SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1428

IZRADA I ANALIZA ODŠUMLJIVAČA ZA SLIKE GENERIRANE POSTUPKOM PRAĆENJA PUTA

Davor Najev

Zagreb, lipanj 2026.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1428

IZRADA I ANALIZA ODŠUMLJIVAČA ZA SLIKE GENERIRANE POSTUPKOM PRAĆENJA PUTA

Davor Najev

Zagreb, lipanj 2026.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. ožujka 2026.

DIPLOMSKI ZADATAK br. 1428

Pristupnik:	**Davor Najev (0036543497)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentorica:	prof. dr. sc. Željka Mihajlović

Zadatak: **Izrada i analiza odšumljivača za slike generirane postupkom praćenja puta**

Opis zadatka:

U modernoj računalnoj grafici metode praćenje zrake (engl. ray tracing) i praćenja puta (engl. path tracing) predstavljaju standard za generiranje fotorealističnih prikaza. No, no njihova je glavna mana visoka razina prisutnog šuma. Kako bi se postigla vizualna čistoća nužna je primjena naprednih tehnika odšumljivanja (engl. denoising) slike. Cilj rada je proučiti tehnike odšumljivanja slike temeljene na strojnom učenju te razvoj odgovarajuće programske arhitekture za odšumljivanje slike. Analizirati utjecaj raznih parametara na kvalitetu ostvarenih rezultata. Analizirati utjecaj skupa za učenje vezanog uz prikazane scene na ostvarene rezultate te dati kritički osvrt. Razmotriti različite metode evaluacije u ocjeni rezultata. Diskutirati upotrebljivost rješenja i dati smjernice za buduća poboljšanja. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 26. lipnja 2026.

Zahvaljujem mentorici na podršci i usmjeravanju pri izradi ovog rada.

Hvala obitelji na strpljenju i podršci tijekom cijelog mojeg školovanja.

Sažetak

Izrada i analiza odšumljivača za slike generirane postupkom praćenja puta

Davor Najev

U ovom radu opisana je izrada i analiza uklanjača šuma za slike generirane postupkom praćenja puta, temeljenog na dubokoj neuronskoj mreži arhitekture U-Net. Skup podataka generiran je programom Mitsuba 3 iz triju različitih scena, pri čemu su uz zašumljene slike spremani i pomoćni spremnici: difuzna refleksivnost, dubinska mapa i mapa normala. Model je treniran da modelira rezidual, tj. šum, umjesto cijele odšumljene slike. Sustavno su analizirani utjecaji ulaznih značajki, različitih funkcija gubitka (L1, Charbonnier, gubitak ovisan o frekvenciji i gubitak svjestan rubova), arhitekturalnih izmjena konvolucijskih blokova (ResNet i ConvNeXt) te predtreniranja na skupu ImageNet. Rezultati su vrednovani kvantitativno pomoću metrika PSNR i SSIM te kvalitativno vizualnom usporedbom s algoritmom BM3D. Pokazano je da pomoćni spremnici značajno poboljšavaju kvalitetu, da L1 i Charbonnierov gubitak u kombinaciji s komponentom svjesnom rubova daju najbolje rezultate te da pristupi temeljeni na dubokom učenju nadmašuju klasični BM3D algoritam u očuvanju detalja i strukture scene.

Ključne riječi: uklanjanje šuma; praćenje puta; duboko učenje; U-Net; konvolucijska neuronska mreža; računalna grafika

Abstract

Davor Najev

This thesis describes the design and analysis of a denoiser for images generated by path tracing, based on a deep neural network with the U-Net architecture. The dataset was generated using Mitsuba 3 from three different scenes, with noisy images accompanied by auxiliary buffers: albedo, depth map, and normal map. The model is trained to predict the residual, i.e., the noise, rather than the full denoised image. The work systematically analyses the influence of input features, different loss functions (L1, Charbonnier, frequency-aware loss, and edge-aware loss), architectural modifications of convolutional blocks (ResNet and ConvNeXt), and pretraining on the ImageNet dataset. The results are evaluated quantitatively using PSNR and SSIM metrics and qualitatively by visual comparison with the BM3D algorithm. It is shown that auxiliary buffers significantly improve the denoising quality, that L1 and Charbonnier losses combined with an edge-aware component yield the best results, and that deep learning based approaches outperform the classical BM3D algorithm in preserving details and scene structure.

Keywords: denoising; path tracing; deep learning; U-Net; convolutional neural network; computer graphics

Sadržaj

1. Uvod

Jedan od dugo razmatranih problema u računalnoj grafici je renderiranje uvjerljivih prikaza. U tu svrhu se već desetljećima koristi algoritam praćenja puta (*engl.* path tracing). Kako on koristi Monte Carlo uzorkovanje, on je u svojoj prirodi iterativan i stohastičan. Posljedica toga je da prikazi u prvoj iteraciji renderiranja imaju mnogo šuma koji se progresivno smanjuje pri povećanju broja iteracija. Zbog toga u praksi moramo raditi kompromis između vremenskog budžeta renderiranja i kvalitete konačnog prikaza. Moderne implementacije ovog algoritma također koriste i uklanjače šuma u svojoj implementaciji kako bi napravile taj kompromis prihvatljivijim.

Uklanjači šuma (*denoisers*) imaju zadaću ukloniti šum iz renderiranog prikaza u mnogo kraćem vremenu nego što bi potrošili daljnjim iteracijama renderiranja kako bi poboljšali kvalitetu prikaza. Najsuvremenije metode uklanjanja šuma temelje se na strojnom i dubokom učenju [1, 2]. Tako ostvareni uklanjači šuma mogu dati odlične rezultate uz vrlo kratko vrijeme izvođenja, pogotovo ako se integriraju u sustav za renderiranje na način da od njega dobiju dodatne podatke o sceni.

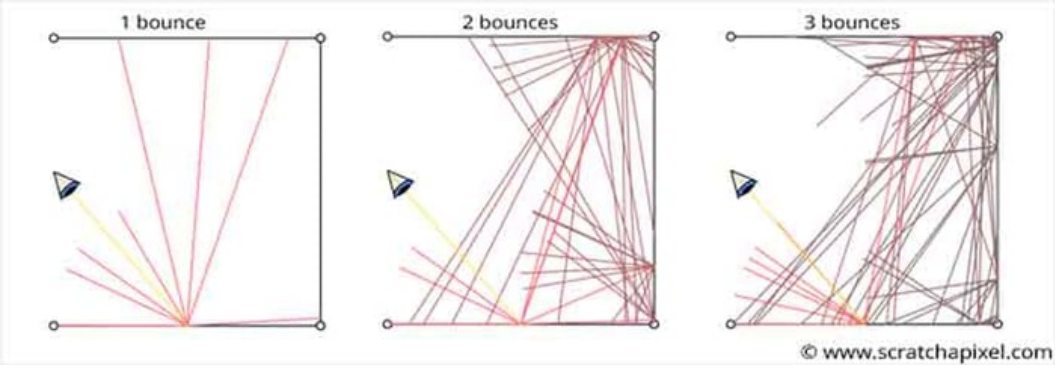
U ovom radu predložit ću svoju implementaciju uklanjača šuma temeljenu na arhitekturi dubokog učenja, analizirati njegovu učinkovitost u ovisnosti o različitim parametrima, arhitekturalnim izmjenama i tehnikama učenja, komentirati dobivene rezultate te iznijeti zaključke o najboljim praksama za izradu uklanjača šuma.

2. Problematika

2.1. Postavljanje problematike

Metoda praćenja zrake (*engl.* ray tracing) temeljna je tehnika u računalnoj grafici koja simulira ponašanje svjetlosti prateći zrake od kamere kroz virtualnu scenu. Za razliku od jednostavnijih metoda renderiranja, metoda praćenja zrake omogućuje realistično prikazivanje efekata poput refleksije, refrakcije i sjena, što je ključno za visokokvalitetne vizualizacije. Kako bi se postiglo globalno osvjetljenje, gdje svjetlost interagira s više površina, koristi se naprednija varijanta poznata kao praćenje puta (*engl.* path tracing), koja uključuje Monte Carlo uzorkovanje za aproksimaciju složenih svjetlosnih interakcija.

Srž renderiranja metodom praćenja puta jest funkcija BRDF (bidirectional reflectance distribution function). Ona govori kako površina reflektira dolazeću svjetlost u određenom smjeru, ovisno o smjeru dolaska i odlaska svjetlosti.



Slika 2.1. Vizualizacija zraka kod renderiranjem praćenjem puta dubine 1, 2 i 3

Algoritam praćenja zrake se konceptualno može zapisati sljedećom jednadžbom koja se naziva jednadžba renderiranja [3]:

$$L_o(p, \omega_o) = \int_\Omega f_r(p, \omega_i, \omega_o)\, L_i(p, \omega_i)\, (\mathbf{n} \cdot \omega_i)\, d\omega_i$$

gdje su

- $L_o(p, \omega_o)$ - izlazno osvjetljenje u točki p u smjeru ω_o
- $L_i(p, \omega_i)$ - ulazno osvjetljenje u točki p iz smjera ω_i
- $f_r(p, \omega_i, \omega_o)$ - funkcija BRDF koja opisuje kako površina reflektira svjetlost
- $(\mathbf{n} \cdot \omega_i)$ - kosinus kuta između normale površine $\mathbf{n}$ i ulaznog smjera ω_i

Opisano riječima: osvjetljenje u nekoj točki ovisi o ulaznom osvjetljenju iz svih smjerova prema toj točki, a način na koji se svjetlost reflektira od površine definira funkcija BRDF (*engl.* bidirectional reflectance distribution function) koja opisuje materijalnu strukturu površine kojoj ta točka pripada. U praksi ovaj integral implementiramo tako da za svaku točku prikaza bacamo niz zraka koje se odbijaju po sceni, pri čemu se za svaku interakciju uzorkuje novi smjer prema funkciji BRDF kako bi se aproksimirao doprinos svjetlosti. Međutim, kako bismo savršeno zadovoljili jednadžbu renderiranja, potrebno je uračunati doprinos svih mogućih zraka. Zbog ograničenih računalnih resursa, za svaki piksel bacamo samo ograničen broj zraka koje se odbijaju samo do ograničene dubine. Konceptualno, to odgovara aproksimaciji integrala u jednadžbi renderiranja Monte Carlo uzorkovanjem:

$$L_o(p, \omega_o) \approx \frac{1}{N} \sum_{k=1}^{N} \frac{f_r(p, \omega_i^k, \omega_o) L_i(p, \omega_i^k)(\mathbf{n} \cdot \omega_i^k)}{p(\omega_i^k)}$$

Ovakva aproksimacija konvergira prema izvornom integralu, ali jako sporo. U praksi to znači da ako uzmemo malo zraka po pikselu (*engl.* SPP — Samples Per Pixel), konačni prikaz će sadržavati šum.

Kako bismo dobili rezultat visoke kvalitete, potreban je jako velik broj zraka po pikselu, obično 1000 i više. To je u praksi nepoželjno zbog dugog vremena izvođenja. Zbog toga su osmišljeni **uklanjači šuma** (*engl.* denoisers) kojima je moguće dobiti visokokvalitetan rezultat uz jako malen broj zraka po pikselu.

U okviru ovog rada bit će ostvaren uklanjač šuma temeljen na dubokom učenju. Ispitat ću različite varijacije u implementaciji:

- kvantitativno — analiziranjem različitih metrika
- kvalitativno — izravnom vizualnom usporedbom rezultata

U nastavku će biti definirani detalji implementacije.

