\{z_\mathrm{n}, z_\mathrm{f}, d, \alpha\}$, kur z_n yra kažkokia pradinė (artima) plokšuma, z_f galinė (tolima) plokštuma (z_f ir z_n skirtumas nusako modelio gylį), d – pjūvių skaičius, o α kameros matymo laukas (angl. field of view). Tokį frukselinį modelį galima konvertuoti į vokselinį padauginant kiekvieno pjūvio mastelį iš koeficiento k, matomo formulėje (5).

$$k = \frac{z_\mathrm{n}}{z_\mathrm{n} + s_\mathrm{z}}; s_\mathrm{z} = \frac{z_\mathrm{f} - z_\mathrm{n}}{d} \tag{5}$$

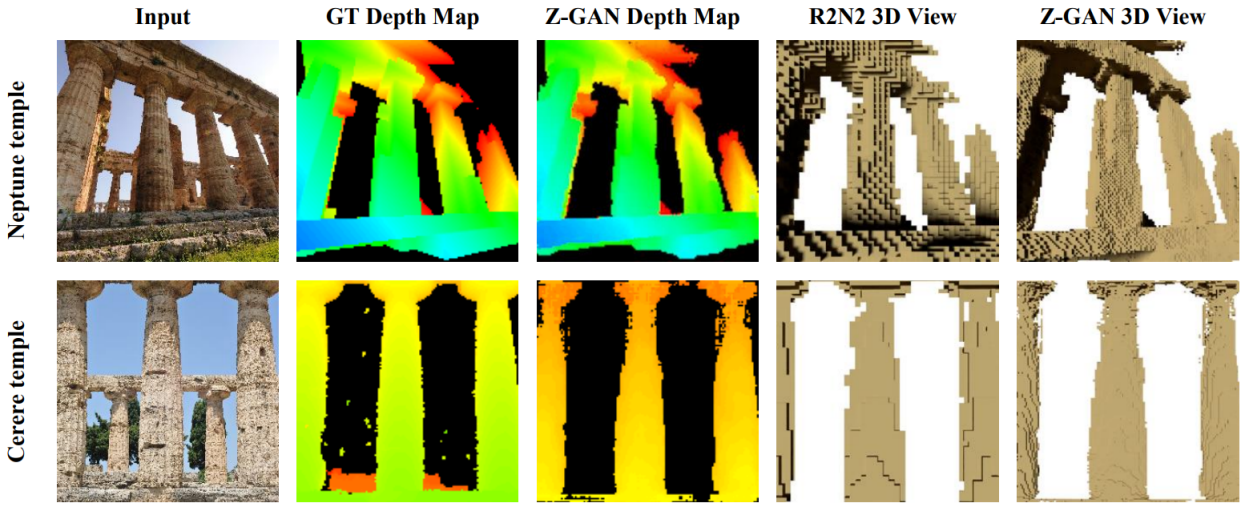
Z-GAN generatorius paima U-Net tinklą [RFB15], kuris originalai buvo sukurtas nuotraukų segmentavimui, kaip pagrindą. Autoriai pakeičia dvimačius dekonvoliucijos sluoksnius į trimačius ir pakoreguoja peršokimo sujungimus (angl. skip connections) taip, kad išlaikyti suderinamumą tarp 2D konvoliucijos ir 3D dekonvoliucijos sluoksniu(žr. 8 pav.). Tuo tarpu diskriminuojantis modelis stebi tiktai atskirus pjūvius, o ne visą modelį, tad generatorius yra skatinamas juos teisingai atkurti.

Kniaz ir kt. kitame, 2019 metais parašytame, straipsnyje [KRK19] aprašomas Z-GAN skaitmeninimo pritaikymas kultūrinio paveldo scenų rekonstrukcijai. Autoriai išsikelia problemą, kad nors modernūs tradicinės rekonstrukcijos metodai pasiekia įspūdingus rezultatus, dažnai yra turimos (išlikusios) tiktai viena ar kelios retos scenos ar objekto nuotraukos, tad yra reikalingas metodas, kuris galėtų gauti bent apytikslius 3D modelius su tokia ribota informacija. Tam taip pat buvo sukurtas naujas duomenų rinkinys sudarytas iš Neptūno šventyklos, Cerere šventyklos (Pestas, Italija) ir Bostra archeologinės zonos (Sirija) 3D modelių ir nuotraukų.

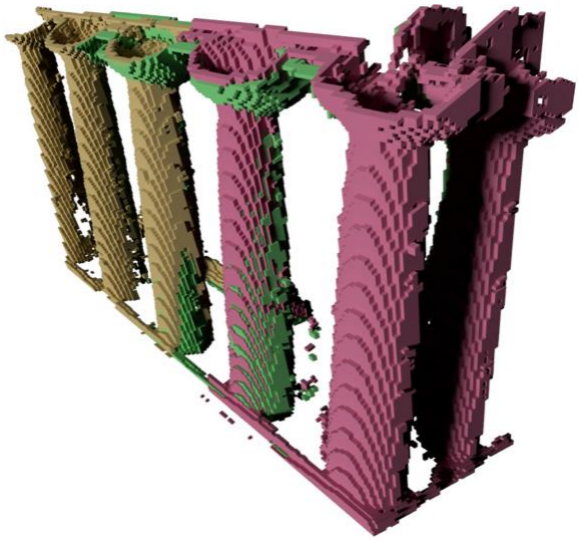


8 pav. Z-GAN generatyvinio modelio architektūra [KKR18].

Pritaikius metodą kultūrinio ir architektūrinio paveldo nuotraukose esančių objektų skaitmeninimui (žr. 9 pav.) gauti aukštesnės raiškos rezultatai nei kituose darbuose ($128 \times 128 \times 128$ vietoje $64 \times 64 \times 64$), buvo galima atkurti mokymo metu nematytus objektus (pvz. arkas kurių nebuvo mokymo duomenų rinkinyje) bei buvo įgyvendinta galimybė panaudoti kelias nuotraukas vieno trimačio modelio kūrimui, taip gaunant geresnius rezultatus (žr. 10 pav).



9 pav. Kniaz ir kt. straipsnyje [KRK19] gauti 3D modeliavimo rezultatai.



10 pav. Kniaz ir kt. straipsnyje [KRK19] gautas Cerere šventyklos 3D modelis sugeneruotas iš trijų gretimų nuotraukų.

Kelių nuotraukų panaudojimas yra pasiekiamas sujungiant kelis su viena nuotrauka gautus frukselinius modelius naudojant ICP algoritmą [CSS+02], kuris padeda suvaldyti skirtumus tarp kameros išorinių parametrų ir nuotraukų persiliejimą (angl. overlap).

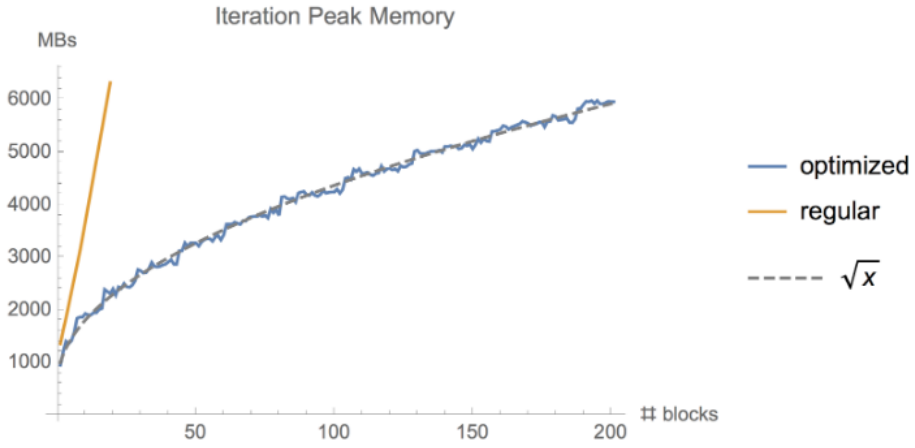
2.4. 3D modelių kokybės tobulinimas

Vienas didžiausių 3D GAN metodų trūkumų yra jų gautų rezultatų vizualus nepatrauklumas. Sugeneruoti 3D modeliai yra labai kampuoti ir mažos raiškos (rezoliucijos). Vienas to sprendimo būdas būtų dirbtinis raiškos padidinimas (angl. upscaling), tačiau 3D modelių raiškos didinimo metodai dar nėra plačiai ištyrinėti, daugiausiai pastangų yra kreipiama i 2D vaizdo raiškos didinimą, kaip NVIDIA DLSS (angl. Deep Learning Super Sampling), o esamos žinios ar straipsniai dažnai nėra viešai prieinami.

Kaip ir buvo aptarta anksčiau, iš karto generuojant aukštos raiškos modelius atsiranda atminties apribojimai, kadangi vokselių skaičius modeliuose auga kubine progresija. Apart Li Sun ir kt. straipsnyje pasiūlytų patobulinimų [SCX+22], taip pat yra paplitęs 2016-aisiais metais T. Chen ir kt. straipsnyje [CXZ+16] pasiūlytas gradientų atskaitos taškų (angl. gradient checkpoint) metodas, kuris sumažina gradientams reikalingos atminties priklausomybę, nuo tinklo sluoksnių kiekių iš tiesinės $O(n)$ į $O(\sqrt{n})$.

Dribtinių neuroninių tinklų mokymas susideda iš dviejų etapų – sklidimo pirmyn (angl. forward propagation) ir atgalinio sklidimo (angl. backward propagation). Atgalinio sklidimo metu einama pro visus tinklo sluoksnius atgaline eiga ir atnaujinami sluoksniuose laikomi svoriai. Tam yra naudojamos reikšmės, kurios buvo gautos sklidimo pirmyn metu. Dažniausiai, taupant skaičiavimų kiekį, visos pirmojo etapo metu gautos reikšmės yra laikomos atmintyje, ir gali būti ištrinamos tiktai tada, kai su konkrečia reikšme jau atliktos visos antrojo etapo operacijos. Norint sutaupyti kuo daugiau atminties būtų galima kiekvienam atgalinio sklidimo žingsniui iš naujo perskaičiuoti visas reikalingas sklidimo į priekį reikšmes, tačiau tai pakeltų skaičiavimų sudėtingumą iki $O(n^2)$.

Straipsnyje siūloma kas $\sqrt{n}$ žingsnių, kur n yra tinklo sluoksnių skaičius, išsaugoti reikalingas reikšmes ir sklidimo atgal metu iš naujo paskaičiuoti trūkstamas. Taip gaunama minėta atminties $O(\sqrt{n})$ priklausomybė, o skaičiavimų kiekis yra padidinamas tiktai 20 %.



11 pav. Atminties kiekis, naudojamas apmokant ResNet tinklą [HZR+15] su didelėmis duomenų partijomis. Lyginamas paprastas gradientų skaičiavimas "regular" ir gradientų skaičiavimas pritaikant atskaitos taškų saugojimą "optimized". Kreivė $\sqrt{x}$ pavaizduoja paprasto gradientų skaičaivimo metu naudojamos atminties šaknis. Matoma, kad optimizuota atmintis labai artimai seka šią kreivę [Bul].

Kitas būdas padidinti modelių vizualų patrauklumą yra išlyginimas (angl. smoothing), kurio metu 3D modelių nelygumai galėtų būti ištiesinti ir perteklinė informacija panaikinta. Vizualios kokybės tobulinimas gali sumažinti modelio atitikimą konkretiems fotografuotiems objektams, tad turėtų būti taikomas į tai atsižvelgiant, ypač tinkama kai vertinamas ne tikslus objekto atkūrimas, o objekto realistiškumas, kaip fono elementų kūrimas filmuose ar kompiuteriniuose žaidimuose.

2021 metų straipsnyje A. Bacciaglia ir kt. [BCL21] rašo apie paviršių išlyginimą 3D modelių topologijos optimizavimui. Topologijos optimizavimas yra atliekamas norint projektuoti lengvus ir efektyvius komponentus naudojamus pramoninės inžinerijos srityje, kaip lėktuvų ar automobilių dalių kūrime. Autoriai siūlo metodą, kuris nepraranda tiek daug informacijos kiek kiti panašūs metodai, kurie dažnai užapvalina objektų kampus ir sukuria gamybai netinkamą išvestį. Tai gali būti pritaikoma ir 3D GAN metodams, kadangi dažniausiai bendra objekto struktūra yra sukuriama tinkamai, tad reikėtų kuo daugiau jos išlaikyti.

Autorių aprašomo metodo pagrindas yra Laplaso išlyginimas (angl. Laplacian smoothing), kuris per daugelį straipsnių ir iteracijų buvo optimizuojamas ir tobulinamas. Ši išlyginimo metodika keičia 3D tinklelio viršūnių pozicijas pagal lokalią informaciją, pritaikant tokią Laplaso difuzijos lygtį:

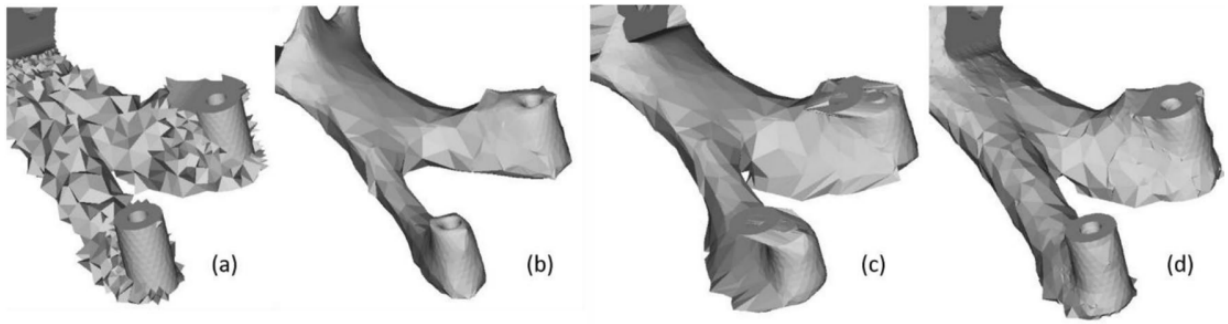
$$\frac{\delta X}{\delta t} = \lambda L(X) \tag{6}$$

Formulėje (6) X yra viršūnių rinkinys (angl. tensor), kuris bus pažingsniui keičiamas išlyginant 3D modelį. L – tai Laplaso funkcija, dažniausiai išreiškiama trimačiams duomenims kaip $\Delta\frac{\delta}{\delta x}i + \frac{\delta}{\delta y}j + \frac{\delta}{\delta z}k$, skaičiuojanti gradiento nuokrypį pagal koordinačių x, y ir z dalines išvestines. λ –

tai svoris nuo 0 iki 1 nusakantis difuzijos greitį, o δt nusako modelio paviršiaus tinklelio kitimą algoritmo eigoje.