2.2. Implementacija

Trenutačno najkvalitetniji uklanjači šuma temelje se na **dubokim neuronskim mrežama**. Uklanjanje šuma u bilo kakvom signalu poznat je problem, ali težak za uspješno riješiti, pogotovo kod netrivijalnih signala kao što su slike. Duboke neuronske mreže pokazale su se prilično uspješnima za rješavanje tog problema pa će taj pristup biti korišten i u ovom radu.

Duboko učenje jedna je od metoda u strojnom učenju. Kao i kod svakog problema koji rješavamo strojnim učenjem, trebamo definirati sljedeće stvari:

- **skup podataka** — potrebni su podatci iz kojih će uklanjač šuma učiti kako ukloniti šum
- **arhitektura modela** — model uklanjača šuma bit će duboka neuronska mreža, specifično konstruirana za rekonstrukciju slike i uklanjanje šuma
- **način učenja** — potrebno je definirati način evaluacije i algoritam kojim će se uklanjač šuma iterativno poboljšavati.

Sve ove stavke bit će definirane u sljedećim potpoglavljima.

2.2.1. Skup podataka

Za kvalitetno učenje bit će potreban raznovrstan skup podataka koji se sastoji od višemodalnog ulaza u model te referentnog izlaza niskog šuma (*engl.* ground truth) s kojim ćemo uspoređivati izlaz iz modela.

Za stvaranje skupa podataka korišten je program Mitsuba 3 [4]. To je program koji omogućuje skriptabilno renderiranje scena zapisanih pomoću YAML datoteka. Pomoću

programskog jezika Python možemo precizno kontrolirati kvalitetu renderiranja postavljanjem broja zraka po pikselu (SPP), kameru i izlazne spremnike.

Za potrebe ovog rada napravljena su 3 različita skupa podataka. Svaki se sastoji od 1000 slika generiranih iz 3 različite scene javno dostupne na Mitsuba 3 web stranici s primjerima scena.

Prva scena naziva "**Cornell Box**" klasik je u računalnoj grafici. To je jednostavna scena koja se sastoji od jedne sobice različitih boja zidova, jednog izvora svjetla na stropu te nekoliko jednostavnih oblika (kocke, kugle). Zbog svoje jednostavnosti, većina prikaza renderiranih iz nje bit će niskofrekventna. Na takvim prikazima se i mali šum može golim okom prepoznati pa trebamo kvalitetan uklanjač šuma kako ne bi bio primijećen.

Kako nam je cilj napraviti uklanjač šuma koji radi na što širem spektru različitih slika, potrebne su nam i nešto kompleksnije scene. Druga scena naziva "**Bedroom**" prikazuje spavaću sobu koja je kompleksnija i detaljnija u usporedbi sa sobom u sceni "Cornell Box". Kod ovakvih scena šum je manje vidljiv ako nije snažan, ali se u principu teže uklanja.

Zadnja scena koja će biti korištena jest "**Bathroom**". Ona je po strukturi slična sceni "Bedroom". Njezina svrha bit će evaluacija uklanjača šuma treniranog na drugim scenama kako bismo vidjeli koliko se dobro ponaša na sceni koju do sada nije vidio, što je pouzdanija procjena kvalitete nego izrada skupa za testiranje (slika na kojima nije treniran) u sceni na kojoj je treniran.

Reprezentativne slike scena mogu se vidjeti na slici 2.2.

(a) Cornell Box **(b)** Bedroom **(c)** Bathroom

Slika 2.2. Scene nad kojima su generirani skupovi podataka

Raznovrsnost skupova podataka osigurana je nasumičnim postavljanjem kamere i njezine orijentacije u sceni.

Kako bismo postigli što bolje rezultate, poželjno je na ulaz neuronske mreže dovesti što više relevantnih podataka koji bi mogli poslužiti za rekonstrukciju. U tu svrhu bit će korišteni sljedeći podatci:

- RGB kanali prikaza koji želimo odšumiti
- RGB kanali difuzne refleksivnosti (*engl.* albedo)
- kanal dubinske mape
- XYZ koordinate mape normala u obliku 3 zasebna kanala.

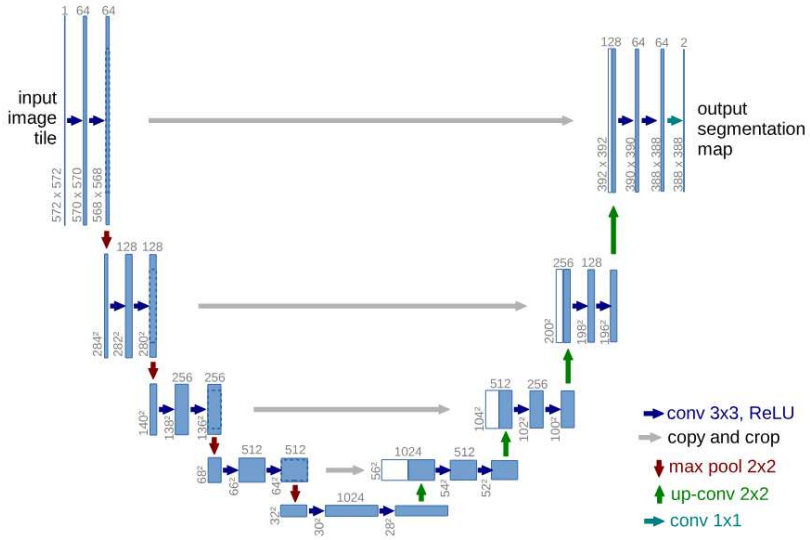
Svrha prva 3 kanala su očita: bez slike kojoj želimo ukloniti šum neuronska mreža neće moći naučiti ništa korisno. Sljedećih 7 kanala odabrano je jer sadrže mnogo informacija o sceni, a mogu se jeftino ili besplatno dobiti iz pogona za renderiranje metodom praćenja zrake čiji izlaz želimo odšumiti. Difuzna refleksivnost se može dobiti vrlo jeftino od pogona za renderiranje jer ne zahtijeva renderiranje sjena, a neki ga pogoni generiraju usputno pri renderiranju praćenjem puta. Dubinska mapa se dobija jednostavnim računanjem udaljenosti prvog sudara svake zrake od kamere što se već ionako računa pri renderiranju, a usputno se zabilježe i koordinate normala u toj točki u mapu normala. Ovo čini sve skupa 10 kanala u ulazu u model.

Referentni izlazi sastoje se samo od RGB kanala visokokvalitetnog prikaza koji točno odgovaraju prikazima koji se nalaze u ulaznim podatcima, samo što su renderirani s jako visokim brojem zraka po pikselu.

U skupu podataka za svaku sliku nalazi se 1000 različitih parova ulaznih podataka i referentnih vrijednosti. Pri treniranju uklanjača šuma poželjno je koristiti što veći broj slika. U implementaciji za potrebe ovog rada broj primjera po sceni je ograničen na 1000 jer je to dovoljan broj primjera da se dobiju kvalitetni rezultati, a brzina treniranja je još uvijek prihvatljiva. Kako cilj ovog rada nije postići najbolji uklanjač pod svaku cijenu, već istražiti što sve utječe na kvalitetu istoga, kvalitetu uklanjača neću promatrati na većim skupovima za treniranje.

2.2.2. Model

Za implementaciju uklanjača šuma bit će korištena neuronska mreža **U-Net** [5]. Arhitektura ove mreže prikazana je na slici 2.3.



Slika 2.3. Arhitektura neuronske mreže U-Net

Glavne značajke ove neuronske mreže:

- autoenkoderska arhitektura
- konvolucijski slojevi te
- rezidualne veze.

Autoenkoderske neuronske mreže sastoje se od 2 glavna dijela: enkodera i dekodera. Ideja je transformirati ulaz u mrežu u njegovu međureprezentaciju u obliku vektora dimenzije 1024 koji sadrži korisne informacije o ulaznoj slici. Iz toga vektora dekoder sastavlja izlaz. Pri tome značajke ulaza koje je vektor uhvatio moraju biti dovoljno bogate i kvalitetne kako bi rekonstruirani izlaz bio kvalitetan. Ovakva simetrična konstrukcija-rekonstrukcija temelj je U-Neta, a po njoj je i dobio ime (mreža je karakterističnog "U" oblika).

Konvolucijski slojevi rade operaciju konvolucije nad njihovim ulazom. Kod enkodera oni služe kako bi izvukli značajke iz slike i to se događa kroz 4 razine. Svaka razina sastoji se od 2 konvolucijska sloja i sloja maksimalnog sabiranja (*engl.* max pool) na čijem

se izlazu nalaze značajke koje su upola manje rezolucije nego ulaz. Ovom kompozicijom razina mogu se izvući kvalitetne značajke te se priprema izvlačenje značajki na nižoj rezoluciji slike. Postupak se ponavlja na svakoj razini koja radi na ulazima upola manje rezolucije od prethodne razine, što omogućuje izvlačenje sve bogatijih značajki. Kod dekodera konvolucijskim blokovima sastavljamo izlaz iz vektora sa značajkama, postupno povećavajući rezoluciju dok ne dobijemo konačni izlaz.

Rezidualne veze pretvaraju model u rezidualni i poboljšavaju kvalitetu rekonstrukcije izlaza. Tijekom kodiranja u vektor značajki, informacije iz prethodnih razina postaju manje vidljive, što otežava rekonstrukciju. Rezidualne veze omogućuju da svaki dekoderski blok uči razliku između ulaza i željenog izlaza (rezidual) umjesto pune reprezentacije. Time dekoder zadržava većinu strukture iz ulaza i samo nadodaje naučenu razliku, što olakšava učenje i poboljšava rezultate.

Ovako konstruirana neuronska mreža može dobro naučiti šum u ulaznim podatcima. U sljedećem poglavlju bit će opisan postupak učenja ove neuronske mreže u svrhu uklanjanja šuma.

2.2.3. Način učenja

Prvo je potrebno odrediti što će neuronska mreža točno učiti.

Prva opcija jest da neuronska mreža uči cijeli odšumljeni prikaz. Ako odšumljeni prikaz označimo s $H(x)$ gdje je x zašumljeni prikaz s pomoćnim spremnicima s dodatnim podatcima, izlaz neuronske mreže je sam taj $H(x)$. Međutim, ovo neće dati dobre rezultate. Razlog tome je što neuronska mreža mora naučiti savršeno rekonstruirati izlaz iz izvučenih značajki. Kako bi to bilo moguće, bit će nam potreban jako velik broj raznovrsnih prikaza uz pomoć kojih će neuronska mreža naučiti dobro rekonstruirati različite dijelove scene. Nažalost, čak i ako je na taj način treniramo, opet će posustati kad pokušamo odšumiti prikaze renderirane iz neke druge scene.

Alternativni pristup jest treniranje neuronske mreže na način da ona **umjesto cijele scene modelira samo šum** [6]. U tom slučaju, neuronska mreža uči rezidual $F(x)$:

$$F(x) = H(x) - x$$

A konačan izlaz odšumljivača će biti zbroj ulaza i reziduala (modeliranog šuma):

$$H(x) = x + F(x)$$

Na ovaj način smo izbjegli potrebu da neuronska mreža uči rekonstrukciju cijele scene iz izlaza. Modeliranje reziduala je mnogo lakši zadatak, dijelom i zbog njegove strukture. Slika koju dovodimo na ulaz neuronske mreže je u decimalnom obliku, tj. u vrijednostima [0, 1]. Rezidual koji na nju nadodajemo kako bismo uklonili utjecaj šuma je u rasponu vrijednosti [-1, 1] s očekivanom vrijednošću 0. To su vrlo poželjna svojstva za učenje neuronskim mrežama i rezultat toga jest da će učenje biti lakše.

Učenje će biti napravljeno u programskom jeziku Python pomoću programskog okvira PyTorch. Bit će korišteni sljedeći hiperparametri:

- stopa učenja (*engl.* learning rate) — $1e^{-4}$
- regularizacija težina (*engl.* weight decay) — $1e^{-5}$
- optimiziranje gubitka — gradijentni spust algoritmom Adam [7]

Vrijednosti ostalih bitnih hiperparametara će biti spomenute po potrebi.

Funkcije gubitka bit će različite:

- L1 gubitak
- Charbonnierov gubitak
- gubitak ovisan o frekvenciji
- gubitak svjestan rubova

2.3. Evaluacija

Rezultati će biti analizirani kvalitativno i kvantitativno.

Za kvalitativnu analizu kvaliteta uklanjača šuma bit će vizualno uspoređena s BM3D algoritmom [8].

Za kvantitativnu evaluaciju koristit će se **funkcije gubitka**, **PSNR** i **SSIM**.

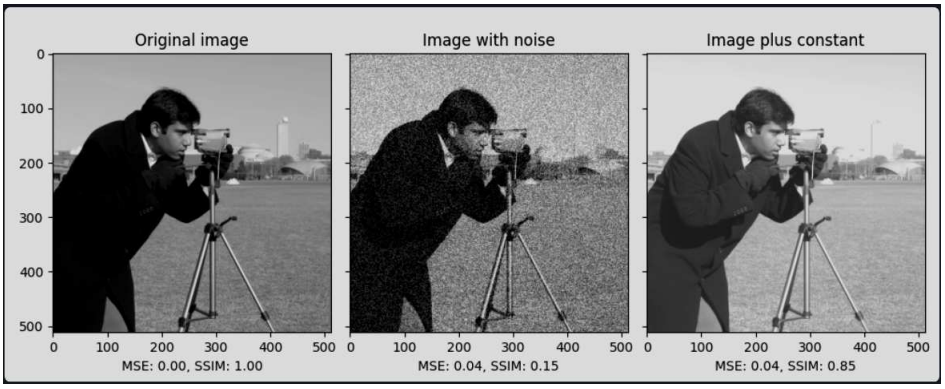
Funkcije gubitka temeljne su i nužne evaluacije u strojnom učenju. Kako je i uobičajeno, uklanjač šuma optimiziramo prema gubitku na skupu za treniranje, određujemo optimalan broj iteracija treniranja pomoću gubitka na validacijskom skupu te određujemo konačne performance modela na temelju gubitka na skupu za testiranje. Moramo imati na umu da ćemo pri ispitivanju koristiti više različitih funkcija gubitaka. Možemo izravno uspoređivati gubitke modela koje smo optimizirali prema istoj funkciji gubitka, ali ne i prema različitoj.

PSNR (Peak Signal to Noise Ratio) standardna je metrika za mjere količine šuma. Računa se kao omjer maksimalne moguće snage signala i snage šuma. Konkretno, kod slike se računa:

$$PSNR = 10 \log_{10} \frac{R^2}{MSE}$$

gdje je R maksimalna moguća promjena signala (255 za 8-bitne slike), a MSE srednja kvadratna pogreška između odšumljene i referentne slike. Valja skrenuti pozornost da je ova metrika izravno izvedena iz srednje kvadratne pogreške koja nema nikakve informacije o prostornoj sličnosti dijelova u slici. Posljedica toga jest da je moguće dobiti visok PSNR za slike koje još uvijek izgledaju zašumljeno. Zbog toga ćemo uvesti još jednu metriku koja također uzima u obzir i prostornu sličnost dijelova slike.

SSIM (Structural Similarity Index Measure) [9] metoda je usporedbe dviju slika koja favorizira prostornu sličnost. Ovu metriku možemo smatrati naprednijom verzijom PSNR-a zbog tog svojstva koje je jako bitno kod vizualne percepcije slike. Može poprimiti vrijednosti $[0, 1]$. Vrijednost 1 dobivamo za potpuno jednake slike, a vrijednost blizu 0 za slike koje su jako različite, npr. za dvije različite slike potpuno nekoreliranog Gaussovog šuma. Ova metrika pogodna je za evaluaciju sličnosti odšumljene slike referentnoj jer penalizira slučajeve gdje strukturalna sličnost nije očuvana. Strukturalnu sličnost želimo postići u odšumljenoj slici. Ako je ne postignemo, odšumljena slika izgledat će zašumljeno čak i kada vrijednosti funkcije gubitka (kao što je npr. L1 gubitak) budu male. Zbog toga je ova metrika nužna za potpunu evaluaciju modela. Na slici 2.4. možemo vidjeti ponašanje ove metrike na različitim primjerima.



Slika 2.4. Demonstracija SSIM metrike

3. Analiza

U ovom poglavlju bit će analizirani utjecaji različitih parametara na performance uklanjača šuma.

Varijacije koje će se razmatrati:

- ablacija ulaznih slojeva modela
- različite funkcije gubitaka
- varijacije u arhitekturi modela
- predreniranje
- raznovrsnost skupa za treniranje

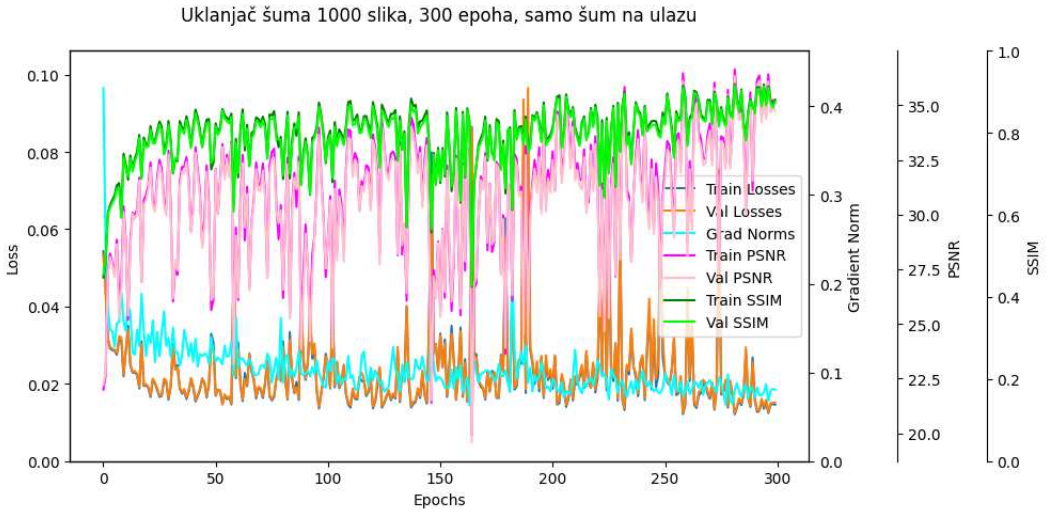
Svaka od varijacija bit će posebno razmatrana i obrađena. Na kraju će biti odabran model s najboljom kombinacijom hiperparametara te će bit podvrgnut daljnjoj analizi.

3.1. Ablacijsko testiranje

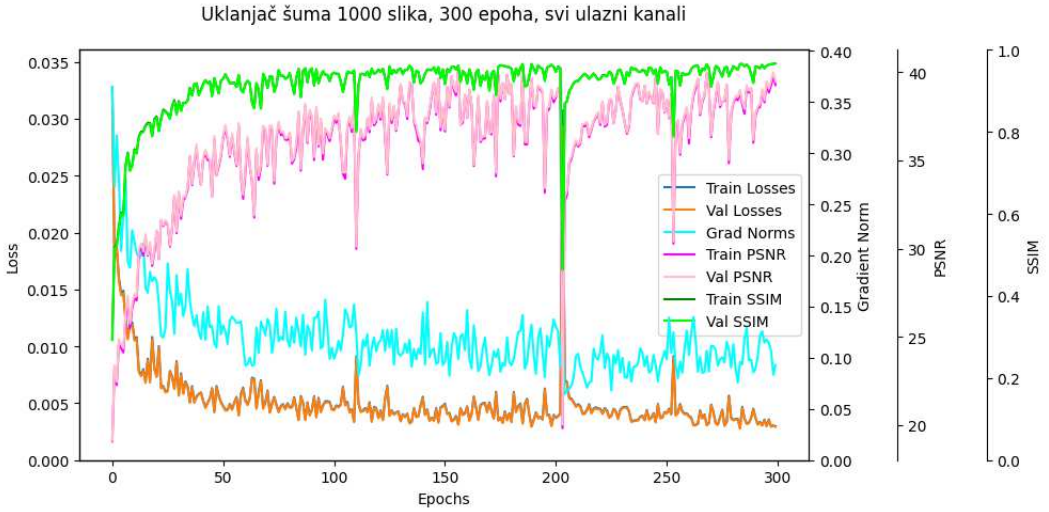
Ablacijskim testiranjem ispitujemo utjecaj pojedinih ulaza u model kako bismo donijeli potencijalno korisne zaključke o njima.

Ulaz u uklanjač šuma jest zašumljena slika i dodatni podatci koji donose dodatne informacije o renderiranoj slici. Pogledat ćemo koliko dodatni podatci utječu na snagu uklanjača uspoređujući uklanjač koji je treniran na svim ulazima i onaj koji je treniran samo na zašumljenoj slici.

Statistike tijekom treniranja u oba slučaja prikazane su na slikama 3.1. i 3.2.



Slika 3.1. Statistike po epohama modela treniranog samo na zašumljenim slikama



Slika 3.2. Statistike po epohama modela treniranog na svim podatcima

Jasno je vidljivo da je treniranje kada su uključeni svi kanali stabilnije i konvergira prema višim PSNR i SSIM vrijednostima. Vizualnom usporedbom na slici 3.3. možemo vidjeti kako se to manifestira na izlaznoj slici. Kao što bismo i očekivali, budući da uklanjač ima manju količinu informacija o sceni, odšumljena slika je vrlo zamućena, što uzrokuje gubitak detalja. Uklanjač koji ima sve kanale ulaza mogao je mnogo bolje odšumiti zbog dodatnih informacija koje se nalaze u ostalim kanalima.

Zaključak: dodatni kanali uvelike pomažu uklanjanju pri rekonstrukciji šuma te u konačnici pri rekonstrukciji odšumljene slike.

Slika 3.3. Usporedba prikaza odšumljenih modelima sa svim i bez svih ulaznih kanala

3.2. Utjecaj različitih funkcija gubitaka

Kada model dobije neki izlaz za neki ulaz, želimo da on bude jednak referentnoj vrijednosti (*engl.* ground truth). Funkcija gubitka služi nam za kvantificiranje njihove različitosti, što omogućuje davanje ocjene kvalitete modela zbrajanjem gubitka između svih podataka kojima treniramo model i njihovih referentnih vrijednosti. U konkretnom slučaju uklanjača šuma, funkcijom gubitka mjerimo srednju udaljenost svakog piksela odšumljene slike od referentne slike renderirane bez šuma.

Vrlo je važno svojstvo funkcije gubitka da je diferencijabilna, što omogućuje da algoritam optimizacije uči model mijenjajući težine modela u negativnom smjeru gradijenta te funkcije gubitka u odnosu na težine.

U osnovnim algoritmima strojnom učenju funkcije gubitka se obično ne mijenjaju jer su one izravno izvedene na temelju nekih poželjnih svojstava. Kod dubokog učenja (pogotovo u praktičnoj domenskoj primjeni), funkciju gubitka prilagođavamo ovisno o problemu koji rješavamo.

Kod rekonstrukcije slike, prva funkcija gubitka koju valja spomenuti jest L2 gubitak, tj. euklidska udaljenost, koja je definirana na sljedeći način:

$$L_{L2}(x, y) = \frac{1}{N} \sum_i (x_i - y_i)^2$$

gdje je x odšumljena slika i x_i svaki njezin pojedinačni piksel, a y referentna slika i y_i svaki njezin pojedinačni piksel. Ova funkcija gubitka općenito je prvi odabir za optimizaciju rekonstrukcije slika, međutim ima nepoželjna svojstva. Njezin najveći problem jest što uzrokuje da algoritam optimizacije konvergira prema slikama koje su zamućene.