Straipsnyje išvardinami keli patobulinimai paprastam Laplaso išlyginimui (gautų išlyginimo rezultatų palyginimą su kitu, tuo metu geriausiu laikomu, išlyginimo algoritmu galima matyti 12 paveikslėlyje):

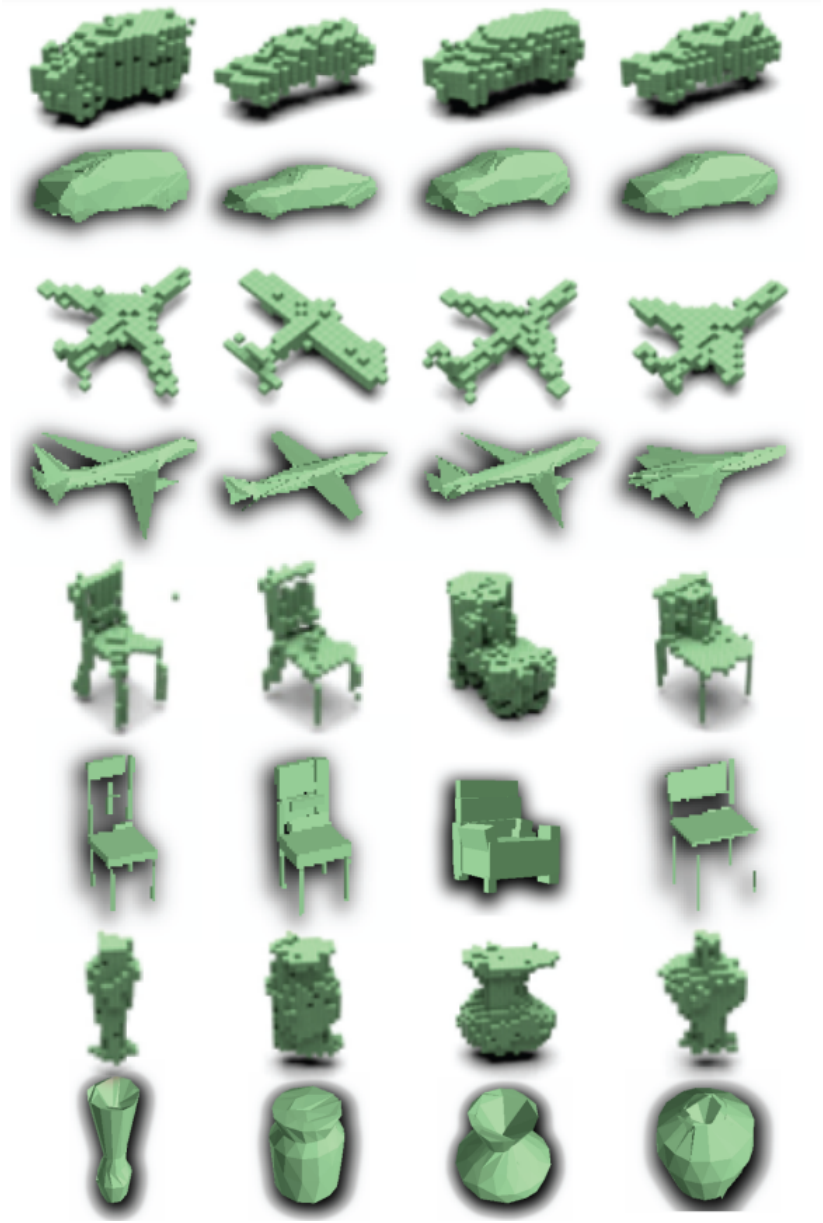
- Atsižvelgimas į mastelį, neleidžiant keisti tarp tinklelio susidariusių trikampių dydžio ir taip išvengiant per didelio viršūnių išbarstymo.
- Atbulinio stūmimo pridėjimas, po kiekvienos iteracijos dalinai grąžinant viršūnes link jų praeitos pozicijos pagal viršūnės ir jos kaimynų vidutinį nuotolį.
- Lygių paviršių aptikimas – siekiama palikti jau esamas lygias (tiesias) modelio dalis nepaliestas, taip išvengiant nereikalingo kampų apvalinimo
- Skylių aptikimas – pritaikoma, kai modelyje yra tikimasi skilių ir jų savybių nereikia pakeisti
- Tūrio išlaikymas, kiekvienoje iteracijoje lyginant gautą modelio tūrį su pradiniu ir atitinkamai padauginant viršūnių koordinates, kad išvengti objekto tūrio sumažėjimo.



12 pav. Bacciaglia ir kt. straipsnyje [BCL21] gauti išlyginimo rezultatai: (a) originalus modelis, (b) Taubino algoritmu gautas modelis [Tau99], (c) Taubino algoritmas neišlyginant tiesių paviršių, (d) Bacciaglia ir kt. algoritmas – optimizuotas HC-SDU

Laplaso išlyginimas, kaip ir buvo minėta, yra pritaikomas turint trimatį taškų tinklelį (angl. 3D mesh), kurio tarpus užpildžius gaunamas tuščiaviduris 3D modelis. Tačiau su apžvelgtais 3D-GAN algoritmais yra gaunamas vokselinis modelis sudarytas iš daug mažų kubelių. Tad norint pritaikyti tokį išlyginimo algoritmą reikia gauti modelius atvaizduotus tinkleliu. Tai gali būti atliekama paprasčiausiai nurodant išorinius modelio vokselius kaip tinklelio viršūnes arba galima pritaikyti 2019 metais P. Handerson ir V. Ferrari pasiūlytą 3D GAN algoritmą [HF19], kuris iš karto generuoja 3D tinkleliu atvaizduojamus modelius vietoje vokselinių modelių.

Šiame algoritme taip pat kuriami 3D modeliai iš vienos 2D nuotraukos, tačiau yra įvedami 3D tinklelio parametrai ir papildomai analizuojama fotografuotų objektų šešėlių bei atspalvių informacija (angl. shading), kuri padeda geriau suprasti daiktų išlinkimus. Kaip matoma 13 paveikslėlyje, vien pritaikius šį metodą jau gaunami daug patrauklesni rezultatai, ypač išlenktiems objektams, tačiau juos galima toliau tobulinti pritaikant išlyginimą.



13 pav. Nelyginėse eilutėse – vokseliais grįstu 3D GAN algoritmu gauti rezultatai [GWM18]. Lyginėse eilutėse – tinkleliu grįsto 3D GAN rezultatai [HF19]

3. Metodologija

Šiame skyriuje aprašomi magistro baigiamojo darbo metu realizuoti metodai. Nurodoma kokia kompiuterinė bei programinė įranga buvo naudojama, kokie mokymo duomenys buvo pasirinkti bei kokie žingsniai buvo atlikti įgyvendinant išsikeltus darbo uždavinius.

3.1. Naudojama programinė ir kompiuterinė įranga

Visiems magistro baigiamojo darbo metu kuriamiems dirbtinio intelekto metodams įgyvendinti buvo pasirinkta naudoti *TensorFlow* [MAP+15]. Tai pilnavertė mašininio mokymosi platforma, kuri palengvina dirbtinių neuroninių tinklų kūrimą, leidžiant visą dėmesį sutelkti į modelio bendrą architektūrą, o ne į konkrečias skirtingų tinklo sluoksnių realizacijas. *TensorFlow* pasirinktas vietoje kitų alternatyvų, kaip *PyTorch* [PGM+19], dėl itin detalios dokumentacijos ir didelio paplitimo, kuris lemia lengvesnį informacijos prieinamumą. Darbų eigoje pastebėta, kad didelė dalis apžvelgtų GAN metodų vis dėlto buvo įgyvendinami su PyTorch, tačiau šių bibliotekų implementacijos yra gana panašios ir konvertacija nebuvo didelė problema.

Tensorflow platforma gali būti naudojama kartu su *Python*, *C++*, *Java* ir *JavaScript* programavimo kalbomis. Daugiausiai informacijos apie neuroninių tinklų kūrimą randama su *Python* ir *C++*. *C++* gali suteikti geresnį programinio kodo našumą, ypač jeigu yra atliekama daug paprastų programavimo operacijų, tačiau darbą pasirinkta atlikti su *Python* kalba, dėl jos paprastesnės sintaksės ir lengvesnio naudojimo. Didžiąją programos vykdymo laiko dalį sudarys tinklų mokymas, kuris atliekamas naudojant *Tensorflow* metodus, kurių pagrindas visais atvejais yra toks pat.

Kuriant gilius neuroninius tinklus, apart programinės įrangos taip pat yra svarbūs ir turimos kompiuterinės įrangos (angl. hardware) pajėgumai. Dirbant su daugiamačiais duomenimis, Svarbiausias šios įrangos atributas yra operatyviosios atminties kiekis. Kuo daugiau atminties turima, ypač vaizdo plokštės atminties, tuo didesni tinklai, duomenų partijos (angl. batch size) ir bendras duomenų kiekis gali būti naudojami giliojo mokymo procese. Šiam darbui naudojamas kompiuteris su tokiais komponentais:

- Procesorius: AMD 7900x – 12 branduolių ir 24 gijos, kurių dažniai 4,7-5,6 GHz.
- Operatyvioji atimintis: 32 GB.
- Vaizdo plokštė: NVIDIA RTX 3090, su 24 GB atminties.

Tai yra itin galingas kompiuteris, tačiau tai ne visada proporcionaliai patobulina programos našumą. Jeigu yra atliekama daug mažų, nuoseklių operacijų, didžiausią laiko dalį užima informacijos perdavimas ir komunikacija tarp procesų, o visos kitos programos dalys turi laukti. Tad norint išnaudoti visus kompiuterio pajėgumus reikia į tai atsižvelgti ir atitinkamai kiek įmanoma optimizuoti programinį kodą, kad pilnai išnaudoti kompiuterinės įrangos pajėgumus.

TensorFlow turi du mašininio mokymo kodo paleidimo rėžimus: nekantrusis vykdymas (angl. eager execution) ir diagrama pagrįstas vykdymas (angl. graph execution). Pirmasis vykdo visas aprašytas operacijas iš eilės, pagal jų eiliškumą kode, o antrasis kodo vykdymo pradžioje sudaro

operacijų diagramą, pagal kurią visos operacijos yra planuojamos iš anksto. Veiksmų suplanavimas iš anksto ženkliai padidina kodo efektyvumą (darbo metu atliekamiems apmokymams pastebėtas nuo 20 iki 30 % pagreitėjimas), tačiau apsunkinamas kodo klaidų radimas ir taisymas, kadangi tikslios kintamųjų reikšmės ne visada žinomos kodo vykdymo metu. Tad abu šie rėžimai turi savo paskirtį ir buvo atitinkamai pasirinkti skirtinguose metodų kūrimo etapuose.

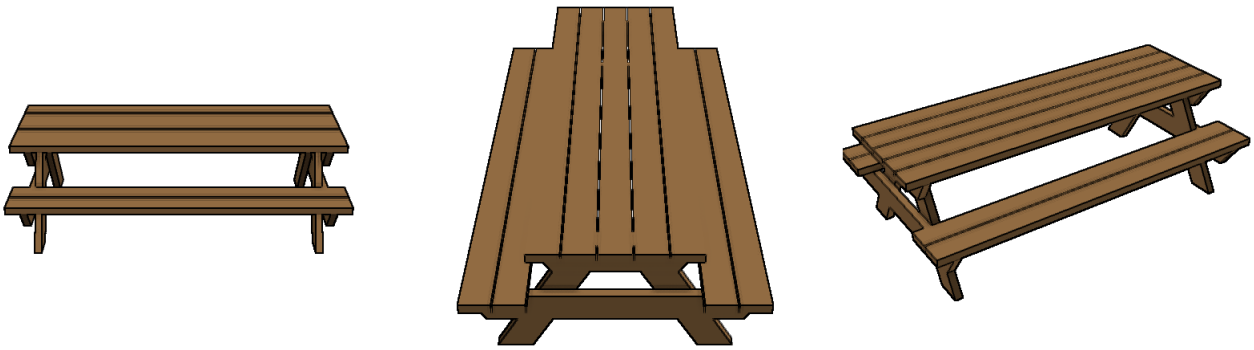
Taip pat, naujausiuose, RTX serijos, NVIDIA vaizdo plokštėse buvo pridėti papildomi matricų daugybą paspartinantys komponentai vadinami tenzorių branduoliais (angl. tensor cores). Paprasti vaizdo plokštės CUDA branduoliai per vieną operatyviojo dažnio taktą gali atlikti tiktai vieną daugybos operaciją, o tenzorių branduoliai yra pritaikyti $4 \times 4 \times 4$ matricų daugybai, tad per tą patį taktą gali atlikti 64 tokias operacijas. Tinkamas tenzorių branduolių panaudojimas gali dramatiškai sutrumpinti neuroninių tinklų apmokymui reikalingą laiką, tačiau šie branduoliai kaip įvestį priima tiktai 16 bitų tikslumo realių skaičių matricas, kurių naudojimas gali sumažinti bendrą mokymo tikslumą. Darbo metu buvo mėginta išnaudoti šiuos branduolius, tačiau susidurta su daug iššūkių dėl rezultatų ir kodo klaidų, tad nuspręsta jo atsisakyti. Plačiau apie tenzorių branduolius rašoma 2018-tųjų metų S. Markidis ir kt. straipsnyje [MCL+18].

3.2. Duomenų rinkinys

3D modelius skaitmeninantiems GAN metodams yra reikalingi mokymo duomenys susidedantys iš porų $\{x_i, y_i\}$, kur x_i yra objekto trimatis modelis, o y_i yra tą objektą atvaizduojanti dvimatė nuotrauka. Norint panaikinti priklausomybę nuo naudojamų duomenų ir tinkamai palyginti visus metodus, kurie bus kuriami ar pritaikomi magistro baigiamojo darbo metu, visiems metodams bus naudojami tie patys duomenys, gauti iš *ShapeNet* duomenų rinkinio.

ShapeNet – tai 2015-taisiais metais Stanfordo universiteto mokslininkų surinktas, kompiuteriu sukurtų CAD (angl. Computer-Aided Design) 3D modelių rinkinys [CFG+15]. Rinkinyje yra laikoma virš 51000 objektų informacija, kuri susideda iš daugelio objektų kategorijų, o objektai yra papildomi su įvairiomis anotacijomis kaip lygiavimo informacija, mastelis ir kt. Taip pat rinkinyje yra laikomos objektų nuotraukos, gautos patalpinant jų modelius į virtualią erdvę ir žvelgiant į juos iš skirtingų kampų.

Anotacijos bei nuotraukos, turimos kiekvienam duomenų rinkinio objektui nėra standartizuotos, tad gali būti sunkiai pritaikomos naudojant automatizuotus duomenų nuskaitymo metodus. Kad



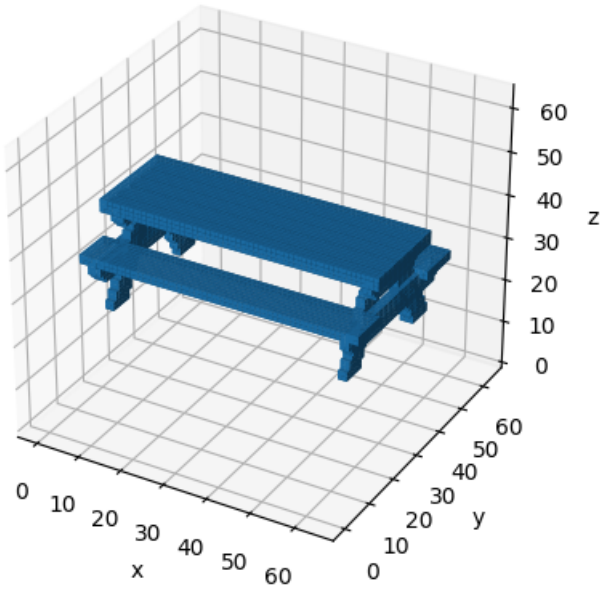
14 pav. Lauko stalo 3D modelio nuotraukos iš trijų skirtingų kampų [SCH15]

pritaikyti duomenis panaudojimo atvejams, kuriems tai yra svarbu, Stanfordo Universitetas 2016-taisiais metais išleido dar vieną duomenų rinkinį – *ShapeNetSem* [SCH15]. Tai pradinio *ShapeNet* duomenų rinkinio poaibis, laikantis virš 12000 objektų iš 270 kategorijų, su informacija apie jų fizinius atributus, kaip sudedamąsias medžiagas, dydžius, svorius ir pan. Pagrindinis šių duomenų privalumas atliekamam darbui yra standartizuoti nuotraukų rinkiniai, kurie atvaizduoja objektus iš trylikos skirtingų kampų ir kiekvienam kampui nuotraukos yra pavadintos vienodai. Tai leidžia atsirinkti norimus atvaizdus nedarant jokių papildomų patikrinimų ar duomenų perdirbimo.

3D modeliai yra laikomi OBJ arba binvox formatu. Pirmasis talpina objekto paviršiaus tinklelio (angl. mesh) viršūnių koordinates, o antrasis šią informaciją laiko kaip loginių, dvejetainių reikšmių trimatę matricą. Tiesos reikšmės binvox formato duomenyse nusako vietas trimatėje erdvėje, kuriose laikomi vokseliai – tūriniai pikseliai. 3D GAN algoritmuose dažniausiai naudojamas vokselinis modelių formatas, kadangi didžioji jų dauguma yra pagristį kitais, su nuotraukomis ir pikseliais dirbančiais algoritmais. Taip laikomi duomenys užima labai mažai vietos, lyginant su kitais trimatės informacijos formatais, kadangi kiekvienai erdvės pozicijai reikalingas tiktai vienas bitas informacijos. Tai itin naudinga giliojo mokymo metodams, kurie naudoja ir atmintyje laiko didelius kiekius mokymo duomenų.

Tarp šių duomenų, pradiniams testaviamams buvo pasirinkta naudoti objektus iš stalų kategorijos. Tai yra daugiausiai egzempliorių turinti objektų rūšis *ShapeNetSem* duomenų rinkinyje ir norint per daug neišplėsti mokymo laiko ir operatyviosios atminties reikalavimų su kitomis objektų kategorijomis pasirinktą visus tinklus mokinti naudojant tuos pačius 670 stalo objektų. Šį pasirinkimą buvo planuota praplėsti, tačiau to nebuvo spėta padaryti, kadangi tinklų apmokymas vien su tokiu kiekiu duomenų jau užtrukdavo iki 20 valandų ir duomenų atsirinkimas reikalauja nemažai rankinio darbo, nes objektai rinkinyje nėra griežtai suskirstyti pagal kategorijas.