Razlog tome je što ta funkcija gubitka ne uračunava nikakvu prostornu sličnost u slici pri računanju gubitka, što dovodi do toga da rubovi postanu mutni kako bi se maksimalno smanjio gubitak. Zbog tog svojstva L2 gubitak neće biti razmatran pri testiranju modela jer je već poznato da postoje funkcije gubitka koje su bolje od njega pa ćemo njih koristiti za usporedbu umjesto L2 gubitka.

Funkcije gubitka koje će biti obrađene:

- L1 gubitak
- Charbonnierov gubitak
- gubitak ovisan o frekvenciji
- gubitak svjestan rubova

L1 gubitak jest gubitak apsolutne udaljenosti izlaza mreže i cilja. Definiran je na sljedeći način:

$$L_{L1}(x, y) = \frac{1}{N} \sum_i |x_i - y_i|$$

odnosno kao srednja apsolutna razlika između piksela odšumljene i referentne slike. Ova funkcija gubitka obično je prvi odabir kod izrade uklanjača šuma jer bolje čuva detalje i rubove u slici u usporedbi s L2 gubitkom.

Međutim, L1 gubitak ima svoje mane. Iako je bolji od L2 gubitka u očuvanju detalja, još uvijek nije savršen. Također ne sadrži informacije o prostornoj sličnosti. Posljedica toga jest da je moguće dobiti nisku vrijednost gubitka u slučajevima kada je šum još uvijek vidljiv. Što je šum vidljiviji, to su prostorno bliski pikseli međusobno različitiji, odnosno prostorna sličnost pada, ali L1 gubitak nema načina to detektirati.

Prva funkcija gubitka kao nadogradnja jest **Charbonnierov gubitak**. On je definiran na sljedeći način:

$$L_{char}(x, y) = \frac{1}{N} \sum_i (x_i - y_i)^2 + \epsilon^2$$

gdje je ϵ jačina regularizacije. Ovaj gubitak možemo opisati kao L2 gubitak s nekom vrstom regularizacije. Ovako definiran gubitak daje bolje rezultate od L2 i L1 gubitka. Jedan od razloga jest bolji protok gradijenata. Optimizacijski algoritmi kao što je stohas-

tički gradijentni spust vole funkcije koje su derivabilne u cijeloj svojoj domeni. Funkcija L1 gubitka ima točku gdje nije derivabilna, u svojem ishodištu, što je nepoželjno. To i glatkost te funkcije gubitka (koju ima zbog strukture slične L2 gubitku) rezultira većom oštrinom rekonstruiranih slika. Međutim, ova funkcija gubitka također ne adresira problem prostorne sličnosti koji smo opisali ranije, što će uzrokovati slične probleme i ovdje.

Sljedeća funkcija gubitka koju ćemo razmatrati jest **gubitak ovisan o frekvenciji** (*engl.* frequency-aware loss). On je definiran na sljedeći način:

$$L_{freq} = |\mathcal{F}(x) - \mathcal{F}(y)|$$

gdje je $\mathcal{F}(x)$ slika transformirana Fourierovom transformacijom, tj. slika u svojoj frekvencijskoj domeni. Možemo reći da ova funkcija gubitka mjeri razliku između slika u njihovoj frekvencijskoj domeni. Nada je da će ovaj gubitak bolje očuvati visoke frekvencije u slici, tj. detalje.

Zadnja funkcija gubitka koja će biti razmatrana jest **gubitak svjestan rubova** (*engl.* edge-aware loss). On je definiran na sljedeći način:

$$L_{edge} = |\Delta_x x - \Delta_x y| + |\Delta_y x - \Delta_y y|$$

gdje su

- $\Delta_x x$ — gradijent referentne slike po x osi
- $\Delta_x y$ — gradijent odšumljene slike po y osi
- $\Delta_y x$ — gradijent referentne slike po x osi
- $\Delta_y y$ — gradijent odšumljene slike po y osi

Ideja je uzeti već postojeći gubitak i na njega dodati još jedan član koji će jače penalizirati ako rubovi nisu cjeloviti, što bi trebalo poboljšati oštrinu slike. U svrhu ovog uklanjača šuma koristit će se L1 gubitak kao bazna funkcija gubitka. To znači da će ukupni gubitak

biti:

$$L_{total} = L_1 + L_{edge}$$

Definirane su sve funkcije gubitka. Sada ćemo prijeći na kvantitativnu i kvalitativnu analizu.

3.2.1. Analiza na skupu podataka jednostavnije scene

Prvo ćemo napraviti kvantitativnu analizu pomoću već određenih metrika. Vrijednosti metrika za modele trenirane na svim spomenutim gubitcima nalaze se u tablici 3.1. Za svaku funkciju gubitka prikazane su metrike najboljeg modela dobivenog nakon nekoliko treniranja.

Gubitak	**Skup za treniranje**		**Skup za testiranje**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
L1	39.32	0.964	38.98	0.964
Charbonnier	39.33	0.977	39.35	0.977
Frequency-aware	16.87	0.592	16.74	0.583
Frequency-aware (L1)	10.72	0.219	10.83	0.229
Edge-aware (L1)	36.78	0.948	36.48	0.945
Edge-aware (Charbonnier)	39.46	0.976	39.20	0.976

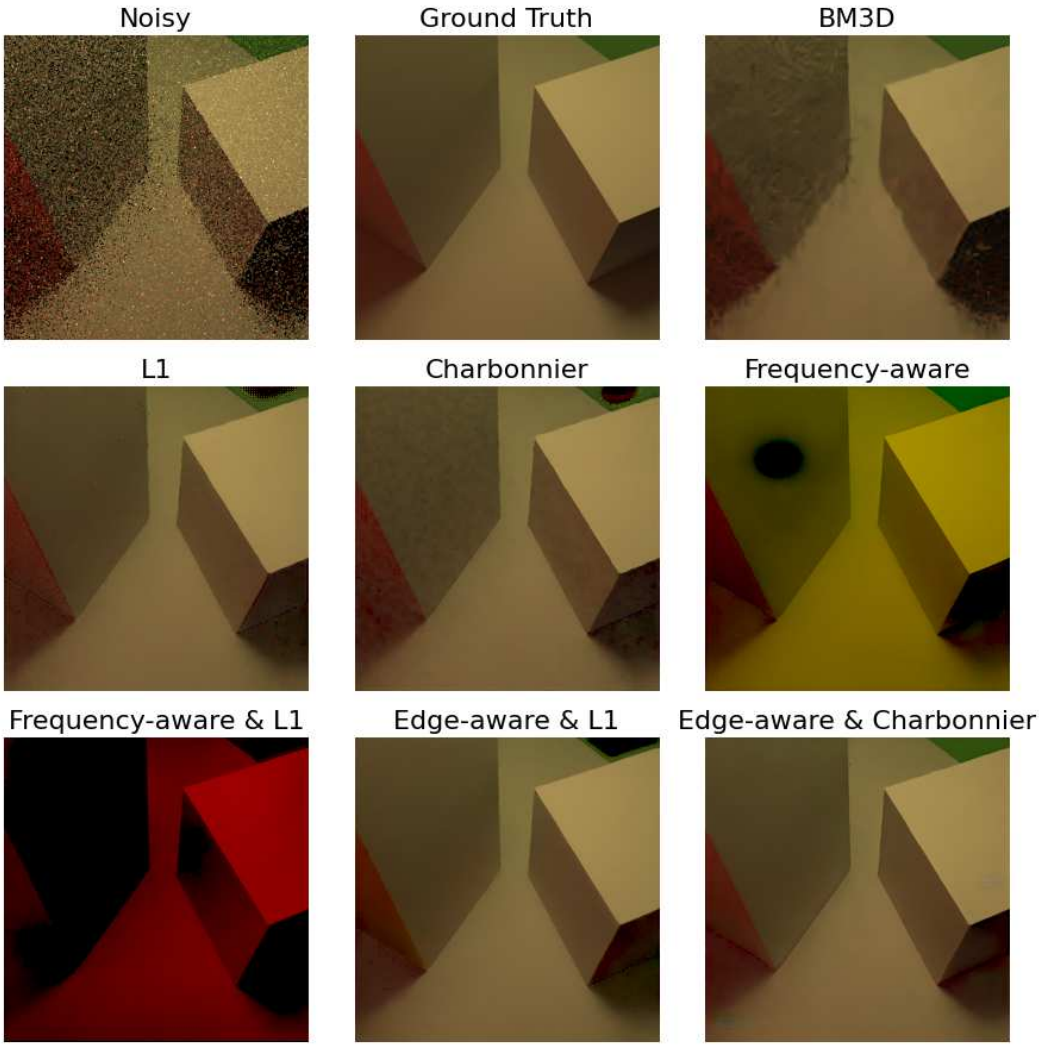
Tablica 3.1. Usporedba modela treniranih pomoću različitih funkcija gubitaka

Po metrikama, L1, Charbonnier i gubitak ovisan o rubovima dali su dobre rezultate. Svi su bili bliski i po SSIM-u i po PSNR-u; najbolji je Charbonnier i po SSIM-u i po PSNR-u na skupu za testiranje, ali za jako malu vrijednost.

Gubitak ovisan o rubovima ispitan je u kombinaciji s L1 te Charbonnierovim gubitkom. U kombinaciji s L1 gubitkom metrike su nešto niže nego u slučaju bez njega, a u kombinaciji s Charbonnierovim su ostale jednakima. Prema tim rezultatima ne možemo tvrditi da komponenta gubitka ovisna o rubovima ima utjecaja na krajnji rezultat.

Gubitak ovisan o frekvenciji dao je najlošije rezultate. Potencijalni razlog tome jest što je ovaj gubitak dosta nestabilan kada se koristi sam, zbog čega ga se preporučuje kombinirati s nekim drugim gubitkom. Zbog toga je uveden novi gubitak koji je kombinacija L1 i gubitka ovisnog o frekvenciji; u tablici je označen u redu "Frequency-aware (L1)". Nažalost, ovo nije dalo bolje rezultate. Upravo suprotno: rezultati su se pogoršali.

Na slici 3.4. se vide demonstracija odšumljivanja modelima treniranim različitim funkcijama gubitka na reprezentativnom primjeru.



Slika 3.4. Vizualizacija prikaza odšumljenih modelima treniranim različitim funkcijama gubitka na sceni "Cornell Box".

Ovdje primjećujemo razlike u kvaliteti izlaza koje nismo uspjeli primijetiti u kvantitativnoj analizi.

Iako su metrike dobivene Charbonnierovim gubitkom najbolje, rezultat je lošiji od

L1 gubitka te gubitaka s komponentom osjetljivom na rub zato jer se još uvijek može vidjeti ostatak neuklonjenog šuma. Metrike nisu uspjele uhvatiti manjak strukturne cjelovitosti, što se jasno vidi na slici. S druge strane, modeli trenirani s gubitcima s komponentom ovisnom o rubovima nemaju taj problem. Štoviše, mogli bismo reći da daju bolji rezultat nego model treniran na čistom L1 gubitku.

Modeli trenirani s gubitcima ovisnima o frekvenciji nisu dali dobre rezultate, što smo očekivali gledajući metrike. Zanimljivo je to što rezultati u principu izgledaju jako dobro odšumljeno, samo što im je nijansa pogrešna. To sugerira da je generalna ideja funkcije gubitka dobra te da bi se pametno određenim izmjenama mogla pretvoriti u funkcionalnu i korisnu funkciju gubitka.