ShapeNetSem paketą sudaro dviejų tipų vokseliniai modeliai – atvaizduojantys tiktai objekto



15 pav. 3D Lauko stalo modelio trimatė vokselinė reprezentacija [SCH15]

paviršių ir atvaizduojantys užpildytą objektą. Realizuojant GAN metodą, kuris aprašomas sekančiame skyrelyje buvo pastebėta, kad pirmasis duomenų variantas yra mažiau tinkamas GAN mokymui. Kai tiktai maža dalis visos turimos trimatės erdvės susideda iš vokselių, netekties (angl. loss) mažinimo algoritmai užstringa prie tokių svorių, su kuriais modelių generatorius visas reikšmes nustato kaip neigiamas ir gaunama tuščia erdvė be jokių vokselių. Lyginant tokį modelį su tikrais, gaunamas santykinai didelis tikslumas ir pasidaro sudėtinga rasti nusileidimo gradientus. Tačiau didelė dalis duomenų rinkinyje turimų užpildytų modelių buvo sugadinti, su trūkstamais ar neteisingai suformuotais duomenimis, taigi jie visi buvo kuriami iš naujo naudojantis Patrick Min sukurtu binvox konvertavimo įrankiu [Min04].

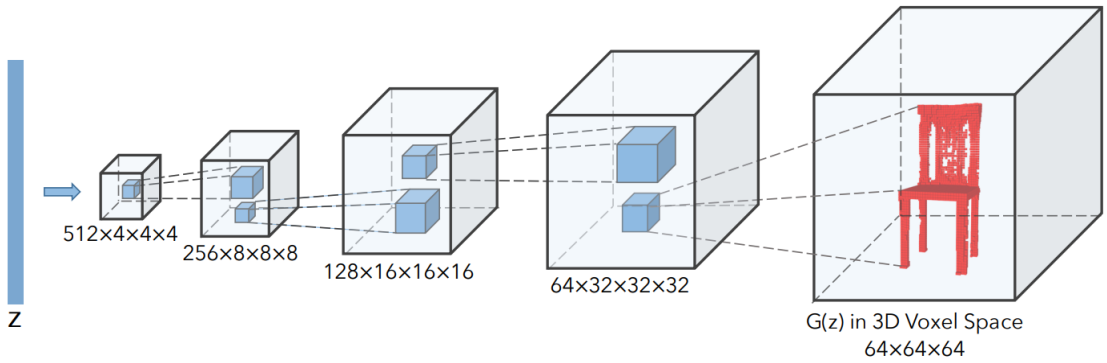
Tolimesniuose eksperimentuose taip pat buvo pastebėta, kad ne visi modeliai turėjo tokį pati pasukimą, ir tam tikros jų dalys nebuvo tinkamai sugeneruotos kuriant modelius iš naujo. Kad išspręsti šias problemas buvo atsižvelgiama į *ShapeNetSem* duomenų rinkinyje įtrauktą meta duomenų (angl. metadata) failą, kuris laikė informaciją apie didžiosios dalies turimų objektų kanonišką priekio orientaciją. Nuspręsta visus objektus standartizuoti ir pasukti taip, kad pavaizduojant juos su ”mathplotlib” kodo paketu, objektai vizualiai būtų pasukti į priekį (į x-ų ašį), tačiau tai nedarė jokios įtakos neuroniniams tinklas, svarbiasia savybė buvo jų vienodumas. Vokselinių modelių kūrimiui naudotą bash kodą (angl. script) galima pamatyti 1 priede.

Pilnai užkraunant abibūdintą stalų duomenų rinkinį, buvo išeikvojamas labai didelis kiekis kompiuterio operatyviosios bei vaizdo plokštės atminties, tad naudojant *TensorFlow* tinkinio ”dataset” motodą, visi duomenys buvo išsaugomi į atskirą failų rinkinį. Imant duomenis iš šių failų, gali būti kraunami tiktai vienos ar kelių partijų duomenys, taip beveik panaikinant duomenų rinkinio poveikį naudojamam atminties kiekui.

3.3. 3D-VAE-GAN metodas

Siekiant įvykdyti pirmąjį magistro baigiamojo darbo uždavinį buvo įgyvendintas JiaJun Wu ir kt. 2016-tųjų metų straipsnyje aprašomas trimačius modelius iš nuotraukų skaitmeninantis 3D-VAE-GAN metodas [WZX+16]. Tai vienas pirmųjų tokio tipo algoritmų ir galėjo sukurti tiktai $64 \times 64 \times 64$ raiškos vokselinius modelius, tačiau yra puikus atskaitos taškas kitų algoritmų kūrimui ir lyginimui.

Metodas susideda iš trijų dalių. Pirmoji dalis yra generatorius – tai tūrinis konvoliucinis neuroninis tinklas, kuris iš 200 reikšmių triukšmo vektoriaus sugeneruoja vokselinį trimatį 3D objektą. Straipsnyje [WZX+16] šis tinklas yra sudarytas iš penkių trimatės dekonvoliucijos sluoksnių su dydžiais $4 \times 4 \times 4$, žingsniais (angl. strides) $\{1^3; 2^3; 2^3; 2^3; 2^3\}$ ir kanalų skaičiais (angl. channels) {512, 256, 128, 64, 1}. Tarp šių sluoksnių yra partijų normalizacijos ir ReLU sluoksniai. Pirmasis normalizuoja praeito sluoksnio išvestį taip, kad visų reikšmių vidurkis būtų artimas 0 ir standartinis nuokrypis artimas 1, o antrasis neigiamus skaičius paverčia į 0. Tai yra daroma tinklo stabilizavimui, susiaurinant jo įmanomas vidines reikšmes. Bendra generatoriaus sudėtis matoma 16 paveikslėlyje.



16 pav. 3D-VAE-GAN generatoriaus tinklas [WZX+16]

Antroji metodo dalis yra diskriminuojantis tinklas – tai generatoriaus veidrodinis atvaizdas, kuris kaip įvestį priima $64 \times 64 \times 64$ 3D modelį ir išveda vieną realųjį skaičių nusakantį tikimybę, kad paduotas modelis yra tikras. Tinklas susideda iš penkių trimatės konvoliucijos sluoksnių su dydžiais $4 \times 4 \times 4$, žingsniais $\{2^3; 2^3; 2^3; 2^3; 1^3\}$ ir kanalų skaičiais $\{64, 128, 256, 512, 1\}$. Vietoje generatoriaus ReLU sluoksnių, naudojami nesandarūs (nutekantys) ReLU sluoksniai (angl. leaky ReLU), kurie neigiamas reikšmes padaugina iš duoto koeficiento, šiuo atveju 0,2, taip sumažinant neigiamų reikšmių svarbą, bet jų nepanaikinant. Abiejų dalių pabaigoje pridedamas sigmoidinės aktyvacijos sluoksnis, kuris susiaurina tinklo išvesties reikškmes į intervalą [0, 1].

Paskutinė modelio dalis yra nuotraukų koduotojas (angl. encoder), kurio idėja paimta iš variacinių automatinių koduotuvų, arba VAE, tinklų architektūros (angl. Variational AutoEncoder). Tai tinklas, kuris išgauna jam paduodamų nuotraukų savybių tikimybinį pasiskirstymą, pagal kurį vėliau sukuriamas nuotrauką apibūdinantis triukšmo vektorius, kuris paduodamas į generatoriaus tinklą, kad sukurti nuotrauką atitinkantį modelį. Pagal straipsnio nurodymus šis tinklas sudarytas iš penkių dvimačių konvoliucinių sluoksnių su atitinkamais dydžiais – $\{11^2; 5^2; 5^2; 5^2; 8^2\}$, žingsniais $\{4^2; 2^2; 2^2; 2^2; 1^2\}$ ir kanalų skaičiais $\{64, 128, 256, 512, 400\}$. Kaip įvestį nurodytas tinklas priima spalvotas, 256×256 raiškos, nuotraukas ir išveda 400 reikšmių masyvą, nusakantį norimo triukšmo vektoriau tikimybinį pasiskirstymą.

Metodo straipsnyje nėra nurodoma, kaip tiksliai iš šio masyvo turėtų būti išgaunamas norimas 200-ų reikšmių triukšmo vektorius, kuris vėliau bus paduodamas į modelius generuojantį tinklą. Tam reikalingos 200 vidurkių reikšmių ir 200 dispersijos reikšmių, pagal kurias bus generuojami atsitiktiniai, šiuos parametrus atitinkantys, normaliojo (Gauso) pasiskirstymo skaičiai. Pirmiausia buvo badoma sekti *TensorFlow* dokumentacijoje pateiktą VAE pavyzdį, kuriame nurodoma dalinti gautą 400 reikšmių masyvą į dvi dalis – pirmoji dalis yra vidurkių reikšmės, o antroji dalis yra dispersijos reikšmės. Tačiau šiuo metodu nebuvo sėkmingai gautas 3D modelių generavimas, tad koduotojas buvo perdarytas pagal kitas pavyzdines VAE implementacijas.

Tinklo pabaigoje pridedami trys papildomi sluoksniai – du tankiai sujungti sluoksniai, kurie išsišakoja nuo paskutinio konvoliucinio sluoksnio ir atitinkamai atrenka po 200 vidurkio ir dispersijos reikšmių, ir vienas ”lambda” sluoksnis kuris pritaiko perparametravimo (angl. reparametrization) funkciją, kad išgauti reikalingą triukšmo vektorių. Vietoje atsitiktinių skaičių generavimo, pritaikomas formulėje (7) matomas veiksmas, kad išvengti negebėjimo atlikti gradientų atgalinės sklaidos

pro atsitiktinius kintamuosius mokymo metu. Formulėje z yra norimas triukšmo vektorius, μ – tikimybių pasiskirstymo vidurkiai, σ – pasiskirstymo dispersija, o ϵ yra atsitiktinės reikšmės iš normaliojo pasiskirstymo su vidurkiais 0 ir nuokrypiu 1. Visų modelių diagramas galima pamatyti 2 priede.

$$z = \mu + \sigma \odot \epsilon \tag{7}$$

Apsibrėžus visas tinklų architektūras gali būti pradėtas 3D-VAE-GAN apmokymas. Laikant, kad $\{x_i, y_i\}$ yra trimačių modelių ir juos atitinkančių nuotraukų poros, D, E ir G yra atitinkamai diskriminatorius, koduotojas ir generatorius, D_{KL} yra Kullback-Leiber nuokrypis, o t iteracijos skaičius, mokymas susideda iš trijų žingsnių:

1. Atnaujinamas diskriminatorius mažinant dvimatės kryžminės entropijos (angl. cross entropy) nuostolį: $\log D(x_i) + \log(1 - D(G(z_t)))$
2. Atnaujinamas koduotojas mažinant KL nuokrypio ir rekonstrukcijos paklaidos nuostolį: $D_{KL}(N(E_{vid}(y_i), E_{dis}(y_i)) \,||\, p(z_t)) + ||G(E(y_i)) - x_i||_2$
3. Atnaujinamas generatorius mažinant kryžminės entropijos prieš diskriminatoriaus spėjimus ir rekonstrukcijos nuostolį: $\log(1 - D(G(z_t))) + ||G(E(y_i)) - x_i||_2$

Kullback-leiber nuokrypis skatina koduotojo tinklą nenutolti nuo pradinio triukšmo vektoriaus tikimybinio pasiskirstimo, kuris apibrėžiamas su vidurkiu 0 ir dispersija 1. Tai leidžia generatoriui lengviau pritaikyti svorius išmoktus mokantis apgauti diskriminatorių, kadangi šio mokymo metu yra naudojamas būtent toks tikimybinis pasiskirstymas triukšmo vektoriaus generavimui. Autoriai nenurodo tikslaus KL nuokrypio skaičiavimo būdo, tad buvo bandomi ir lyginami įvairūs, šiek tiek tarpusavyje išsiskiriantys, metodai.

Pirmiausia buvo bandoma pritaikyti paprastą *TensorFlow* "kl_divergence" netekties funkciją, tačiau šis metodas skaičiuoja nuokrypi tiktai tarp dviejų paprastų vektorių, o ne tikimybių pasiskirstymų, tad buvo manyta, kad jis nėra tinkamas. Sekantys bandymai buvo atliekami aprašant pilną nuokrypio matematinę formulę, kurios yra keli variantai priklausomai nuo panaudojimo konteksto, tačiau radus geriausiai tinkantį, buvo susidurta su dalybos iš nulio (arba skaičiaus labai arti nulio), gauto skaičiuojant dispersijos reikšmių diskriminantą, problema, dėl kurios visos mokyme naudojamos reikšmės įgauna ne skaitines NaN (angl. Not a Number) reikšmes. Vienas stabiliausių rastų metodų buvo gautas pasitelkiant papildomą "*TensorFlow Probabilities*" programinės įrangos paketą, tačiau buvo gaunamos labai didelės reikšmės ir su jomis nebuvo gaunamas geras nuostolio mažinimas mokymo metu. Galiausiai buvo sugrįžta prie paprastosios *TensorFlow* KL nuokrypio netekties, sudedant vidurkių ir dispersijos reikšmių vektorius į vieną masyvą ir lyginant juos su taip pat sukonfigūruotu tikslo tikimybiniu pasiskirstymu, padalinus galutinį rezultatą iš partijos dydžio.

Rekonstrukcijos netektis straipsnyje nurodyta kaip tikrų ir sugeneruotų modelių Euklido atstumas, t.y. šaknis ištraukta iš visų trimatės erdvės koordinačių reikšmių kvadratinių skirtumų. Atlikus daugelį eksperimentų ir atsižvelgiant į kitus panašių tinklų pavyzdžius pastebėtą, kad geresni rezultatai yra gaunami pritaikant vidutinį kvadratinį Euklido atstumą, neatliekant šaknies ištraukimo.

Nors taip gaunama didesnė reikšmė, optimizatoriaus tikslas išlieka tas pats, o su gautomis didesnėmis reikšmėmis, gradientų radimas tampa paprastesnis.

Straipsnyje nurodomi tokie mokymo parametrai:

- Epochų skaičius lygus 100;
- Partijos dydis lygus 100;
- Naudojama ADAM optimizacija su $\beta = 0{,}5$;
- Generatoriaus ir diskriminatoriaus mokymo sparta (angl. learning rate) lygi $0{,}0025$ ir 10^{-5} atitinkamai;
- Koduotojo mokymo sparta nėra nurodoma, tačiau minima, kad šis tinklas yra paremtas Larsen ir kt. 2016-tųjų metų straipsnyje aprašomu metodu, kuriame nurodoma $0{,}0003$ sparta [LSW15];
- Diskriminatorius yra apmokomas tiktai tada, kai jo tikslumas yra mažesnis nei 80 %.

Nors šios reikšmės buvo naudotos kaip atskaitos taškas atliktiems eksperimentams, jos buvo pritaikytos tiksliam autorių naudotam duomenų rinkiniui, jų konkretiems visų naudotų nuostolio funkcijų įgyvendinimams ir panašiems faktoriams. Bandant išgauti geras mokymosi tendencijas pastebėta, kad labai sudėtinga išlaikyti generatoriaus ir diskriminatoriaus tinklus lygioje kovoje, kuri yra GAN tinklų idėjos pamatas. Dažnai arba vienas arba kitas tinklas apsimoko per greitai ir jo priešininkas negali išgauti gradientų informacijos kuri reikalinga tolimesniam tobulėjimui. Šios kovos rezultatai priklausydavo nuo mažiausių pokyčių mokymo nustatymuose, nuo naudojamo duomenų poaibio ir nuo tam tikrų kiekvienos paleisties skirtumų, netgi kai išlaikomi tie patys nustatymai. Ant diskriminatoriaus uždėtas limitas, sustabdantis jo apmokymą, jam pasiekus 80 % ar didesnį tikslumą šiek tiek padėjo šioms situacijoms, tačiau to dažniausiai neužtekdavo ir tai niekaip nepadėdavo, jeigu generatoriaus tinklas išsiveržda vo į priekį. Norint šiek tiek stabilizuoti tinklų mokymą taip pat buvo pridedamos nedidelės atsitiktinės triukšmo reikšmės prie tikrųjų trimačių modelių bei diskriminatoriaus siekiamų tiesios reikšmių, ko pasekmėje stabilumo patobulėjimas buvo neženklus, tačiau pastebimas.

Galutiniai magistro darbe gauti rezultatai buvo pasiekti naudojant duomenų rinkinio skiltyje apibūdintą objektų rinkinį, 80 % jo priskiriant mokymo duomenims, o likusius 20 % – testavimui. Kadangi viena epocha su tokiu duomenų kiekiu užtrukdavo maždaug 10 minučių, nebuvo galimybės pilnai apmokinti tinklus su įvairias nustatymais ir lyginti gautus rezultatus. Todėl kiekvieno eksperimento metu iš pradžių buvo apskaičiuojamos pirmos kelios epochos ir stebimos mokymo tendencijos. Tikslas pamatyti kažkokius konkurencijos ženklus tarp generatoriaus ir diskriminatoriaus tinklų. Jeigu matoma, kad po kiekvienos, ar kas kelias, partijas apgavimų ir tikslių atspėjimų skaičius svyruoja į vieną ir į kitą pusę, tai duoda gerą indikaciją, kad tokį mokymą vertą tęsti, kadangi jis gali privesti prie tinkamų rezultatų.

Pastebėta, kad pirmosios dvi ar trys epochos gali parodyti klaidingas tendencijas, ir vėliau persiversti visiškai į kitą pusę. Todėl ieškant tinkamų mokymo nustatymų sėkmingiausias rastas būdas

buvo apskaičiuoti pirmas penkias epochas ir matant geras tendencijas pratęsti mokymą iki 20-30 epochų.

Taip pat autoriai nurodo 2 svorius α_1 ir α_2 kurių reikšmės lygios 5 ir 10^{-4} atitinkamai. Šie svoriai nusako KL nuokrypio ir rekonstrukcijos nuostolio svarbą bendram tinklų mokymui. Nors jie minimi tiktai prie bendros modelio netekties aprašymo ir nėra aprašyti prie mokymo žingsnių, buvo nuspręstą juos panaudoti, tačiau taip pat pritaikyti prie konkrečių mokymo sąlygų, kad padidinti modelio šansus tinkamai apsimokyti. Svoriai priklauso nuo konkrečios nuostolio funkcijų implementacijos, tad šiam darbui kaip stabiliausias variantas buvo pasirinktos reikšmės $\alpha_1 = 5$ ir $\alpha_2 = 0.0005$.

3.4. HA-GAN metodas

Norint sukurti už kitus pranašesnį trimačio skaitmeninimo GAN algoritmą, sekančiam pritaikomam metodui buvo pasirinkta pakoreguoti Li Sun ir kt. 2020 metais pasiūlytą ir 2022 metais pilnai pabaigtą HA-GAN algoritmą [SCX+22]. Originaliai metode prie GAN yra pridedamas trimačių modelių koduotojas, kuris padeda padidinti dirbtinių neuroninių tinklų mokymo stabilumą ir išvengti modos griūties (angl. mode collapse), kurios metu generatorius pradeda kurti tiktai vieną labai įtikinantį modelį. Jeigu šis koduotojas galėtų būti sėkmingai pakeistas dvimačiu kuoduotoju, išgaunančiu objekto nuotraukos ypatybes, būtų galima gauti konkurencingus objektų skaitmeninimo iš vienos nuotraukos rezultatus.

HA-GAN basivaržantieji tinklai mokomi neįprastu eiliškumu (žr. 6 pav.). Iš viso yra trys generatyviniai tinklai ir du diskriminuojantieji tinklai. Pirmasis yra bendrasis generatorius G^A, iš 1024 reikšmių dydžio triukšmo vektoriaus sukuriantis $64 \times 64 \times 64 \times 64$ išvestį, po kurios modelis išsišakoja į dalinių aukštos raiškos modelių iškarpų generavimą su generatoriumi G^H ir pilnų žemos raiškos modelių generavimą su generatoriumi G^L. Bendrojo generatoriaus G^A mokymas ir mokymo metu gauti svoriai yra priklausomi nuo abiejose šakose gaunamų rezultatų, taigi galutinio

3 lentelė. Sun ir kt. straipsnyje [SCX+22] naudojamas bedrasis generatoriaus tinklas.

G^A		
Sluoksnis	**Dydis, žingsniai**	**Išvesties dydis**
Įvestis	-	1×1024
Dense	-	$4 \times 4 \times 4 \times 512$
Conv3D	$3 \times 3 \times 3, 1$	$4 \times 4 \times 4 \times 512$
GroupNorm + ReLU	-	$4 \times 4 \times 4 \times 512$
Upscale3D	-	$8 \times 8 \times 8 \times 512$
Conv3D	$3 \times 3 \times 3, 1$	$8 \times 8 \times 8 \times 512$
GroupNorm + ReLU	-	$8 \times 8 \times 8 \times 512$
Upscale3D	-	$16 \times 16 \times 16 \times 512$
Conv3D	$3 \times 3 \times 3, 1$	$16 \times 16 \times 16 \times 256$
GroupNorm + ReLU	-	$16 \times 16 \times 16 \times 256$
Upscale3D	-	$32 \times 32 \times 32 \times 256$
Conv3D	$3 \times 3 \times 3, 1$	$32 \times 32 \times 32 \times 128$
GroupNorm + ReLU	-	$32 \times 32 \times 32 \times 128$
Upscale3D	-	$64 \times 64 \times 64 \times 128$
Conv3D	$3 \times 3 \times 3, 1$	$64 \times 64 \times 64 \times 64$
GroupNorm + ReLU	-	$64 \times 64 \times 64 \times 64$

3D modelio kūrimo metu jis perima bendras viso objekto savybes iš G^L ir perduoda jas į G^H, taip leidžiant jam sugeneruoti tinkamesnius, pilno dydžio, aušto raiškos vokselinius modelius.

G^A tinklas susideda iš vieno pilnai sujungto (angl. Dense) sluoksnio ir penkių trimatės konvoliucijos sluoksnių, tarp kurių įterpiamos grupių normalizacijos, ReLU ir mastelio padidinimo (interpoliacijos) funkcijos (žr. 3 lentelę). Grupių normalizacija – tai Yuxin Wu ir Kaiming He 2018 metų straipsnyje pasiūlytas metodas [WH18], kuris padalina sluoksnio kanalus į norimą skaičių grupių (HA-GAN atveju 8) ir atlieka normalizaciją pagal kiekvienos grupės statistinius duomenis. Ji yra naudojama vietoje partijų normalizacijos matytos 3D-VAE-GAN metode, dėl ženkliai mažesnės paklaidos apmokant tinklus su ypač mažomis partijomis, kurios yra naudingos kompiuterinio matymo (angl. computer-vision) srityje. Tuo tarpu mastelio padidinimo funkcija yra reikalinga, kadangi tinkle yra naudojami paprastieji trimatės konvoliucijos sluoksniai, vietoje dekonvoliucijos sluoksnių, kurie patys galėtų padidinti gaunamą išvestį.

Aukštos raiškos modelius generuojantis tinklas G^H pratęsia bendrojo generatoriaus atliktus pakeitimus priklausomai nuo jam paduotos įvesties. Mokymo metu, iš G^A gautų rezultatų yra pasirenkama $\frac{1}{8}$ dydžio iškarpa, prasidedanti nuo atsitiktinio indekso r. Visos r reikšmės turi tokią pačią tikimybę, todėl pasirinkti modelio pjūviai gali įvairiai persilieti, taip geriau padengiant galimus pjūvių sandūros taškus. Atsižvelgiant į turimus mokymo duomenis ir juose atliktą pasukimo suvienodinimą, pjaustymas yra atliekamas pagal pirmąją (x) koordinačių ašį, taip išvengiant tuščių pjūvių, kuriuose neturi būti jokių vokselių. Modelių generavimo metu yra paduodama visa 64^4 dydžio G^A išvestis ir generuojamas pilnas, aukštos raiškos (256^3) 3D modelis, kadangi atmintyje nebeturi būti laikomi mokymo metu reikalingi gradientai ir atminties trūkumas nebėra aktualus.

Žemos raiškos generatorius G^L yra naudojamas tiktai mokymo metu, perduodant visą G^A išvestį ir kuriant 64^3 dydžio trimačius modelius. Abiejuose tinkluose yra naudojami tokie patys dirbtinių neuronų sluoksniai kaip ir bendrajame generatoriuje, tačiau G^L nenaudoja mastelio padidinimo ir abiejų tinklų pabaigoje yra pridedama hiperbolinio tangento "Tanh" aktyvacijos funkcija, kuri susiaurina tinklo išvestį į rėžį $[-1, 1]$. Tai reiškia, kad realių objektų vokseliniuose duomenyse nuliai turi būti pakeičiami į -1, kad atitikti iš generatorių gaunamas reikšmes. Pilnos generatorių struktūros matomos 4 lentelėje.

4 lentelė. Sun ir kt. straipsnyje [SCX+22] naudojami aukštos ir žemos raiškos modelių generatoriai.

G^H			G^L		
Sluoksnis	**Dydis, žings.**	**Išvesties dydis**	**Sluoksnis**	**Dydis, žings.**	**Išvesties dydis**
Įvestis	-	$64 \times 64 \times 64 \times 64$	Įvestis	-	$64 \times 64 \times 64 \times 64$
Upscale3D	-	$128 \times 128 \times 128 \times 64$			
Conv3D	$3 \times 3 \times 3, 1$	$128 \times 128 \times 128 \times 32$	Conv3D	$3 \times 3 \times 3, 1$	$64 \times 64 \times 64 \times 32$
GroupNorm + ReLU	-	$128 \times 128 \times 128 \times 32$	GroupNorm + ReLU	-	$64 \times 64 \times 64 \times 32$
Upscale3D	-	$256 \times 256 \times 256 \times 32$			
Conv3D	$3 \times 3 \times 3, 1$	$256 \times 256 \times 256 \times 1$	Conv3D	$3 \times 3 \times 3, 1$	$64 \times 64 \times 64 \times 16$
Tanh	-	$256 \times 256 \times 256 \times 1$	GroupNorm + ReLU	-	$64 \times 64 \times 64 \times 16$
			Conv3D	$3 \times 3 \times 3, 1$	$64 \times 64 \times 64 \times 1$
			Tanh	-	$64 \times 64 \times 64 \times 1$

Generatoriams G^H ir G^L atitinkamai priskiriami aukštos ir žemos raiškos modelių diskrimi-

5 lentelė. Sun ir kt. straipsnyje [SCX+22] naudojami aukštos ir žemos raiškos modelių diskriminatoriai.

D^H			D^L		
Sluoksnis	**Dydis, žings.**	**Išvesties dydis**	**Sluoksnis**	**Dydis, žings.**	**Išvesties dydis**
Įvestis	-	$32 \times 256 \times 256 \times 1$	Įvestis	-	$64 \times 64 \times 64 \times 1$
Conv3D	$4 \times 4 \times 4, 2$	$16 \times 128 \times 128 \times 16$	Conv3D	$4 \times 4 \times 4, 2$	$32 \times 32 \times 32 \times 32$
SpectralNorm + LeakyReLU	-	$16 \times 128 \times 128 \times 16$	SpectralNorm + LeakyReLU	-	$32 \times 32 \times 32 \times 32$
Conv3D	$4 \times 4 \times 4, 2$	$8 \times 64 \times 64 \times 32$	Conv3D	$4 \times 4 \times 4, 2$	$16 \times 16 \times 16 \times 64$
SpectralNorm + LeakyReLU	-	$8 \times 64 \times 64 \times 32$	SpectralNorm + LeakyReLU	-	$16 \times 16 \times 16 \times 64$
Conv3D	$4 \times 4 \times 4, 2$	$4 \times 32 \times 32 \times 64$	Conv3D	$4 \times 4 \times 4, 2$	$8 \times 8 \times 8 \times 128$
SpectralNorm + LeakyReLU	-	$4 \times 32 \times 32 \times 64$	SpectralNorm + LeakyReLU	-	$8 \times 8 \times 8 \times 128$
Conv3D	$2 \times 4 \times 4, 2$	$2 \times 16 \times 16 \times 128$	Conv3D	$4 \times 4 \times 4, 2$	$4 \times 4 \times 4 \times 256$
SpectralNorm + LeakyReLU	-	$2 \times 16 \times 16 \times 128$	SpectralNorm + LeakyReLU	-	$4 \times 4 \times 4 \times 256$
Conv3D	$2 \times 4 \times 4, 1$	$1 \times 8 \times 8 \times 256$	Conv3D	$4 \times 4 \times 4, 1$	$1 \times 1 \times 1 \times 1$
SpectralNorm + LeakyReLU	-	$1 \times 8 \times 8 \times 256$	Flatten	-	1
Conv3D	$1 \times 4 \times 4, 1$	$1 \times 4 \times 4 \times 512$			
SpectralNorm + LeakyReLU	-	$1 \times 4 \times 4 \times 512$			
Conv3D	$1 \times 4 \times 4, 1$	$1 \times 1 \times 1 \times 128$			
SpectralNorm + LeakyReLU	-	$1 \times 1 \times 1 \times 128$			
Flatten	-	128			
Concatenate($r \div r_{max}$)	-	129			
Dense	-	64			
SpectralNorm + LeakyReLU	-	64			
Dense	-	1			

natoriai D^H ir D^L. Pirmasis lygina generatoriaus sukurtas modelio iškarpas su realiomis modelio iškarpomis, kurios buvo išgautos naudojant r pjūvio indeksą, proporcingai pritaikytą prie indekso naudoto generatoriuje G^H, taip užtikrinant, kad lyginamos tos pačios modelio dalys. Šio diskriminatoriaus tikslas yra užtikrinti, kad mažos 3D modelio detalės yra atkuriamos realistiškai. Antrasis diskriminatorius tuo tarpu lygina žemos raiškos pilnus modelius, bandant nuspėti kurie yra tikri ir kurie yra sugeneruoti, taip užtikrinant, kad yra išlaikoma bendra objekto struktūra.

Abu diskriminatoriaus tinklai yra konstruojami naudojant trimatės konvoliucijos sluoksnius, tačiau kitaip nei generatorių tinkluose, naudojami spektrinės normalizacijos ir nesandarūs ReLU sluoksniai su koeficientu $0{,}2$. Spektrinė normalizacija, taip pat pirmą kartą pasiūlyta 2018 metais, Tekeru Miyato ir kt. straipsnyje [MKK+18], kuriame autoriai parodė, kad šis normalizacijos būdas ypač palankus GAN tinklų stabilizavimui pritaikant jį diskriminatoriaus tinkle. Tai matematiškai gana sudėtinga, tačiau mažai kompiuterinių resursų reikalaujanti funkcija, kuri suspaudžia dirbtinio neuroninio tinklo svorių matricos masteli, sumažinant jos spektrinę normą (angl. spectral norm). D^H pabaigoje taip pat pridedami du (naudojama metodo pavyzdiniame kode) arba trys (nurodyta straipsnyje) pilnai sujungti sluoksniai, naudojami galutiniam atsakymui išgauti. Tikslūs abiejų tinklų sluoksnių išsidėstymai matomi 5 lentelėje.

D^H ir D^L tinklų pabaigoje nėra pridedama aktyvacijos funkcija, tad grąžinami rezultatai neapribojami į tam tikrą rėžį, tačiau mokymo metu, skaičiuojant netektį, vienetai nusako diskriminatoriaus spėjimą, kad objektas yra tikras, o nuliai – spėjimą, kad objektas nėra tikras. Taip pat, prie paskutiniojo D^H konvoliucijos bloko išvesties yra prijungiama r reikšmė, suspausta į rėžį [0, 1], taip leidžiant diskriminatoriaus svoriams geriau pritaikyti galutinę išvesti pagal konkrečia tikrinamą iškarpą.

Pritaikant HA-GAN metodą prie dvimačių nuotraukų skaitmeninimo, tam sukuriant naują koduotojo tinklą, nuspręsta kad geriausiai tinkantis variantas yra perdaryti jau esamus metodo koduotojus E^H ir E^G, sujungiant juos į vieną dvimačių atvaizdų koduotoją E^{2D}. Taip išlaikoma

6 lentelė. Modifikuotame HA-GAN naudojamas koduotojas.