3.2.2. Analiza na skupu podataka kompleksnije scene

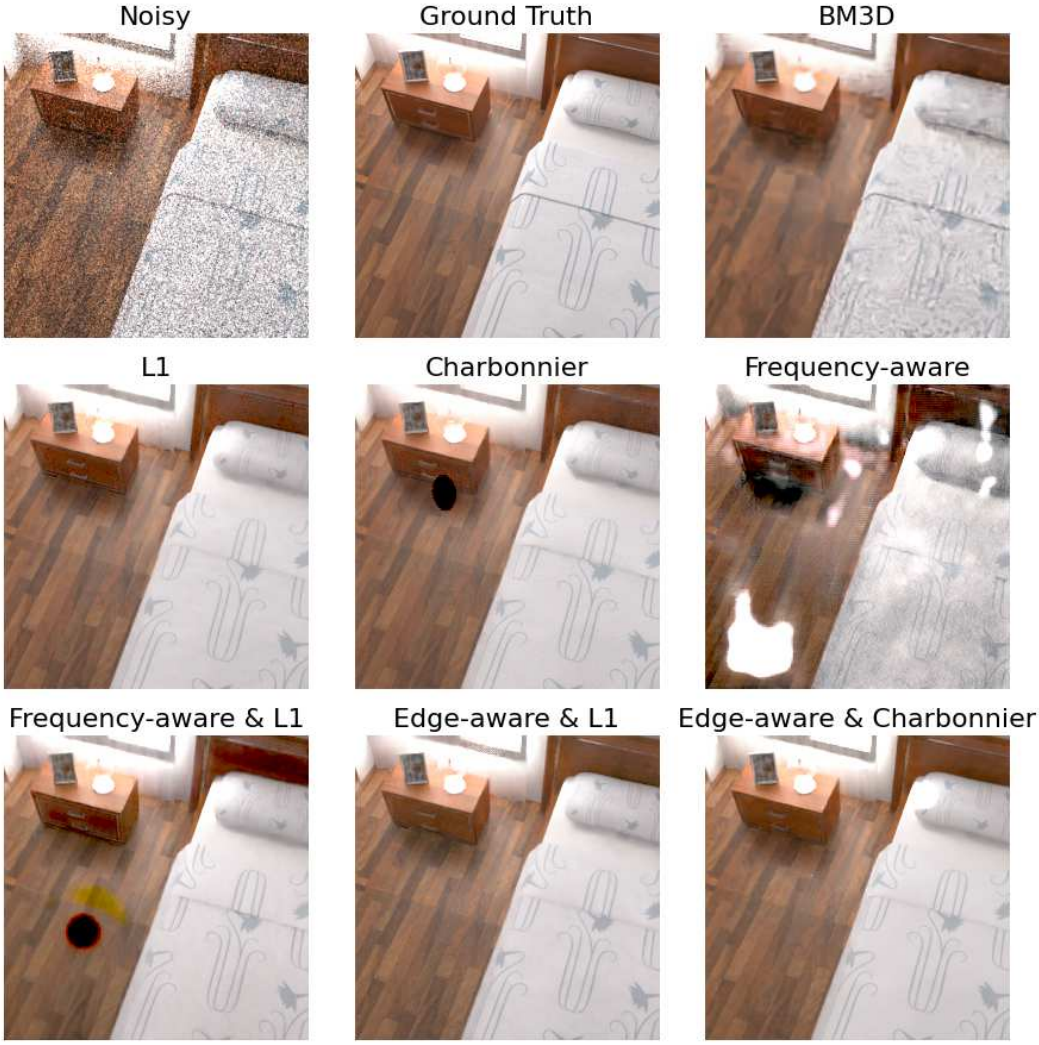
Sada ćemo napraviti istu analizu na kompleksnijoj sceni. U tu svrhu bit će korištena testna scena "Bedroom". Metrike dobivene kvantitativnom analizom su prikazane u tablici 3.2.

Gubitak	**Skup za treniranje**		**Skup za testiranje**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
L1	33.14	0.877	32.98	0.873
Charbonnier	34.89	0.887	34.92	0.887
Frequency-aware	22.35	0.725	22.4	0.717
Frequency-aware (L1)	28.84	0.858	29.22	0.851
Edge-aware (L1)	36.26	0.920	36.08	0.922
Edge-aware (Charbonnier)	35.59	0.914	35.31	0.911

Tablica 3.2. Usporedba modela treniranih pomoću različitih funkcija gubitaka

Opet primjećujemo superiorne performance gubitaka s komponentom ovisnom o rubovima. Gubitci L1 i Charbonnier bez komponente s rubovima daju nešto niže metrike u usporedbi s prijašnjom scenom, ali još uvijek visoke. Gubitci ovisni o frekvenciji daju u ovom slučaju mnogo bolje rezultate, pogotovo varijanta kombinirana s L1 gubitkom.

Vizualna usporedba na reprezentativnom primjeru se nalazi na slici 3.5.



Slika 3.5. Vizualizacija prikaza odšumljenih modelima treniranim različitim funkcijama gubitka na sceni "Bedroom".

I ovdje možemo potvrditi superiornost gubitaka s komponentom ovisnom o rubovima, dok ih prate čisti L1 i Charbonnier gubitci. Gubitci temeljeni na frekvenciji ponovno daju najlošije rezultate, iako je verzija kombinirana s L1 gubitkom mnogo bolja nego u prijašnjem slučaju, ali ne dovoljno da bude bolja od prethodno spomenutih slučajeva. BM3D opet daje zamućene i donekle šumovite rezultate. Šum je nešto manje primjetan jer je slika kompleksnija i ima više detalja/visokofrekventnih elemenata, ali nije dovoljno neprimjetan da bismo mogli reći da je slika uspješno odšumljena.

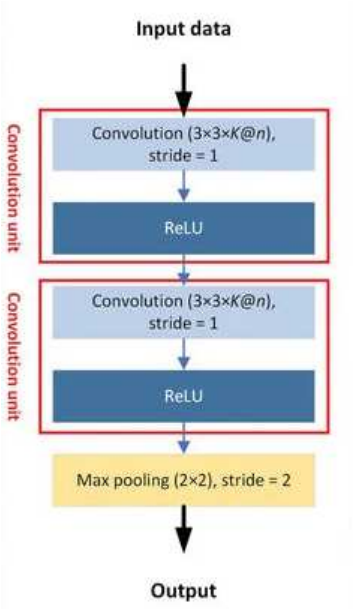
Zaključak: L1 i Charbonnierov gubitak daju odlične rezultate kada se kombiniraju s komponentom gubitka ovisnog o rubovima. Sam L1 gubitak radi dovoljno dobar posao, a

Charbonnierov može ostaviti vidljiv šum. Gubitak temeljen na frekvenciji ne daje dobre rezultate, barem u trenutačnom obliku.

3.3. Utjecaj arhitekturalnih promjena modela

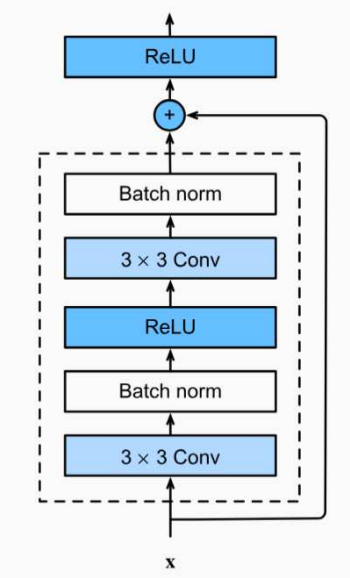
Za model je odabrana arhitektura U-Net koja je dobar izbor za uklanjač šuma. Međutim, tijekom godina pronađene su arhitekturne izmjene pomoću kojih je moguće postići bolje rezultate. Neke od njih su ResNet i ConvNeXt blokovi koji zamjenjuju standardne konvolucijske blokove. Bit će napravljena zamjena konvolucijskih blokova tim dvama blokovima te će biti ispitan utjecaj na performance uklanjača šuma.

Standardni konvolucijski blok prikazan je na slici 3.6.



Slika 3.6. Konvolucijski blok u standardnom U-Netu

ResNet[10] blokovi jedna su od prvih nadogradnji na standardne konvolucijske blokove, a ističu se svojom univerzalnošću i generalno dobrim performansama pri izvlačenju značajki iz slike. Prikaz ResNet bloka nalazi se na slici 3.7.



Slika 3.7. ResNet blok

ConvNeXt[11] blokovi još su jedna, novija nadogradnja na standardne konvolucijske blokove. Nastali su primjenjući neke arhitekturalne posebnosti modela temeljenih na vizualnim transformerima (ViT-ovi)[12] na ResNet blok. Vizualni transformeri alternativna su porodica arhitektura neuronskih mreža za računalni vid koja se ne temelji na konvolucijskim mrežama. Nastali su velikom inspiracijom iz obitelji transformer arhitektura za obradu prirodnog jezika (*engl.* NLP — Natural Language Processing) kao što su BERT (*engl.* Bidirectional Encoder Representations from Transformers) [13] i GPT (*engl.* Generative Pretrained Transformer) [14].

Tajna njihovog uspjeha leži u revolucionarnom mehanizmu pažnje (*engl.* attention) [15] koji im omogućava da uspješno nauče jako detaljne relacije u podatcima. Najveći problem vizualnih transformera jest što zahtijevaju jako velik broj primjera za treniranje, što zahtijeva velike računalne resurse ako bismo htjeli istrenirati jedan takav model. Iz tog razloga arhitekture temeljene na izravnim transformerima neće biti korištene za izradu uklanjača šuma u sklopu ovog rada.

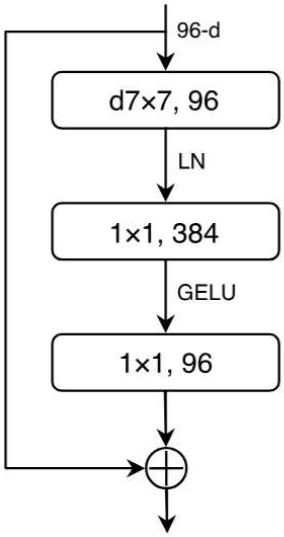
Razlike ConvNeXt bloka u odnosu na ResNet:

- prijelaz iz “bottleneck” arhitekture u “inverted bottleneck” — umjesto da su ulaz i izlaz velike dimenzije, a centralni sloj male, vrijedi obrat
- aktivacijska funkcija ReLU prelazi u GELU [16] — GELU ima kontinuirane gradi-

jente, što poboljšava učenje

- veličina konvolucijske jezgre raste s 3 na 7 — konvolucijski blok ima veće receptivno polje, tj. više susjednih piksela ima utjecaja pri ekstrakciji značajki
- sloj batch norm prelazi u layer norm koji se pojavljuje samo jedanput

Ispostavilo se da te izmjene u nekim slučajevima vode do poboljšanja performanci. Prikaz ConvNeXt bloka nalazi se na slici 3.8.



Slika 3.8. ConvNeXt blok

Sada ćemo usporediti metrike modela koji se razlikuju samo po postavljenoj konfiguraciji blokova:

- standardni konvolucijski blok
- resnet-18
- resnet-34
- resnet-108
- convnext-tiny

Različite ResNet konfiguracije razlikuju se po broju slojeva. Naime, moguće je povećati broj slojeva u bloku, što povećava broj izvučenih značajki iz slika, a time potencijalno i performance konačnog modela.