E^{2D}		
Sluoksnis	**Dydis, žingsniai**	**Išvesties dydis**
Įvestis	-	$256 \times 256 \times 3$
Conv2D	$4 \times 4, 2$	$128 \times 128 \times 32$
GroupNorm + ReLU	-	$128 \times 128 \times 32$
Conv2D	$4 \times 4, 2$	$64 \times 64 \times 64$
GroupNorm + ReLU	-	$64 \times 64 \times 64$
Conv2D	$4 \times 4, 2$	$32 \times 32 \times 32$
GroupNorm + ReLU	-	$32 \times 32 \times 32$
Conv2D	$4 \times 4, 2$	$16 \times 16 \times 64$
GroupNorm + ReLU	-	$16 \times 16 \times 64$
Conv2D	$4 \times 4, 2$	$8 \times 8 \times 128$
GroupNorm + ReLU	-	$8 \times 8 \times 128$
Conv2D	$4 \times 4, 2$	$4 \times 4 \times 256$
GroupNorm + ReLU	-	$4 \times 4 \times 256$
Conv2D	$4 \times 4, 2$	$1 \times 1 \times 1024$

kiek įmanoma panašesnė tinklų sudėtis ir suderinamumas su kitomis metodo dalimis. Visi abiejų koduotojų sluoksniai buvo surinkti į bendrą tinklą, pakeičiant trimatę konvoliuciją į dvimatę, taip panaikinant vieną matmenį iš kiekvieno sluoksnio išvesties. Koduotojas priima 256×256 raiškos spalvotas nuotraukas kaip įvestį, ir išveda viena 1024-ių reikšmių triukšmo vektorių, apibūdinantį paduotų vaizdų savybes. Pilna tinklo struktūra, kartu su grupių normalizacijos ir ReLU sluoksniais pavaizduojama 6 lentelėje.

Baigiamojo darbo metu sukurto, koreguoto HA-GAN metodo mokymas yra padalinamas į tris žingsnius, atliekamus su kiekviena duomenų partija:

1. Diskriminatorių atnaujinimas – paėmus atsitiktinių reikšmių triukšmo vektorių, generatoriaus tinklai sukuria aukštos raiškos modelių dalines iškarpas ir žemos raiškos modelius. Tuomet diskriminatoriai D^H ir D^L atitinkamai yra mokinami atskirti juos nuo paduotų realių iškarpų ir modelių, mažinant kryžminės entropijos nuostolį tarp gautų spėjimų ir teisingų atsakymų – $\log D^H(x_i^H) + \log(1 - D^H(G^H(G^A(z_t)))) + \log D^L(x_i^L) + \log(1 - D^L(G^L(G^A(z_t))))$.

2. Generatorių atnaujinimas – taip pat kaip ir praeitame žingsnyje, generatoriaus tinklai sukuria aukštos raiškos modelių dalines iškarpas bei žemos raiškos modelius ir diskriminatoriai bando juos atskirti nuo tikrųjų, tačiau šį kartą generatorių tinklai G^A, G^H ir G^L mokinami apgauti diskriminatorius mažinant kryžminės entropijos nuostolį tarp diskriminatoriaus spėjimų apie sugeneruotus modelius ir vienetų (tiesios reikšmių) vektoriaus.

3. Koduotojo atnaujinimas – paėmus trimačių modelių nuotraukas, koduotojas yra mokinamas sukurti tokį triukšmo vektorių, kurį padavus į generatorių tinklus G^A ir G^L būtų gaunama kuo mažesnė rekonstrukcijos netektis lyginant gautus modelius su realiais. Kitaip nei 3D-VAE-GAN metode, vietoje Euklido atstumo yra skaičiuojama vidutinė absoliučioji paklaida – $||G^L(G^A(E^{2D}(y_i))) - x_i||_1$.

3D-VAE-GAN mokymo metu buvo įtariama, jog naudotas duomenų skirstymas į $\{x_i, y_i\}$ modelių ir jų nuotraukų poras galėjo pabloginti metodo persimokymo prie mokymo duomenų problemą, dėl 3D modelių pasikartojimo kartu su kiekviena jo nuotrauka mokymo metu. Todėl mokinant

HA-GAN metodą, buvo pakeistas duomenų surinkimo būdas – objektai surenkami į $\{x_i^H, x_i^L, Y_i\}$ grupes, kur x_i^H yra $256 \times 256 \times 256$ raiškos objekto modelis, x_i^L – $64 \times 64 \times 64$ raiškos modelis, o Y_i yra visų objektą atvaizduojančių nuotraukų masyvas. Mokymo metu, koreguojant nustatymus, galima koduotoją kiekvienos iteracijos metu nuosekliai apmokinti su visomis arba su norimu skaičiumi atsitiktinai pasirinktų nuotraukų.

Visų tinklų apmokymui ir vertinimui, taip pat kaip ir su 3D-VAE-GAN metodu, naudotas 670 *ShapeNetSem* stalų duomenų rinkinys, pirmuosius 80 % priskiriant mokymo poaibiui ir likusius 20 % priskiriant rezultatų įvertinimo poaibiui. HA-GAN straipsnyje autoriai nurodo tokias geriausias mokymo parametrų reikšmes:

- Mokymo iteracijų (partijų) skaičius – 80000;
- Partijos dydis – 4;
- Naudojama ADAM optimizacija su $\beta_1 = 0$ ir $\beta_2 = 0{,}999$;
- Generatoriaus, diskriminatoriaus ir koduotojo mokymo sparta (angl. learning rate) lygi 10^{-4}, 10^{-4} ir 4×10^{-4} atitinkamai;
- Rekonsrukcijos netekčių svoriai $\lambda_1 = 5$ ir $\lambda_2 = 5$.

Atliekant bandymus su įgyvendintu HA-GAN metodu pastebėta, kad mokymas vyksta daug stabiliau, nei 3D-VAE-GAN metode. Nėra staigių šokinėjimų tarp vieno ar kito tinklo dominavimo, tačiau pasirinkus netinkamus mokymo nustatymus galiausiai vienas iš neuroninių tinklų vis tiek pradeda atitrūkti nuo kito. Dažniausiai metodo diskriminatorių ir generatorių varžybose, diskriminuojantysis tinklas pradeda ženkliai pirmauti ir jo priešininkas nebegali atsigauti, tad siekiant geriausių rezultatų buvo mėginamos įvairios mokymo nustatymų kombinacijos.

Apart mokymo spartos pasirinkimo, HA-GAN autorių pateiktame metodo programiniame kode taip pat įgyvendinta galimybė per kiekvienos partijos apdorojimą atlikti daugiau nei vieną generatoriaus iteraciją. Kadangi magistro baigiamojo darbo metu buvo naudojami daug paprastesni trimačiai modeliai, diskriminatoriai daug lengviau atskiria tikruosius modelius nuo sugeneruotų. Du kartus apmokant generatorių tinklus po kiekvieno diskriminatorių apmokymo gaunama daug konkurencingesnė jų tarpusavio kova, ypač kartu padidinus ir generatorių mokymo spartą.

Taip pat pastebima, kad koduotojo mokymas yra visiškai atskirtas nuo generatoriau mokymo. Pagal autorių pateiktas išvadas, tai akivaizdžiai neigiamai nepaveikė jų galutinių rezultatų kokybės, tačiau testuojant šį mokymo eiliškumą su turimais *ShapeNetSem* stalų duomenis ir dvimačiu koduotoju rasta, kad toks mokymo eiliškumas nėra tinkamas. Kartu ir dėl Kullback-Leiber nuokrypio trūkumo, koduotojas užstringa prie svorių, kuriuos pritaikant su visomis nuotraukomis yra gaunamos lygiai tokios pačios reikšmės ir jos neatitinka generatorių mokymo metu naudojamo triukšmo vektoriaus tikimybinio pasiskirstymo, tad yra generuojami netinkami modeliai arba tuščios koordinačių erdvės. Išmėginti šių problemų sprendimo būdai aprašomi rezultatų skiltyje.

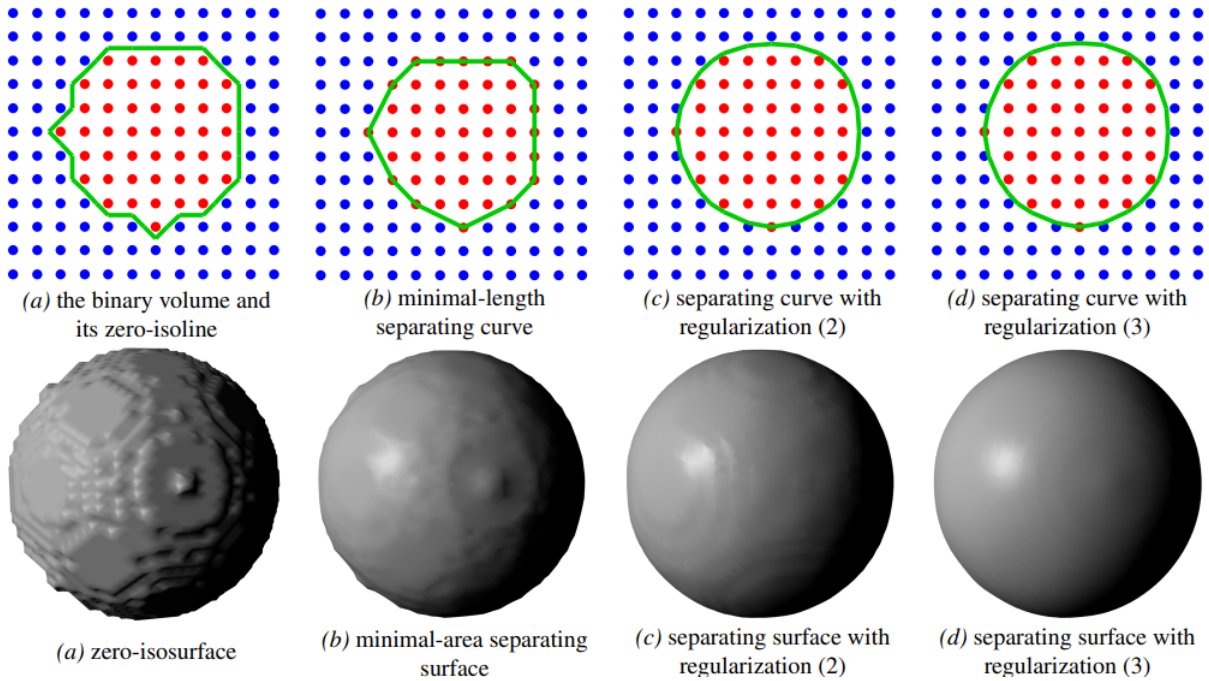
3.5. Sugeneruotų modelių papildomas apdorojimas

Atlikus 3D modelių generavimą su GAN metodais, norint patobulinti gautų rezultatų išvaizdą buvo pritaikomi papildomo apdorojimo (angl. post-processing) žingsniai. Magistro baigiamojo darbo metu pritrūko laiko išsamiai išnagrinėti ir išbandyti visus vokselinių modelių išvaizdos tobulinimo metodus, tad buvo apsiribota prie kelių nesudėtingų žingsnių.

Generuojant trimačius modelius su įgyvendintais metodais, yra gaunama trimatė matrica, kiekvienoje koordinatėje laikanti generatoriaus tinklo apskaičiuotą tikimybę, kad toje vietoje turi būti padedamas vokselis. Nustačius žemesnę slenksčio reikšmę, virš kurios vokseliai bus pavaizduojami, dažnai gaunami labiau užpildyti trimačio modelio variantai, tačiau taip pat gaunama daugiau ”triukšmo” – mažų objektui nepriklausančių dalių. Modelio išvalymas tokiais atvejais buvo atliekamas randant didžiausią sujungtą objektą arba panaikinant išskirties objektus, mažesnius nei tam tikras nusistatytas dydis.

Tai buvo atlikta su atvirai prieinama, atviro kodo *Python* biblioteka *”connected-components-3d”*, kuri gali rasti sujungtus objektus trimatėje matricoje tikrinant iki 26 aplink kiekvieną vokslelį esamų koordinačių ($3 \times 3 \times 3$ kubas aplink vokselį). Toks paieškos dydis yra tinkamas 64^3 raiškos modeliams, tačiau geriausiems rezultatams su 256^3 raiškos generuojamiems labiau būtų tinkamas didesnis.

Atlikus 3D modelio apvalymą, tolimesniam jo tobulinimui gali būti pritaikomas trimačių objektų išlyginimas. Kadangi didžioji dauguma kompiuterinės grafikos srityje naudojamų išlyginimo algoritmų yra pritaikyti tinkleliu pagrįstiems 3D modeliams, pirmasis žingsnis buvo tokio objekto pavaizdavimo išgavimas iš vokselinių duomenų. Tai buvo pasiektą naudojant 1987 metais William



17 pav. Victor Lempitsky straipsnyje [Lem10] pasiūlyto išlyginimo algoritmo pritaikymas dvimatėje ir trimatėje erdvėje. *a* – Su žygiuojančių kubų algoritmu gautas modelio tinklelis be išlyginimo, (c,d) - rezultatai gauti pritaikant siūloma išlyginimo metodą.

Loransen ir kt. straipsnyje pateiktu žygiuojančių kūbų (angl. *marching cubes*) metodą [LC87], kurio įgyvendinimui taip pat pasitelkta papildoma *Python* biblioteka – "*PyMCubes*".

Šiame pakete taip pat pateikiamas ir Victor Lempitsky 2010 metų straipsnyje pasiūlytas išlyginimo algoritmas [Lem10]. Transformacijos iš vokselių į tinklelį metu, vokselinė informacija yra pakeičiama į nuolatinių verčių įterpimo funkciją (angl. embedding function) nusakančią išlygintą, tačiau turinę informaciją išlaikančią modelio reprezentaciją. Pavyzdinius metodo pritaikymus galima pamatyti 17 paveikslėlyje.

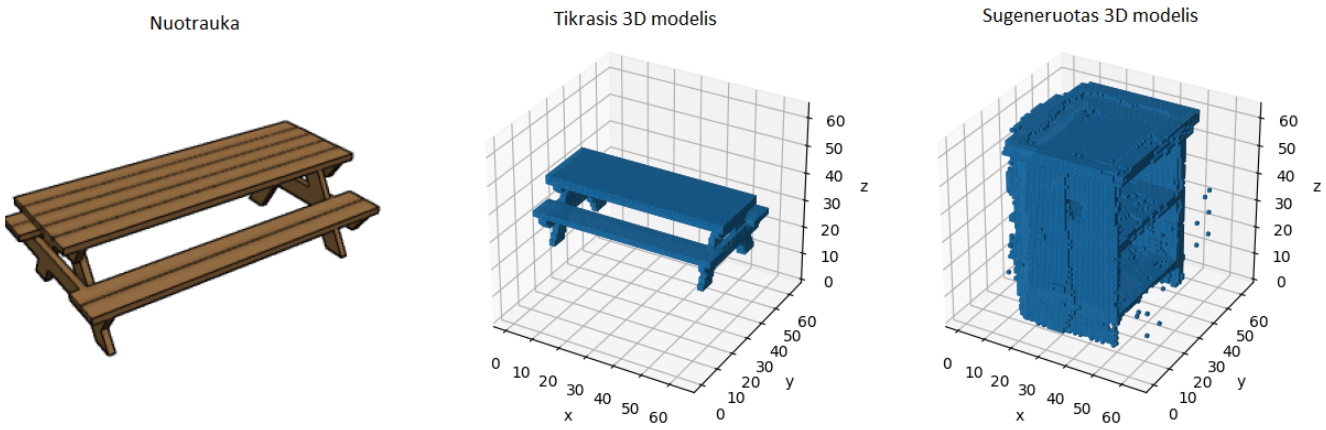
4. Atlikti bandymai ir rezultatai

Šiame skyriuje plačiau aprašomi praeitoje skiltyje pateiktų metodų vertinimui atlikti bandymai bei gauti rezultatai, parodomi metodų privalumai ir trūkumai, bei tendencijos, pagal kurias galėtų būti atliekami tolimesni tobulinimai.

4.1. Rezultatai gauti su 3D-VAE-GAN

Per visą magistro baigiamojo darbo trukmę buvo atliktas labai didelis kiekis įvairiausių bandymų išgauti, priimtinus rezultatus iš 3D-VAE-GAN metodo. Buvo padaryta daug klaidų, kurios atsirado dėl detalių apie konkrečius dirbtinių neuroninių tinklų įgyvendinimus Wu ir kt. straipsnyje [WZX+16], tačiau jos visos pažingsniui buvo ištaisomos ir pagal metodologijos skiltyje aprašomą įgyvendinimą pagaliau gauti trimačio skaitmeninimo rezultatai, nors ir turintys vietos tobulėjimui.

Pirmasis dalinai sėkmingas bandymas sugeneruoti nuotraukose matomų objektų suskaitmenintus trimačius modelius parodė klaidą programos kode, tačiau taip pat parodė ir kaip lengvai šis metodas gali persimokyti su mokymo duomenimis. Buvo nepastebėta, kad objektai yra neteisingai suporuojami su juos atvaizduojančiomis nuotraukomis ir atlikus 100 mokymo epochų su generatoriaus, diskriminatoriaus ir koduotojo mokymo sparta nustatyta kaip $0{,}0025$, 5×10^{-5} ir $0{,}001$ atitinkamai bei partijos dydžiu – 64. Gauti rezultatai matomi 18 paveikslėlyje, nors paduodama lauko stalo nuotrauka, sugeneruojamas naktinio staliuko (spintelės) modelis. Koduotojo išgautos savybes nenusako nieko realaus apie jam pateikiamą nuotrauką, metodas tiesiog atsimena mokymo metu matytas figūras ir bando jas kuo tiksliau atkurti.

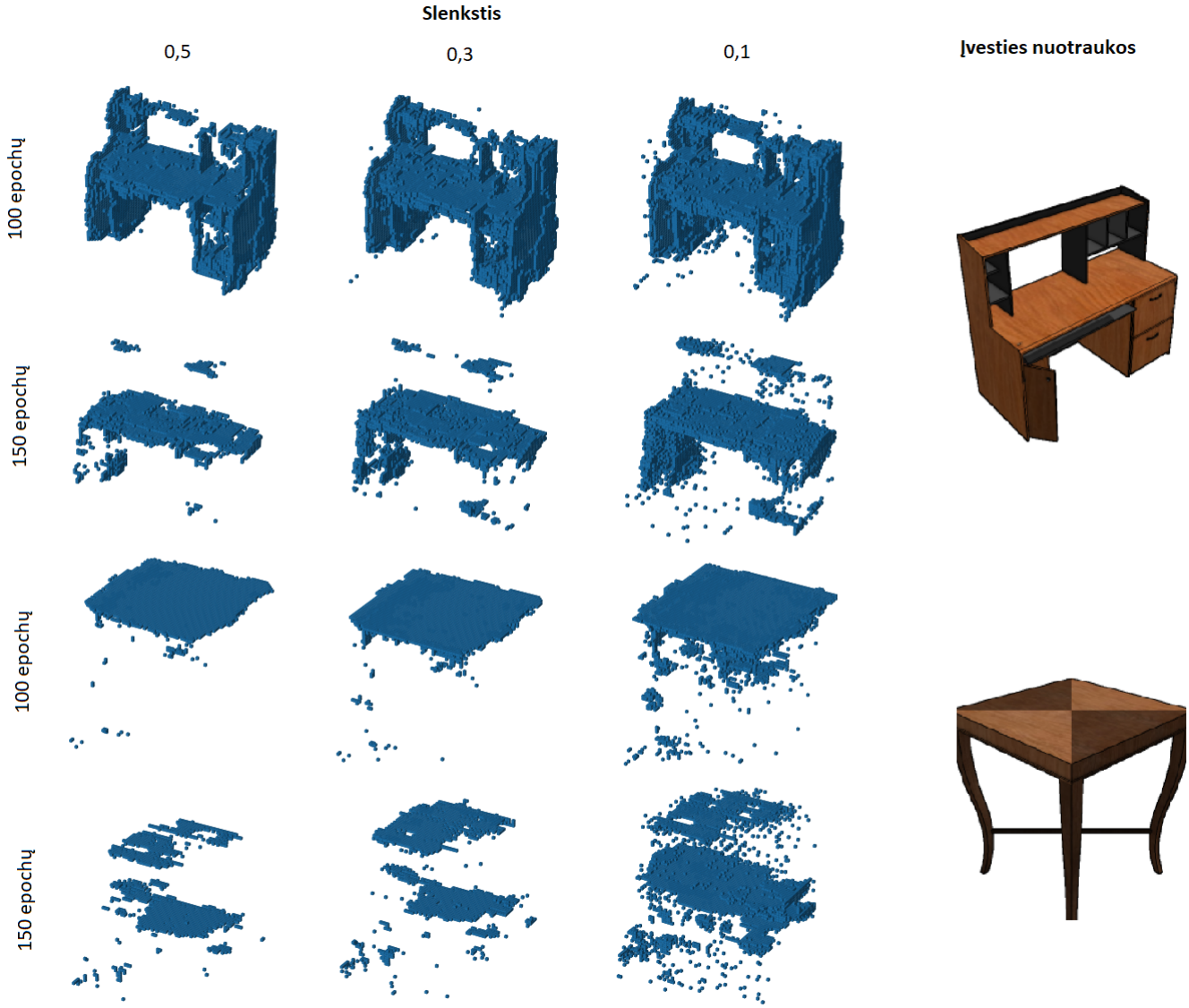


18 pav. 3D-VAE-GAN rezultatai gauti klaidingai suporavus objektų nuotraukas ir 3D modelius. Kairėje generavimui paduodama nuotrauka, viduryje tikrasis objektas trimatėje erdvėje, o dešinėje metodo sugeneruotas trimatis objektas (tinklų mokimo metu dar nebuvo sutvarkytas modelių pasukimo kampas).

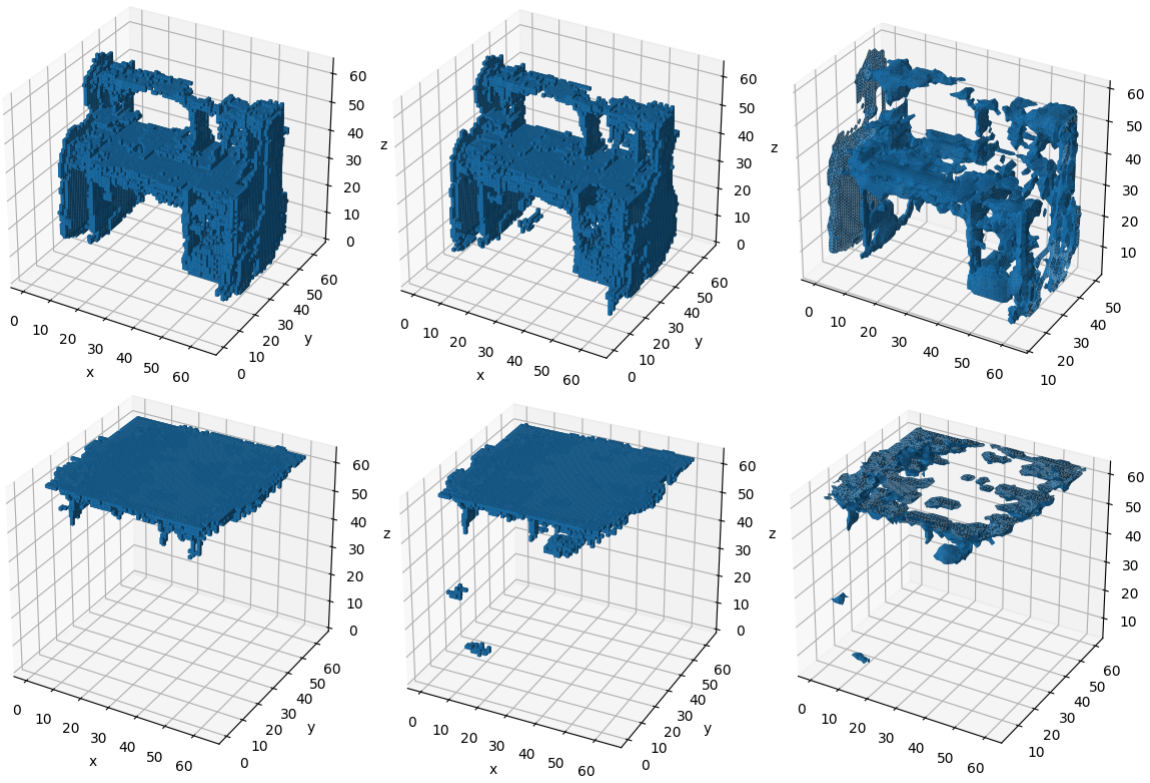
Pataisius duomenų poravimą ir pasukimą, toliau buvo atliekami bandymai su įvairiais mokymo spartos bei α_1 ir α_2 reikšmėmis, tačiau kaip ir su daugelių, ypač ankstyvųjų, GAN algoritmų, mokymo eiga beveik visada buvo labai nestabili. Pirmi priimtini rezultatai buvo gauti su generatoriaus, diskriminatoriaus ir koduotojo mokymo sparta nustatyta kaip $0{,}0021$, 2×10^{-5} ir $0{,}001$

atitinkamai ir $\alpha_1 = 5$ ir $\alpha_2 = 5 \times 10^{-4}$. Tačiau, geriausi rezultatai gauti pakeitus generatoriaus ir diskriminatoriaus mokymo spartas į $0{,}0025$ ir $3{,}3 \times 10^{-5}$.

Su šiais nustatymais gautų rezultatų pavyzdžius galima pamatyti 19 paveikslėlyje. Metodas gana neblogai atkuria mokymo duomenyse sutinkamus stalus, tačiau retai sėkmingai atkuria duomenis iš testavimo duomenų poaibio. Visų atliktų mokymų metu buvo stebima, kad nuostolio funkcijų rezultatai tikrinant tinklus pagal testavimo duomenis nerodė pastovios nusileidimo tendencijos, ir dažnai svyruodavo tarp geresnių ar blogesnių reikšmių. Taip pat, kaip ir matoma paveikslėlyje, didesnis epochų kiekis, po tam tikro kiekio, pradeda mažinti generuojamų modelių tikslumą, nors rekonstrukcijos nuostolio funkcija toliau rodė pastovų mažėjimą pagal mokymo duomenis. Bandymuose buvo sukurti tiktai 100 ir 150 epochų variantai, tad optimaliausias epochų skaičius nėra



19 pav. Įgyvendinto 3D-VAE-GAN metodo rezultatai skaitmeninant *ShapeNetSem* stalų objektus generatoriaus, diskriminatoriaus ir koduotojo mokymo spartas nustatant kaip $0{,}0025$, $3{,}3 \times 10^{-5}$ ir $0{,}001$. Pirmosios dvi eilutės rodo objekto skaitmeninimą iš mokymo duomenų poaibio, o sekančios – iš testavimo duomenų poaibio. Slenksčio reikšmės panaudojamos konvertuojant sugeneruotą modelį į dvimatį vokselinį formatą – visos reikšmės, kurios yra didesnės arba lygios slenksčiui, skaitomos kaip tiesos reikšmės.



20 pav. Įgyvendinto 3D-VAE-GAN metodo rezultatai gauti su slenksčiu 0.1, ištraukiant didžiausią sujungtą komponentą, panaikinant mažus komponentus, bei išlyginant geriausius iš pirmų dviejų žingsnių gautus rezultatus.

žinomas.

Gautiems rezultatams taip pat buvo pritaikyti metodologijos skiltyje aprašomi papildomo apdorojimo metodai. 20 paveikslėlyje matomi rezultatai gauti ištraukus didžiausią sujungtą komponentą, panaikinus komponentus, kurie susideda iš mažiau nei 30 vokselių, bei išlyginus 3D modelį. Matoma, kad pirmieji du trimačio modelio aptvarkymo metodai ženkliai patobulina jų išvaizdą, panaikinant nereikalingus išsišakojimus, ypač didžiausiojo sujungto komponento išgavimas, tačiau išlyginimo pritaikymas buvo nesėkmingas turimiems duomenims. Plonesni bei tiesūs modelio elementai yra panaikinami, paliekant dideles skyles, ir sumažinant modelio atitikimą tikrajam objektui. Jeigu šia tema būtų atliekami tolimesni darbai, galimai pritaikius vieną iš literatūros analizėje apžvelgtų Laplaso išlyginimo algoritmų būtų gaunami geresni rezultatai.

Kaip lengviausiai intuityviai suprantamas kiekybinis metodo įvertinimas buvo skaičiuojamas rekonstrukcijos tikslumas. 7 lentelėje surašomos reikšmės nusako koks procentas visų sugeneruoto modelių koordinačių yra priskiriamos teisingai. Rezultatai gaunami naudojant formulę (8), kur x_i yra objekto trimatis modelis, o y_i yra modelio nuotrauka. Vertinant rezultatus taip pat reikia atsižvelgti į tai, kad visiškai tuščia erdvė, neturinti nei vieno vokselio, turės maždaug 89 % tikslumą, todėl esminiai skirtumai tarp gaunamų rezultatų yra pastebimi tarp 90 ir 100 %. Iš pateiktų tikslumo reikšmių matoma, kad nereikalingų vokselių komponentų panaikinimo metodų pritaikymas ne visada patobulina gaunamu rezultatų kokybę. didžiausi privalumai yra matomi pasirinkus labai mažą slenksčio reikšmę, kuomet trimačiuose vaizduose yra daugiau triukšmo, kurį pravartu panaikinti.

$$1 - ||x_i - G(y_i)||_1 \tag{8}$$

7 lentelė. Įgyvendinto 3D-VAE-GAN metodo procentinis rekonstrukcijos tikslumas (imamas vidurkis iš trijų bandymų.

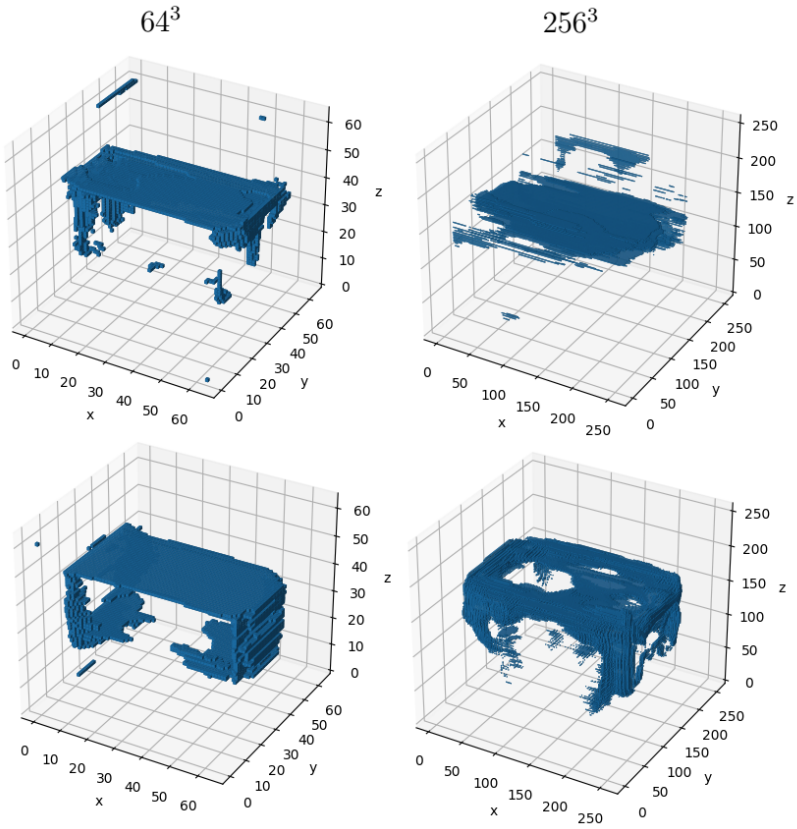
	Skaitmeninimo tikslumas procentais			
	Mokymo duomenys		**Testavimo duomenys**	
	100 epochų	150 epochų	100 epochų	150 epochų
Slenksnis = 0.3				
Be apdirbimo	96,6727	95,5425	94,4338	94,6541
Didžiausias sujungtas komponentas	96,6288	95,5764	94,5802	94,7542
Išskirčių panaikinimas ($<$ 30 vokselių)	96,6740	95,5607	94,4714	94,6855
Išskirčių panaikinimas ($<$ 60 vokselių)	96,6662	95,5568	94,4851	94,7058
Slenksnis = 0.1				
Be apdirbimo	96,4915	95,4089	93,7897	94,4447
Didžiausias sujungtas komponentas	96,5230	95,4821	93,9982	94,6085
Išskirčių panaikinimas ($<$ 30 vokselių)	96,5238	95,4546	93,8454	94,5324
Išskirčių panaikinimas ($<$ 60 vokselių)	96,5248	95,4557	93,8790	94,5445

4.2. Rezultatai gauti su HA-GAN

Įgyvendinant HA-GAN metodą nespėta atlikti tokio pačio kiekio eksperimentų kaip su 3D-VAE-GAN, tad nebuvo sukurtas tinkamus trimačius modelius generuojančio rezultatas. Pirmiausia buvo atliekami bandymai su straipsnyje [SCX+22] pateiktais nustatymais – generatorių, diskriminatorių ir koduotojo mokymo spartas nustatant kaip 10^{-4}, 10^{-4} ir 4×10^{-4} atitinkamai. Su tokiais mokymo nustatymais jau po kelių pirmųjų epochų diskriminatoriaus tinklai išsiveržią į priekį, o generatorių tinklai palaipsniui nustoja tobulėti, tačiau lyginant su 3D-VAE-GAN mokymu progresas iškomas daug ilgiau. Diskriminatoriaus dominavimas galimai atsitinka dėl darbo metu naudojamo duomenų rinkinio palygintino paprastumo. Originaliai HA-GAN metodas yra skirtas medicininių vaizdų generavimui, kuriuose daug didesnė trimatės erdvės dalis yra užpildoma vokseliais. Tokius vaizdus yra ganėtinai sudėtingiau atskirti nuo sugeneruotų, tad ir nustatymai yra pritaikomi atitinkamai.