3.3.1. Analiza na skupu podataka jednostavnije scene

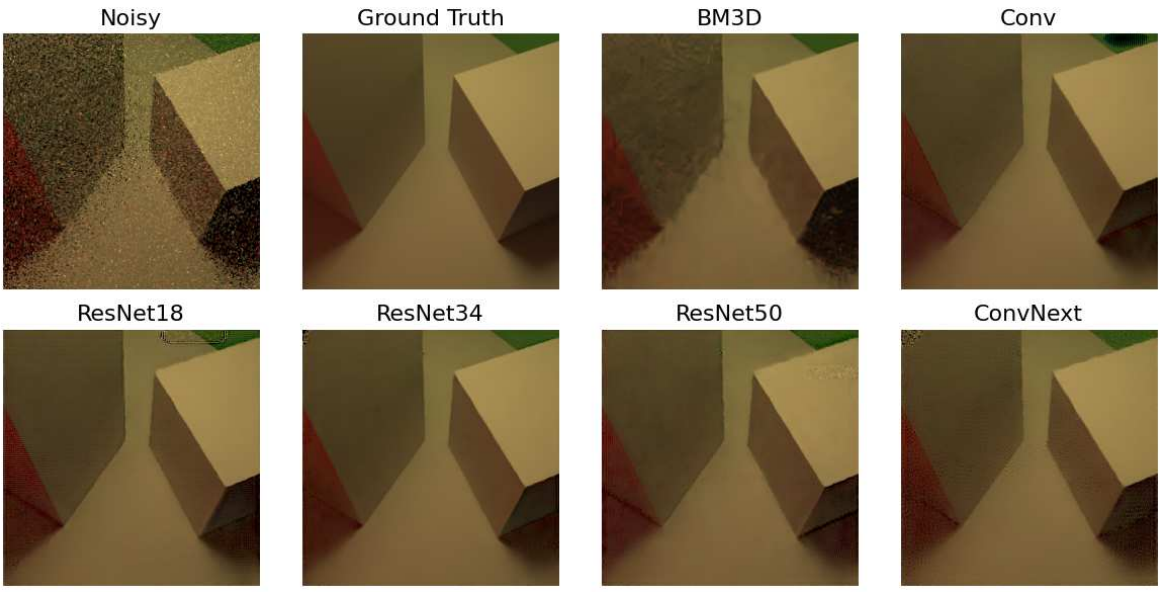
Prvog ćemo pogledati kakve rezultate daju različiti enkoderi na jednostavnijoj sceni. Tablica performanci modela s različitim enkoderima dana je u tablici 3.3.

Model	Skup za treniranje			Skup za testiranje		
	Gubitak (L1)	PSNR	SSIM	Gubitak (L1)	PSNR	SSIM
Stand. konv.	3.66e-3	39.32	0.964	3.68e-3	38.98	0.964
BM3D	6.16e-3	35.45	0.944	6.19e-3	35.34	0.943
resnet-18	6.03e-3	35.06	0.869	6.02e-3	34.9	0.866
resnet-34	6.48e-3	33.56	0.836	6.47e-3	33.43	0.835
resnet-50	8.1e-3	32.54	0.81	8.07e-3	32.35	0.807
convnext-tiny	9.56e-3	31.52	0.738	9.58e-3	31.38	0.737

Tablica 3.3. Usporedba modela s različitim enkoderima

Prvo što možemo primijetiti je da su performance praktički jednake na skupu za treniranje i testiranje. To je dobar pokazatelj jer to znači da model dobro generalizira, ali i da ima kapaciteta da nauči odšumiti i kompleksniju scenu. Sljedeće, možemo primijetiti da su performance najviše kod standardne konvolucije, tj. bez arhitekturnih izmjena. Razumno objašnjenje tog rezultata jest da su ResNet i ConvNeXt napredniji i doprinose većem kapacitetu modela, što znači da bi bilo potrebno imati više podataka u skupu za treniranje, ali da bismo onda mogli očekivati i bolje rezultate.

Zanimljivo je da su metrike dobivene usporedbom referentne slike te slike odšumljene BM3D algoritmom prilično dobre, usporedive su s najboljim testiranim modelima. Kako bismo dobili bolji uvid u kvalitetu modela, valja napraviti osvrt na vizualnu usporedbu rezultata. Vizualna usporedba modela sa svim isprobanim blokovima nalazi se na slici 3.9.

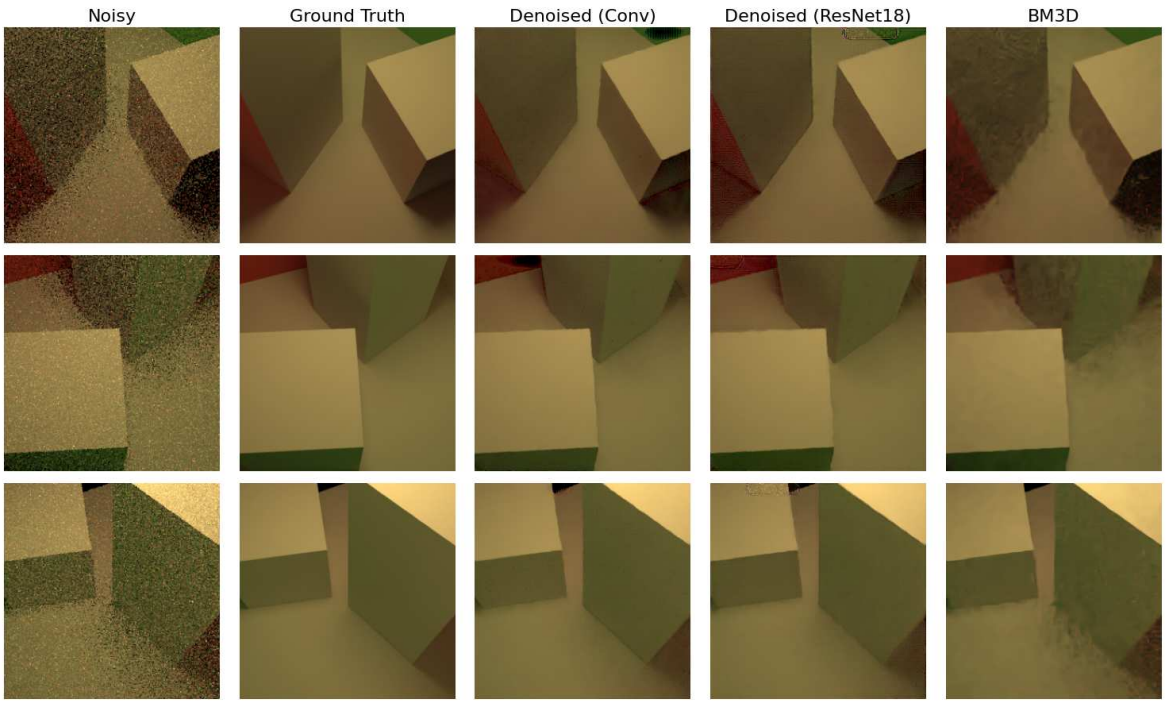


Slika 3.9. Usporedba modela s različitim blokovima s referentnom slikom i BM3D treniranih na sceni "Cornell Box"

Ovdje vidimo da model treniran na normalnim konvolucijskim blokovima stvarno daje vizualno najkvalitetnije rezultate. ResNet modeli su poprilično blizu po kvaliteti, iako se kod njih vide blagi vizualni artefakti.

BM3D je uvjerljivo najlošiji. Nije uspio dobrim dijelom ukloniti šum, uz cijenu znatne zamućenosti. U pojedinim dijelovima napravio je pogotovo loš posao, kao što se to vidi kod sjene koju je pretjerano zamračio. Nijedan uklanjač šuma ostvaren dubokim učenjem nije imao problema s tim.

Iako su svi uklanjači šuma ostvareni neuronskom mrežom dali dobre rezultate, kod ResNet i ConvNeXt modela može se u nekim slučajevima vidjeti blaga anomalija. Može se primijetiti da se tijekom treniranja modela postupno smanjuje tako da nakon još nekoliko iteracija vjerojatno bi nestala. Međutim, ne pojavljuje se u svakom primjeru. Na slici 3.10. u stupcu resnet18 može se vidjeti više primjera, a na nekima se uopće ne vidi anomalija.



Slika 3.10. Usporedba modela konvolucijskog (2. red) i resnet18 (3. red) blokovima

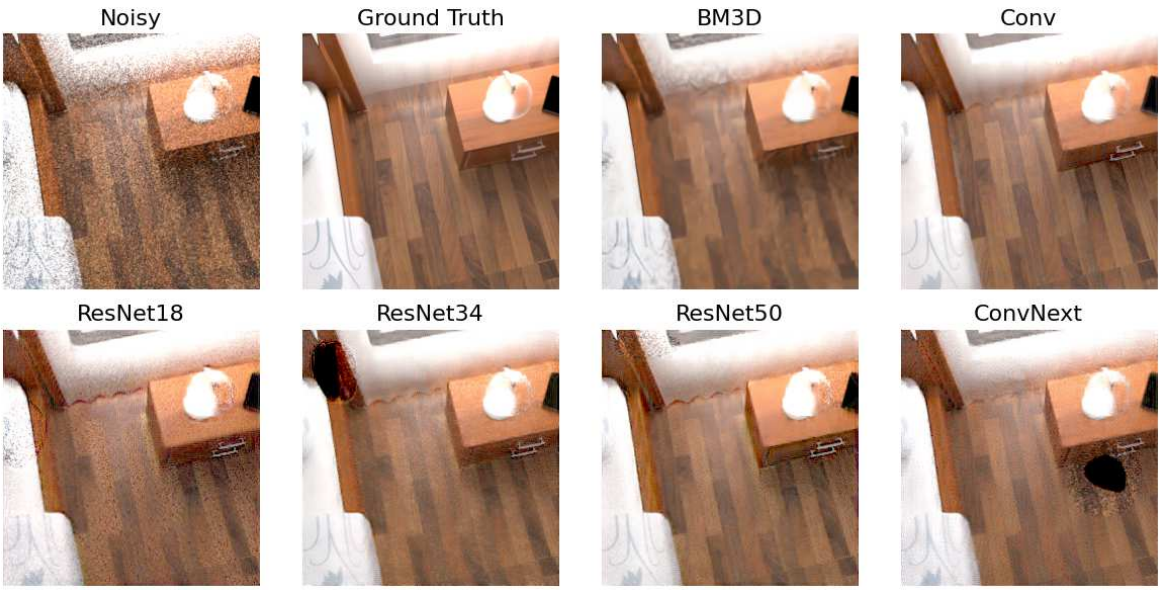
3.3.2. Analiza na skupu podataka kompleksnije scene

Sada ćemo pogledati kako se ponaša model treniran na skupu podataka generiranom iz scene “Bedroom”. Metrike se nalaze u tablici 3.4.

Model	Skup za treniranje			Skup za testiranje		
	Gubitak (L1)	PSNR	SSIM	Gubitak (L1)	PSNR	SSIM
Stand. konv.	20.34e-3	32.80	0.873	20.94e-3	32.98	0.873
BM3D	24.69e-3	31.79	0.779	25.18e-3	31.62	0.776
resnet-18	24.69e-3	25.82	0.509	25.98e-3	25.42	0.520
resnet-34	21.15e-3	26.93	0.551	20.39e-3	26.96	0.573
resnet-50	18.86e-3	28.93	0.601	18.74e-3	29.03	0.614
convnext-tiny	24.60e-3	26.69	0.458	24.41e-3	27.08	0.476

Tablica 3.4. Usporedba modela s različitim enkoderima treniranih na sceni “Bedroom”

Vizualna usporedba se nalazi na slici 3.11.