Atsižvelgiant į tai, buvo bandoma pakoreguoti tinklų nustatymus pagal turimus duomenis, didinant generatorių arba mažinant diskriminatorių mokymo spartą, bei atliekant kelias generatorių iteracijas kiekvienai diskriminatorių iteracijai. Ilgiausiai besivaržančiųjų tinklų kovą išlaikantis variantas, rastas bandymų metu, buvo gautas pakėlus generatorių mokymo startą iki 4×10^{-4} ir atliekant po dvi jų iteracijas kiekvienai partijai.

Nors besivaržančiųjų tinklų tarpusavio kova išlikdavo pakankamai pastovi, visuose atliktuose bandymuose su daugiau nei 20 epochų buvo susiduriama su generatorių ir koduotojo modos griūtimi. Su visomis į metodą paduodamomis įvestimis buvo gaunama ta pati išvestis (jeigu išlaikomi tos pačios iteracijos svoriai). Kadangi koduotojas su visomis nuotraukomis išgaudavo lygiai tokį pati triukšmo vektorių, taip pat buvo gaunamas ir lygiai toks pat trimatis modelis, be jokios aiškios



21 pav. Du HA-GAN generuojamų objektų pavyzdžiai, kairėje – žemos raiškos 64^3 modeliai, dešinėje tie patys modeliai generuojami su G^H gaunant 256^3 raiškos objektą.

formos, o generatorius gebėjo atkurti tiktai dvi skirtingas išvestis – vieną su koduotojo paduodamu triukšmo vektoriumi ir vieną su atsitiktinai kuriamu vektoriumi, kurio reikšmių vidurkis yra 0, o dispersija 1.

Pastebėjus šias tendencijas buvo bandoma spręsti problemą pridedant Kullback-Leiber nuokrypio nuostolį prie koduotojo tinklo bei rekonstrukcijos nuostolio funkciją prie generatoriaus tinklų. Pritrūkus laiko tinkamai pritaikyti koduotojo tinklą prie KL nuokrypio, buvo bandoma tiesiogiai lyginti gautus triukšmo vektorius su atsitiktiniais vektoriais naudojamais generatoriaus mokyme pritaikant paprastąją *TensorFlow* teikiamą nuostolio funkciją. Tai nedavė gerų rezultatų ir reikšmės kartais netgi patapdavo neigiamos, kas neturėtų būti įmanoma pagal teorinį KL nuokrypio apibrėžimą. Tuo tarpu rekonstrukcijos netektis prie generatoriaus tinklų visais bandymais užstrigdavo prie būsenos, kurioje, atliekant nuotraukoje matomų objektų skaitmeninimą, buvo gaunama tuščia trimatė erdvė. Tas pats buvo stebima rekonstrukcijai naudojant ir vidutinės absoliučiosios paklaidos, ir vidutinės kvadratinės paklaidos, ir kvadratinio Euklido atstumo formules (bent naudojant svorius, kuriuos buvo spėta išmėginti).

21 paveikslėlyje matomi dviejų sugeneruotų trimačių modelių aukštos ir žemos raiškos variantai, gauti nepaduodant jokios nuotraukos ir leidžiant generatoriui kurti išvestį pagal atsitiktinių reikšmių triukšmo vektorių. Abu rezultatai gauti užkraunant skirtingų epochų svorius, todėl jie nėra vienodi. Šie rezultatai parodo, kad Li Sun ir kt. siūlomas išsišakojęs GAN modelis mokymo metu tinkamai perduoda viso objekto ypatybes iš žemos raiškos generatoriaus G^L į aukštos raiškos generatorių G^H, kadangi abejomis raiškomis yra kuriamas panašios formos modelis.

5. Išvados

Trimatis skaitmeninimas yra plačiai pritaikoma technologija, nuo skaitmeninių pramogų srities iki kultūrinio paveldo išsaugojimo. Literatūros analizėje buvo apžvelgta dalis 3D skaitmeninimo metodų, koncentruojantis į GAN tinklų taikymą, tačiau apžvelgiant ir tradicinius metodus.

Analizuojant trimačių modelių skaitmeninimui naudojamų, generatyviniais besivaržančiais neuroniniais tinklais pagrįstus, metodus bei jų patobulinimus, atlikta pirmoji išsikelta užduotis ir įgyvendintas veikiantis 3D modelių skaitmeninimo algoritmas. Atliekant antrąją užduoti buvo imami pirmosios vykdymo metu atrasti bei panaudoti sprendimai ir, pamodifikavus vieną naujausiųjų trimačių modelių generavimo metodų, suformuluotas naujas skaitmeninimo iš vienos nuotraukos metodas, tačiau nebuvo spėta pilnai įrodyti naujojo metodo veiksmingumo. Galiausiai, atliekant trečiąją užduotį, pritaikytas išlyginimo algoritmas generuojamų objektų kokybės tobulinimui.

Atlikus tyrimus gautos išvados:

- Trimačių modelių generavimo ir skaitmeninimo iš dvimačių nuotraukų užduotis yra itin sudėtingas iššūkis šiuolaikiniams dirbtinio intelekto metodams bei kompiuterinei įrangai. Net bandant atkartoti jau sukurtus metodus dažnai susiduriama su kliūtimis prie įvairiausių implementacijos detalių, kaip duomenų rinkinio pasirinkimo ar jo išankstinio apdorojimo, mažo aukštos kokybės, lengvai prieinamų, duomenų pasirinkimo, konkretaus nuostolio funkcijų skaičiavimo ir t.t. Gerų rezultatų išgavimas reikalauja daug laiko eksperimentavimui su skirtingais mokymo nustatymais ar konkretaus įgyvendinimo detalėmis, tačiau netgi ir geriausi literatūros analizėje apžvelgti metodai dar vis nėra tinkami pritaikyti realiuose panaudojimo atvejuose.

- Įgyvendinant pirmąjį trimačio skaitmeninimo metodą – 3D-VAE-GAN [WZX+16] susidurta su dideliu kiekiu iššūkių, išbandytos įvairiausios neuroninių tinklų ir reikalingų nuostolio funkcijų įgyvendinimo variacijos bei duomenų rinkinio išankstiniai apdorojimai. Galiausiai gautas metodas gebantis visai neblogai atkurti mokymosi duomenyse matytus modelius, tačiau dar nevisai galintis tinkamai sugeneruoti nematytų modelių. Kad išspręsti šią problemą turėtų būti atliekami tolimesni darbai pakeičiant duomenų surikimo į objektų ir nuotraukų poras būdą, netekties svorių bei mokymo spartos pasirinkimą ir (arba) pritaikant kituose GAN srityse arba naujesniuose straipsniuose siūlomus mokymo tobulinimus.

- Lyginant senąjį 3D-VAE-GAN ir naująjį HA-GAN [SCX+22] metodus rastas ženklus patobulėjimas besivaržančių tinklų mokymosi stabilume bei generuojamų modelių raiškoje, tačiau sutinkamos kitos kliūtys. Nenaudojant Kullback-Leiber netekties funkcijos HA-GAN mokymo metodikoje, prarandamas koduotojo apsiribojimas prie generatoriui naudingo tikimybinio triukšmo vektoriaus pasiskirstymo, o koduotojo ir generatoriaus mokymo atskyrimas lemia, kad abu tinklai dažnai kenčia nuo modos griūties (angl. *mode collapse*) kuomet su visais įvesties variantais yra gaunama ta pati išvestis. Tačiau, atradus tinkamus tinklo nustatymus ir optimizavus dvimačių nuotraukų koduotojo tinklą, siūlomas algoritmas gali tapti vienas iš geriausių trimačio skaitmeninimo metodų.

- Iš riboto skaičiaus atliktų trimačių modelių tobulinimo bandymų galima tarti, kad tinkamiems rezultatams gauti neužtenka panaudoti bet kokį išlyginimo algoritmą, jis turi būti pritaikytas prie konkretaus panaudojimo ir turimų modelių formos. Išbandytas išlyginimo metodas [Lem10] galimai būtų sėkmingas apvalesnių objektų modelių tobulinimui, tačiau pritaikant prie darbe naudoto stalų duomenų rinkinio jis buvo visiškai netinkamas, kadangi yra panaikinamos didelės modelių dalys ir paliekamos skyles jų vietoje.

Literatūra

[AOV12] Alexandre Alahi, Raphael Ortiz ir Pierre Vandergheynst. FREAK: Fast Retina Keypoint. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, p.p. 510–517, 2012. DOI: 10.1109/CVPR.2012.6247715. URL: https://www.researchgate.net/publication/258848394_FREAK_Fast_retina_keypoint.

[BAC+18] Miles Brundage, Shahar Avin, Jack Clark, Helen Toner ir k.t. The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation. *ArXiv*, abs/1802.07228, 2018. URL: https://arxiv.org/abs/1802.07228.

[BCL21] Antonio Bacciaglia, Alessandro Ceruti ir Alfredo Liverani. Surface smoothing for topological optimized 3D models. *Structural and Multidisciplinary Optimization*, 64, 2021-12. DOI: 10.1007/s00158-021-03027-6. URL: https://www.researchgate.net/publication/353966431_Surface_smoothing_for_topological_optimized_3D_models.

[BET+08] Herbert Bay, Andreas Ess, Tinne Tuytelaars ir Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2007.09.014. URL: https://people.ee.ethz.ch/~surf/eccv06.pdf. Similarity Matching in Computer Vision and Multimedia.

[Bul] Yaroslav Bulatov. Saving memory using gradient-checkpointing - GitHub. URL: https://github.com/cybertronai/gradient-checkpointing. Accessed: 2023-01-09.

[CFG+15] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan ir k.t. ShapeNet: An Information-Rich 3D Model Repository. (arXiv:1512.03012 [cs.GR]), 2015.

[CSS+02] D. Chetverikov, D. Svirko, D. Stepanov ir P. Krsek. The Trimmed Iterative Closest Point algorithm. *2002 International Conference on Pattern Recognition*, 545–548 vol.3, 2002. DOI: 10.1109/ICPR.2002.1047997.

[CXZ+16] Tianqi Chen, Bing Xu, Chiyuan Zhang ir Carlos Guestrin. Training Deep Nets with Sublinear Memory Cost, 2016. DOI: 10.48550/ARXIV.1604.06174. URL: https://arxiv.org/abs/1604.06174.

[FB81] Martin A. Fischler ir Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395, 1981. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: https://doi.org/10.1145/358669.358692.

[FP10] Yasutaka Furukawa ir Jean Ponce. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010. DOI: 10.1109/TPAMI.2009.161. URL: https://www.researchgate.net/publication/44683582_Accurate_Dense_and_Robust_Multiview_Stereopsis.

[GFR+16] Rohit Girdhar, David F. Fouhey, Mikel D. Rodriguez ir Abhinav Kumar Gupta. Learning a Predictable and Generative Vector Representation for Objects. *ArXiv*, abs/1603.08637, 2016. URL: https://arxiv.org/abs/1603.08637.

[GPM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville ir Yoshua Bengio. Generative Adversarial Nets. Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence ir K. Q. Weinberger, redaktoriai, *Advances in Neural Information Processing Systems*, tom. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[GWM18] Matheus Gadelha, Rui Wang ir Subhransu Maji. Multiresolution Tree Networks for 3D Point Cloud Processing, 2018. arXiv: 1807.03520. URL: https://arxiv.org/abs/1807.03520.

[Har97] R.I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, 1997. DOI: 10.1109/34.601246. URL: http://www.cs.cmu.edu/afs/andrew/scs/cs/15-463/f07/proj_final/www/amichals/fundamental.pdf.

[HF19] Paul Henderson ir Vittorio Ferrari. Learning single-image 3D reconstruction by generative modelling of shape, pose and shading. *CoRR*, abs/1901.06447, 2019. arXiv: 1901.06447. URL: http://arxiv.org/abs/1901.06447.

[Hir05] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, tom. 2, 807–814 vol. 2, 2005. DOI: 10.1109/CVPR.2005.56. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.8897&rep=rep1&type=pdf.

[Hop08] Hugues Hoppe. Poisson Surface Reconstruction and Its Applications. *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*. Association for Computing Machinery, 2008. ISBN: 9781605581064. DOI: 10.1145/1364901.1364904. URL: https://doi.org/10.1145/1364901.1364904.

[HS88] C Harris ir M Stephens. A combined corner and edge detector. proc. of the 4th Alvey Vis. Conf, 1988. URL: http://www.bmva.org/bmvc/1988/avc-88-023.pdf.

[HZR+15] Kaiming He, Xiangyu Zhang, Shaoqing Ren ir Jian Sun. Deep Residual Learning for Image Recognition, 2015. DOI: 10.48550/ARXIV.1512.03385. URL: https://arxiv.org/abs/1512.03385.

[YCH+13] Ming-Der Yang, Chih-Fan Chao, Kai-Siang Huang, Liang-You Lu ir Yi-Ping Chen. Image-based 3D scene reconstruction and exploration in augmented reality. *Automation in Construction*, 33:48–60, 2013. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2012.09.017. URL: https://www.sciencedirect.com/science/article/pii/S0926580512001690. Augmented Reality in Architecture, Engineering, and Construction.

[IZZ+17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou ir Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p.p. 5967–5976, 2017. DOI: 10.1109/CVPR.2017.632. URL: https://arxiv.org/abs/1611.07004.

[KAL+18] Tero Karras, Timo Aila, Samuli Laine ir Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *International Conference on Learning Representations*, 2018. URL: https://openreview.net/forum?id=Hk99zCeAb.

[KKR18] Vladimir A. Knyaz, Vladimir V. Kniaz ir Fabio Remondino. Image-to-Voxel Model Translation with Conditional Adversarial Networks. *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018-09. URL: https://openaccess.thecvf.com/content_eccv_2018_workshops/w6/html/Knyaz_Image-to-Voxel_Model_Translation_with_Conditional_Adversarial_Networks_ECCVW_2018_paper.html.

[KL17] Ali Khaloo ir David Lattanzi. Hierarchical Dense Structure-from-Motion Reconstructions for Infrastructure Condition Assessment. *Journal of Computing in Civil Engineering*, 31(1):04016047, 2017. DOI: 10.1061/(ASCE)CP.1943-5487.0000616. URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000616.

[KRK19] Vladimir Kniaz, Fabio Remondino ir V. Knyaz. GENERATIVE ADVERSARIAL NETWORKS FOR SINGLE PHOTO 3D RECONSTRUCTION. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W9:403–408, 2019-01. DOI: 10.5194/isprs-archives-XLII-2-W9-403-2019. URL: https://www.researchgate.net/publication/330766571_GENERATIVE_ADVERSARIAL_NETWORKS_FOR_SINGLE_PHOTO_3D_RECONSTRUCTION.

[KW13] Diederik P Kingma ir Max Welling. Auto-Encoding Variational Bayes, 2013. DOI: 10.48550/ARXIV.1312.6114. URL: https://arxiv.org/abs/1312.6114.

[LC87] William E. Lorensen ir Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. SIGGRAPH '87:163–169, 1987. DOI: 10.1145/37401.37422. URL: https://doi.org/10.1145/37401.37422.

[LCS+16] Yu-Fei Liu, Soojin Cho, B. F. Spencer ir Jian-Sheng Fan. Concrete Crack Assessment Using Digital Image Processing and 3D Scene Reconstruction. *Journal of Computing in Civil Engineering*, 30(1):04014124, 2016. DOI: 10.1061/(ASCE)CP.1943-5487.0000446. URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000446.

[Lem10] Victor Lempitsky. Surface extraction from binary volumes with higher-order smoothness:1197–1204, 2010. DOI: 10.1109/CVPR.2010.5539832.

[Low04] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[LPT13] Joseph J. Lim, Hamed Pirsiavash ir Antonio Torralba. Parsing IKEA Objects: Fine Pose Estimation. *2013 IEEE International Conference on Computer Vision*, p.p. 2992–2999, 2013. DOI: 10.1109/ICCV.2013.372. URL: https://people.csail.mit.edu/torralba/publications/ikea_iccv2013.pdf.

[LSL+15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle ir Ole Winther. Autoencoding beyond pixels using a learned similarity metric, 2015. DOI: 10.48550/ARXIV.1512.09300. URL: https://arxiv.org/abs/1512.09300.

[LSW15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby ir Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015. arXiv: 1512.09300. URL: http://arxiv.org/abs/1512.09300.

[LTH+17] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang ir Wenzhe Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*:105–114, 2017. URL: https://arxiv.org/abs/1609.04802.