Slika 3.11. Usporedba modela s različitim blokovima s referentnom slikom i BM3D treniranih na sceni “Bedroom”

U ovom slučaju uklanjač šuma nije ostvario zadovoljavajuće rezultate u usporedbi s prijašnjom scenom. Metrike su kod svih enkodera bile mnogo lošije, pogotovo PSNR. SSIM je najviši kod standardne konvolucije te kod resnet-50 bloka, a vizualna usporedba potvrđuje da su oni stvarno najbolji. Razlog zašto je veći ResNet blokovi daju bolje rezultate jest što izvlače veći broj značajki iz ulaza, neuronska mreža ima više informacija o ulaznoj slici pa je rekonstruirani šum bolji.

BM3D također daje lošije rezultate. Po metrikama je bolji od većine varijanti neuronskih mreža, ali vizualnom inspekcijom vidi se da su detalji u slici mnogo zamućeni, što je vrlo nepoželjno. Standardna konvolucija i resnet-50 blok bolje su očuvali detalje.

Zaključak: u jednostavnoj sceni (uglavnom niske frekvencije u konačnom prikazu) standardna konvolucija i manji ResNet blokovi daju bolje rezultate; u kompleksnijoj sceni (uglavnom visoke frekvencije u konačnom prikazu) standardna konvolucija i veći ResNet blokovi daju bolje rezultate. BM3D i ConvNeXt zaostaju u oba slučaja.

3.4. Utjecaj predtreniranja

U dubokom učenju, jedna od negativnih činjenica je da kada promijenimo bilo koji od parametara modela, moramo model ponovno trenirati od početka. Pri tome je izgubljena

apsolutno sva moguća informacija o ulaznim podatcima, bila ona dobra ili loša. Predtreniranje je praksa treniranja modela počevši od nekog prethodno treniranog modela. Razlog zašto bismo to radili jest iskorištavanje korisne informacije koju je model već naučio. U računalnom vidu česta je praksa treniranja modela predtreniranog na skupu podataka "ImageNet" [17].

U ovom konkretnom slučaju predtreniranjem se nadamo postići bržu konvergenciju modela (tj. brže treniranje, što je poželjno za proces treniranja) i bolje performance modela.

3.4.1. Analiza na skupu podataka jednostavnije scene

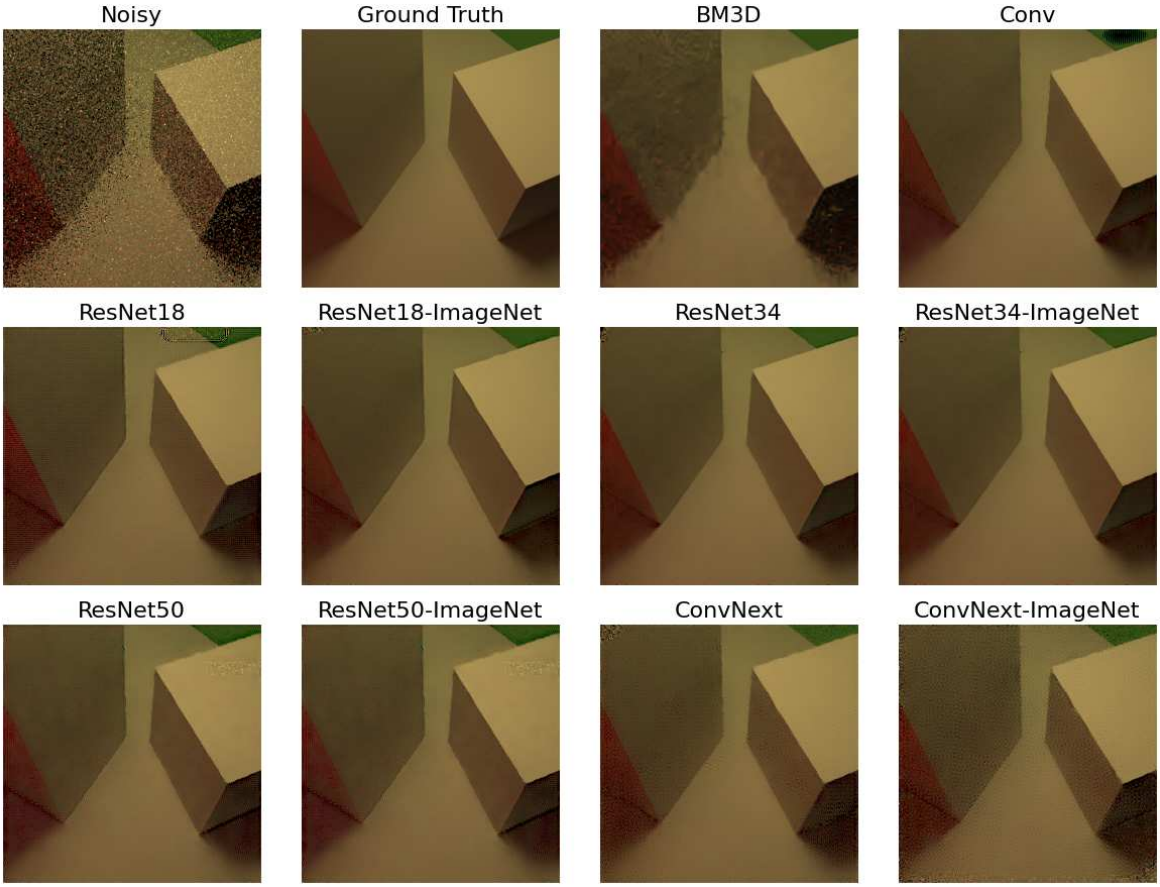
Tablica performanci različitih modela na sceni "Cornell Box" dana je u tablici 3.5.

Model	Skup za treniranje			Skup za testiranje		
	Gubitak (L1)	PSNR	SSIM	Gubitak (L1)	PSNR	SSIM
BM3D	24.69e-3	31.79	0.779	25.18e-3	31.62	0.776
resnet-18	24.69e-3	25.82	0.509	25.98e-3	25.42	0.520
resnet-18 (ImageNet)	20.94e-3	27.91	0.618	20.97e-3	27.94	0.612
resnet-34	21.15e-3	26.93	0.551	20.39e-3	26.96	0.573
resnet-34 (ImageNet)	19.68e-3	28.14	0.536	19.63e-3	28.12	0.548
resnet-50	18.86e-3	28.93	0.601	18.74e-3	29.03	0.614
resnet-50 (ImageNet)	14.36e-3	30.59	0.687	14.52e-3	30.68	0.701
convnext-tiny	24.60e-3	26.69	0.458	24.41e-3	27.08	0.476
convnext-tiny (ImageNet)	20.84e-3	28.26	0.544	20.79e-3	28.54	0.562

Tablica 3.5. Kvantitativna usporedba nepredtreniranih i predtreniranih modela na sceni "Cornell Box"

Vidljivo je da su sve kvantitativne metrike više za predtrenirani model, s iznimkom ConvNeXt modela.

Prikaz odšumljenih slika ovim modelima se nalazi na slici 3.12.



Slika 3.12. Vizualna usporedba nepredtreniranih i predtreniranih modela na sceni “Cornell Box”

Vizualno se ne može utvrditi razlika između odšumljenih prikaza uklanjačima šuma koji su prethodno predtrenirani i koji nisu.

Ovaj rezultat nije neočekivan jer uklanjači i bez predtreniranja daju jako dobre rezultate. Osim toga, ova scena je vrlo jednostavna i kao takva nema predmeta iz svakodnevnog života.

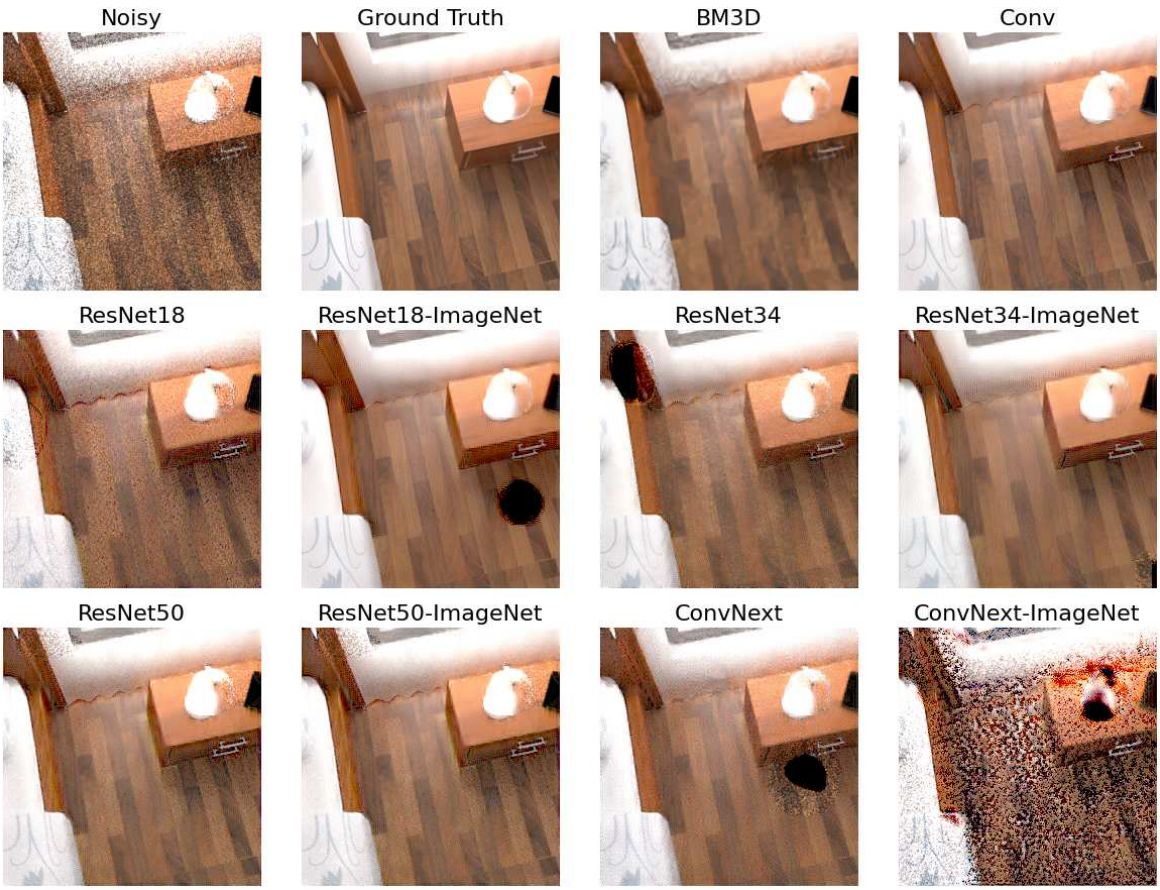
3.4.2. Analiza na skupu podataka kompleksnije scene

Skup podataka ImageNet pun je slika s takvim predmetima, stoga očekujemo da bi to moglo biti od veće koristi pri odšumljivanju scena koje su bliže onima iz stvarnog života, kao što je scena “Bedroom”. U tablici 3.6. nalazi se kvantitativna analiza modela treniranih na sceni “Bedroom” i predtreniranih na skupu podataka “ImageNet”.

Model	Skup za treniranje			Skup za testiranje		
	Gubitak (L1)	PSNR	SSIM	Gubitak (L1)	PSNR	SSIM
BM3D	6.16e-3	35.45	0.944	6.19e-3	35.34	0.943
resnet-18	6.03e-3	35.06	0.869	6.02e-3	34.9	0.866
resnet-18 (ImageNet)	4.14e-3	37.02	0.909	4.16e-3	36.76	0.907
resnet-34	6.48e-3	33.56	0.836	6.47e-3	33.43	0.835
resnet-34 (ImageNet)	4.7e-3	35.71	0.894	4.73e-3	35.4	0.89
resnet-50	8.1e-3	32.54	0.81	8.07e-3	32.35	0.807
resnet-50 (ImageNet)	7.54e-3	34.04	0.839	7.54e-3	34.04	0.838
convnext-tiny	9.56e-3	31.52	0.738	9.58e-3	31.38	0.737
convnext-tiny (ImageNet)	11.99e-3	30.36	0.673	11.97e-3	30.26	0.675

Tablica 3.6. Kvantitativna usporedba nepredtreniranih i predtreniranih modela na sceni “Bedroom”

Vizualna usporedba se nalazi na slici 3.13.