[MAP+15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo ir k.t. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. URL: https://www.tensorflow.org/. Software available from tensorflow.org.

[MCL+18] Stefano Markidis, Steven Wei Der Chien, Erwin Laure, Ivy Bo Peng ir Jeffrey S. Vetter. NVIDIA Tensor Core Programmability, Performance & Precision, 2018. DOI: 10.1109/ipdpsw.2018.00091. URL: https://doi.org/10.1109%2Fipdpsw.2018.00091.

[MY09] Jean-Michel Morel ir Guoshen Yu. ASIFT: A New Framework for Fully Affine Invariant Image Comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009. DOI: 10.1137/080732730. URL: https://doi.org/10.1137/080732730.

[Min04] Patrick Min. binvox. http://www.patrickmin.com/binvox or https://www.google.com/search?q=binvox, 2004.

[MKK+18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama ir Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks, 2018. arXiv: 1802.05957 [cs.LG].

[ML18] Zhiliang Ma ir Shilong Liu. A review of 3D reconstruction techniques in civil engineering and their applications. *Advanced Engineering Informatics*, 37:163–174, 2018. ISSN: 1474-0346. DOI: https://doi.org/10.1016/j.aei.2018.05.005. URL: https://www.sciencedirect.com/science/article/pii/S1474034617304275.

[Nis04] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, 2004. DOI: 10.1109/TPAMI.2004.17. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.8769&rep=rep1&type=pdf.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer ir k.t. PyTorch: An Imperative Style, High-Performance Deep Learning Library:8024–8035, 2019. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[PHL+17] Ashok Kumar Patil, Pavitra Holi, Sang Keun Lee ir Young Ho Chai. An adaptive approach for the reconstruction and modeling of as-built 3D pipelines from point clouds. *Automation in Construction*, 75:65–78, 2017. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2016.12.002. URL: https://www.sciencedirect.com/science/article/pii/S0926580516304745.

[RD06] Edward Rosten ir Tom Drummond. Machine Learning for High-Speed Corner Detection. Aleš Leonardis, Horst Bischof ir Axel Pinz, redaktoriai, *Computer Vision – ECCV 2006*, p.p. 430–443, Berlin, Heidelberg. Springer Berlin Heidelberg, 2006.

[RFB15] Olaf Ronneberger, Philipp Fischer ir Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *CoRR*, abs/1505.04597, 2015. arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[RGL+14] Pablo Rodriguez-Gonzalvez, Diego Gonzalez-Aguilera, Gemma Lopez-Jimenez ir Inmaculada Picon-Cabrera. Image-based modeling of built environment from an unmanned aerial system. *Automation in Construction*, 48:44–52, 2014. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2014.08.010. URL: https://www.sciencedirect.com/science/article/pii/S0926580514001897.

[SBM+15] Jūratė Sužiedelytė-Visockienė, Renata Bagdžiūnaitė, Naglis Malys ir Vida Maliene. Close-range photogrammetry enables documentation of environment-induced deformation of architectural heritage. *Environmental Engineering and Management Journal*, 14, 6, 2015. URL: http://eprints.nottingham.ac.uk/id/eprint/34653. Full text of article not available online. No doi. Permission given by journal publishers to deposit in repository. KJB 11.07.2016.

[SC13] Changhun Sung ir Myung Jin Chung. Multi-Scale Descriptor for Robust and Fast Camera Motion Estimation. *IEEE Signal Processing Letters*, 20(7):725–728, 2013. DOI: 10.1109/LSP.2013.2264672.

[SCH15] Manolis Savva, Angel X. Chang ir Pat Hanrahan. Semantically-Enriched 3D Models for Common-sense Knowledge. *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality*, 2015.

[SCX+22] Li Sun, Junxiang Chen, Yanwu Xu, Mingming Gong, Ke Yu ir Kayhan Batmanghelich. Hierarchical Amortized GAN for 3D High Resolution Medical Image Synthesis, 2022. DOI: 10.1109/JBHI.2022.3172976.

[SGZ+16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen ir Xi Chen. Improved Techniques for Training GANs. D. Lee, M. Sugiyama, U. Luxburg, I. Guyon ir R. Garnett, redaktoriai, *Advances in Neural Information Processing Systems*, tom. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.

[SKK15] Hyojoo Son, Changmin Kim ir Changwan Kim. Fully Automated As-Built 3D Pipeline Extraction Method from Laser-Scanned Data Based on Curvature Computation. *Journal of Computing in Civil Engineering*, 29(4):B4014003, 2015. DOI: 10.1061/(ASCE)CP.1943-5487.0000401. URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000401.

[Tau99] Gabriel Taubin. A Signal Processing Approach To Fair Surface Design. *Computer Graphics (Proceedings of Siggraph '95)*, 29, 1999-07. DOI: 10.1145/218380.218473. URL: https://www.researchgate.net/publication/2374153_A_Signal_Processing_Approach_To_Fair_Surface_Design.

[TMH+00] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley ir Andrew W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis. *Vision Algorithms: Theory and Practice*, p.p. 298–372, 2000. ISBN: 978-3-540-44480-0.

[WH18] Yuxin Wu ir Kaiming He. Group Normalization, 2018. arXiv: 1803.08494 [cs.CV].

[WZX+16] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman ir Joshua B. Tenenbaum. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *NIPS*, 2016. URL: https://arxiv.org/abs/1610.07584.

[ZHK+14] Stefanie Zollmann, Christof Hoppe, Stefan Kluckner, Christian Poglitsch, Horst Bischof ir Gerhard Reitmayr. Augmented Reality for Construction Site Monitoring and Documentation. *Proceedings of the IEEE*, 102:137–154, 2014-02. DOI: 10.1109/JPROC.2013.2294314. URL: https://www.researchgate.net/publication/261045453_Augmented_Reality_for_Construction_Site_Monitoring_and_Documentation.

[ZXL+17] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang ir Dimitris N. Metaxas. StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. *2017 IEEE International Conference on Computer Vision (ICCV)*:5908–5916, 2017. URL: https://arxiv.org/abs/1612.03242.

Priedas 1

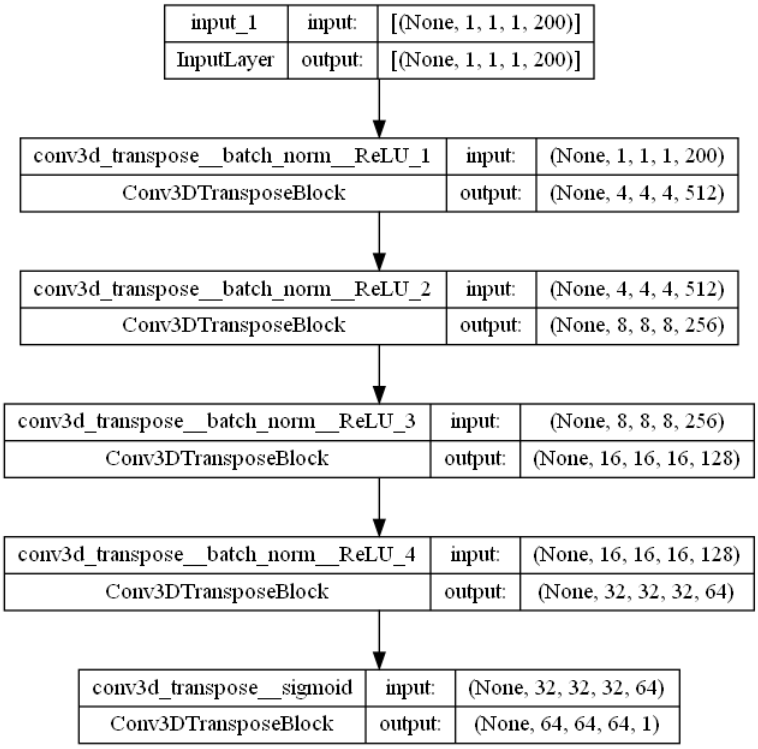
Vokselinių modelių kūrimas pagal OBJ formato duomenis. Norint panaudoti šį kodą buvo sukurintas CSV failas su stalo tipo objektų *ShapeNet* kodais, ir skaiciumi 90 laipsnių z ašies pasukimų reikalingų norint atsukti juos pasirinkta kryptimi:

```
while IFS="," read -r column1 column2; do
shape_code=${column1//[$'\t\r\n ']}
rotation=${column2//[$'\t\r\n ']}
resolution=256
if ! test -f "tables-binvox-$resolution/$shape_code.binvox"; then
    if [ $rotation == '1' ]; then
        ../binvox.exe -rotz -e -aw -cb -d $resolution "models-OBJ/$shape_code.obj"
        mv "models-OBJ/$shape_code.binvox" "tables-binvox-$resolution/$shape_code.binvox"
    elif [ $rotation == '2' ]; then
        ../binvox.exe -rotz -rotz -e -aw -cb -d $resolution "models-OBJ/$shape_code.obj"
        mv "models-OBJ/$shape_code.binvox" "tables-binvox-$resolution/$shape_code.binvox"
    elif [ $rotation == '3' ]; then
        ../binvox.exe -rotz -rotz -rotz -e -aw -cb -d $resolution "models-
OBJ/$shape_code.obj"
        mv "models-OBJ/$shape_code.binvox" "tables-binvox-$resolution/$shape_code.binvox"
    else
        ../binvox.exe -e -aw -cb -d $resolution "models-OBJ/$shape_code.obj"
        mv "models-OBJ/$shape_code.binvox" "tables-binvox-$resolution/$shape_code.binvox"
    fi
fi; done < Table_voxelization.csv
```

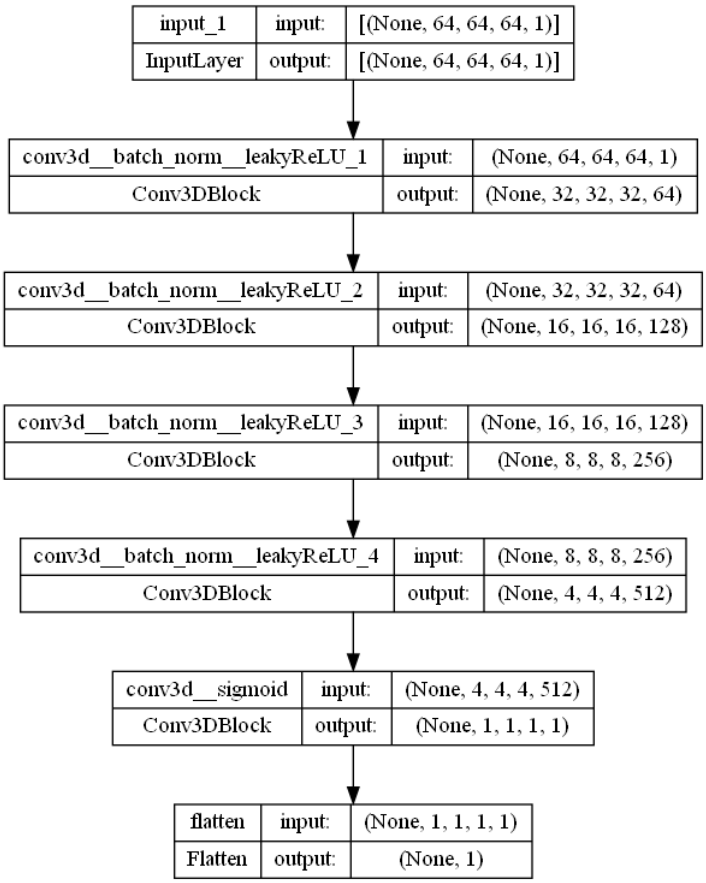
Priedas 2

3D-VAE-GAN neuroninių tinklų diagramos.

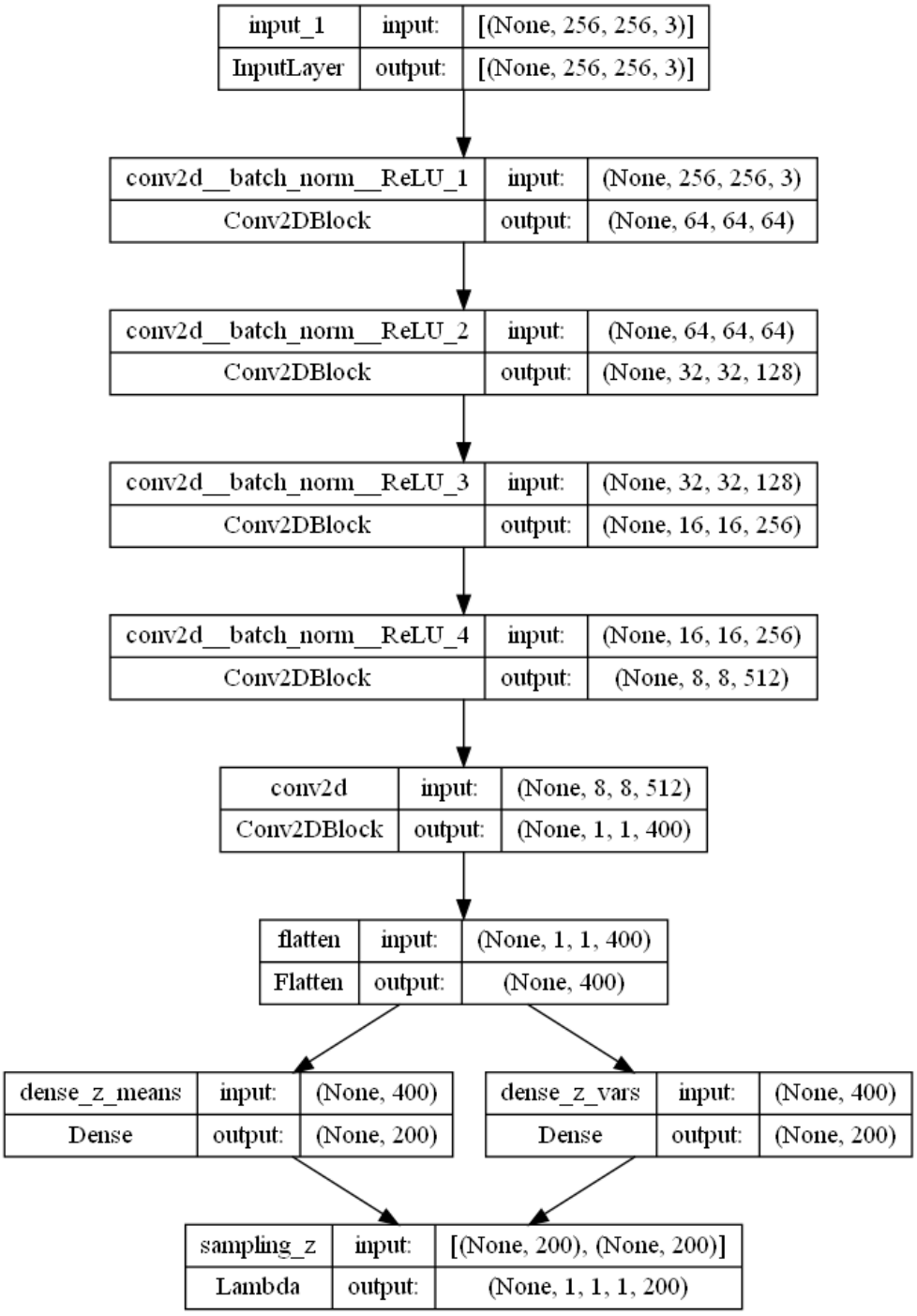
Generatorius:



Diskriminatorius:



Koduotojas:


input_1 | input: | [(None, 256, 256, 3)]
InputLayer | output: | [(None, 256, 256, 3)]
conv2d__batch_norm__ReLU_1 | input: | (None, 256, 256, 3)
Conv2DBlock | output: | (None, 64, 64, 64)
conv2d__batch_norm__ReLU_2 | input: | (None, 64, 64, 64)
Conv2DBlock | output: | (None, 32, 32, 128)
conv2d__batch_norm__ReLU_3 | input: | (None, 32, 32, 128)
Conv2DBlock | output: | (None, 16, 16, 256)
conv2d__batch_norm__ReLU_4 | input: | (None, 16, 16, 256)
Conv2DBlock | output: | (None, 8, 8, 512)
conv2d | input: | (None, 8, 8, 512)
Conv2DBlock | output: | (None, 1, 1, 400)
flatten | input: | (None, 1, 1, 400)
Flatten | output: | (None, 400)
dense_z_means | input: | (None, 400)
Dense | output: | (None, 200)
dense_z_vars | input: | (None, 400)
Dense | output: | (None, 200)
sampling_z | input: | [(None, 200), (None, 200)]
Lambda | output: | (None, 1, 1, 1, 200)