Slika 3.13. Vizualna usporedba nepredtreniranih i predtreniranih modela na sceni “Bedroom”

Metrike pokazuju malo poboljšanje, ali ne toliko značajno koliko smo se nadali. Vizualno se može primijetiti da su bolje očuvani detalji u nekim slučajevima, kao što su resnet18 i resnet34, ali ne u značajnijoj mjeri. Od svih najviše odskače predtrenirani ConvNeXt koji nije dobro uspio rekonstruirati šum. Izgleda da predtreniranje u ovom slučaju uzrokuje divergenciju.

Zaključak: skup podataka “ImageNet” na kojem su ovi modeli predtrenirani nosi korisne informacije koje model može iskoristiti, što se vidi po višim kvantitativnim metrikama, ali vizualnom usporedbom utvrđeno je da su razlike vrlo male. Poboljšanje nije značajno ni kod jednostavnih prikaza ni kod kompleksnijih koji su komponirani od elemenata koji se nalaze i u skupu podataka “ImageNet”.

3.5. Sažetak konačnog modela

Na temelju analiza provedenih u prijašnjim poglavljima, najbolji model je U-Net s konvolucijskim blokovima ili ResNet18 blokovima s enkoderom predtreniranim na skupu podataka "ImageNet", treniran s kombiniranim gubitkom koji se sastoji od Charbonnierovog gubitka i gubitka ovisnog o rubovima.

Ovaj model će biti korišten u sljedećem poglavlju za ispitivanje raznovrsnosti skupa za treniranje te za konačnu analizu.

3.6. Utjecaj raznovrsnosti skupa za treniranje

Do sada su promatrane performance modela na dvama odvojenim scenama, a evaluacijske metrike su računate na slikama renderiranim iz tih scena. Međutim, takva evaluacija ima manu: ne daje nam odgovor je li uklanjač šuma uspješan i na slikama renderiranim iz drugih scena.

Kako bi dali odgovor na to pitanje, napravit ćemo analizu uklanjača šuma treniranog na obje scene koje su do sada korištene za treniranje ("Cornell Box", "Bedroom"), usporediti metrike s modelima koji su trenirani na samo jednoj od tih scena, a metrike će biti izračunate evaluacijom na te obje scene te još jedne na kojoj nismo trenirali podatke ("Bathroom"). Ispitivanje performanci na sceni na kojoj nije treniran model **ključno je za ispitivanje koliko model dobro generalizira**.

Metrike modela se nalaze u tablici 3.7. Hiperparametri modela koji su korišteni su najbolji hiperparametri dobijeni u prijašnjim poglavljima.

Model	Cornell Box		Bedroom		Bathroom	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
Cornell Box	**39.35**	**0.977**	14.56	0.276	19.13	0.383
Bedroom	25.16	0.688	**36.08**	**0.922**	31.82	0.896
Cornell Box + Bedroom	**35.04**	**0.913**	**32.46**	**0.843**	31.94	0.848

Tablica 3.7. Usporedba modela treniranih na različitim kombinacijama scena, evaluiranih na svim scenama

Metrike koje su evalirane nad scenom koja je također bila u skupu za treniranje za taj model su masno otisnute. Kako bi i očekivali, one su više jer su slike iz te scene bile u skupu za treniranje; uklanjač šuma je izravno učio kako ukloniti šum iz tih scena.

Metrike evaluirane nad scenama koje nisu bile u skupu za treniranje su zanimljivije za interpretaciju jer nam govore koliko dobro neuronska mreža kojom je implementiran uklanjač šuma generalizira.

Model koji je treniran na jednostavnoj "Cornell Box" sceni ima jako loše performance na obje kompleksnije scene. To je očekivano jer su scene međusobno vrlo različite, a "Cornell Box" scena nema prikaza koji imaju mnogo detalja pa odšumljivač nije imao priliku naučiti kako izgleda šum kod složenih, visokofrekventnih prikaza.

U slučaju kada model treniramo samo na kompleksnijoj "Bedroom" sceni, performance koje dobijemo na također kompleksnoj "Bathroom" sceni daleko su više, gotovo usporedive s metrikama evaluiranim na sceni na kojoj je model treniran. To su odlične vijesti jer je to dokaz da **uklanjač šuma ima kapacitet za generalizaciju**, a dodavanjem daljnjih scena bi se performance samo poboljšavale. Vrijedno je također osvnuti se na činjenicu da evaluacija na sceni jednostavnijoj sceni "Cornell Box" daje lošije metrike, ali to nije neočekivano jer je ta scena toliko drugačija od one na kojoj smo trenirali model. Ohrabljujuće je to što je pad performanci manji nego u slučaju kada je model treniran na jednostavnoj sceni, a želimo odšumiti prikaz iz složene scene.

U zadnjem slučaju smo model trenirali i na jednostavnoj i na kompliciranoj sceni

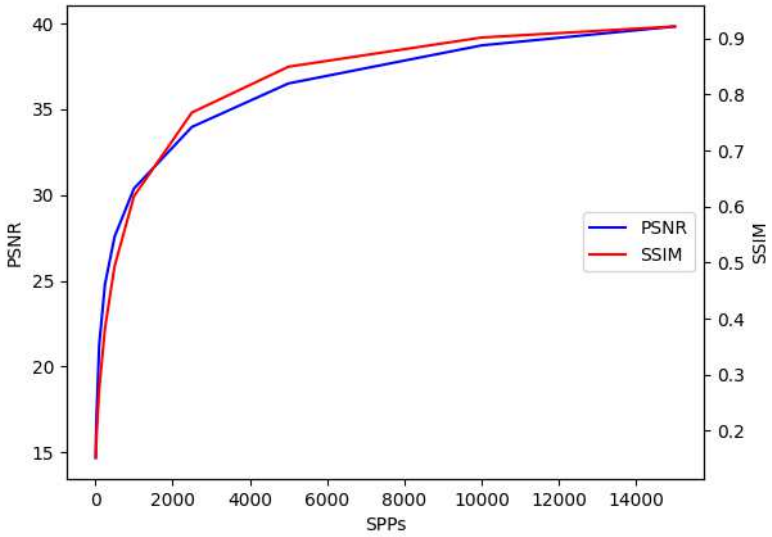
“Cornell Box” i “Bedroom”. U ovom slučaju imamo poprilično visoke performance na sva tri scene. Nisu visoke kao u nekim pojedinačnim slučajevima gdje je model treniran samo na jednoj sceni, ali to je posljedica toga što je model treniran na manje slika iz svake scene zbog brzine treniranja.

Zaključak: uklanjač šuma ima zadovoljavajuće performance na scenama na kojima nije treniran, ako je treniran na dovoljno raznolikim podatcima.

3.7. Konačna evaluacija

Sve varijacije modela su ispitane. Još je preostalo pokazati praktično ubrzanje dobijeno ovim pristupom.

Kao što smo već definirali, kvaliteta prikaza renderiranog tehnikom praćenja puta ovisi o broju zraka po pikselu (SPP). Vrijeme renderiranja je gotovo potpuno linearno s brojem zraka po pikselu. Međutim, kvaliteta se ne povećava linearno nego logaritamski što se vidi na slici 3.14.



Slika 3.14. Prikaz PSNR i SSIM metrika u ovisnosti o SPP-u, usporedba s referentnom slikom renderiranom s SPP=20000, scena “Bedroom”

Poboljšanje kvaliteta prikaza poprilično brzo usporava pri povećanju zraka.

Kako bi uklanjač šuma bio najkorisniji, potrebno je odabrati dobar SPP pri kojem

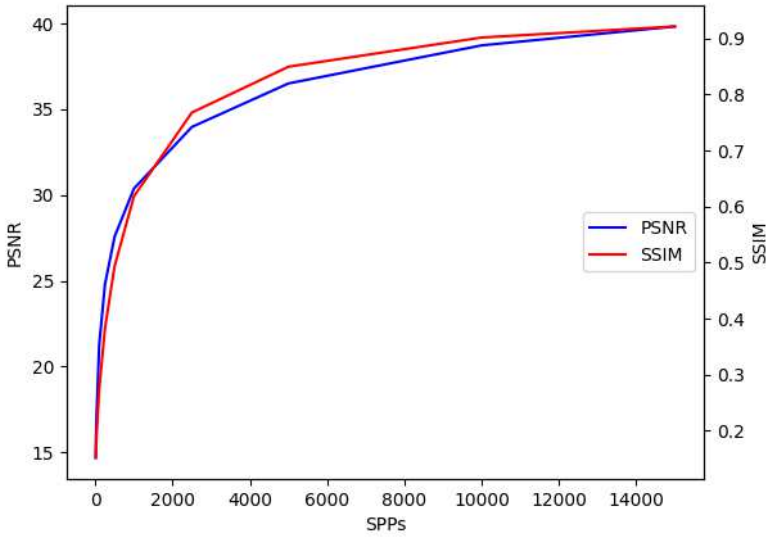
ćemo zaustaviti renderiranje slike i umjesto daljnjeg renderiranja je odšumiti. Što raniji trenutak odaberemo, to ćemo uštedjeti više vremena na renderiranju, ali će ulazna slika u uklanjač biti lošije kvalitete te je on potencijalno neće moći dobro odšumiti. Što kasniji trenutak odaberemo, to će ulaz biti pogodniji za odšumljivanje, ali će se utrošiti više vremena na renderiranje i vremenska ušteda bit će manja.

Sudeći po grafu na slici 3.14., zlatna sredina nalazi se za vrijednosti SPP-a koje odgovaraju između 30 i 35 dB vrijednosti PSNR. U tom slučaju možemo za samo 1/10 ukupnog vremena renderiranja visokokvalitetnog prikaza dobiti kvalitetne rezultate ako koristimo kvalitetan uklanjač šuma.

Međutim, još je potrebno provjeriti vremenske troškove korištenja uklanjača šuma. Naime, uklanjač šuma temeljen na dubokom učenju zahtijeva unaprijedan prolaz kroz neuronsku mrežu što ima svoj određeni vremenski trošak. Dobra vijest je da je taj vremenski trošak vrlo malen za razliku od samog treniranja neuronske mreže, pogotovo ako imamo sklopovsku podršku koja može ubrzati proces — a ona je gotovo uvijek prisutna jer se ona također koristi i za samo renderiranje.

Kako bi stavili u perspektivu koliko vremena uštedimo uklanjanjem šuma, pogledat ćemo vizualizaciju na slici 3.15. Po stupcima vidimo ukupno vrijeme dobivanja konačnog izlaza koje uključuje vrijeme renderiranja (svijetlo plavo) i vrijeme uklanjanja šuma (tamno plavo). Zadnji stupac je visokokvalitetna referentna slika na kojoj je treniran uklanjač šuma. Nju nema potrebe odšumljavati jer je za naše potrebe smatramo nao zlatni standard koji nema šuma; ona ga teoretski još uvijek sadrži, ali je za naše potrebe zanemariv. Predzadnji stupac sadrži prikaz čiji je PSNR naspram konačne slike 38 dB što je približno ekvivalentno perfromancama našeg uklanjača šuma; izlaz uklanjača šuma daje prikaze koji su približno kvalitetne kao ovaj prikaz. Ostali stupci prikazuju vrijeme dobivana kombiniranim pristupom renderiranja i odšumljivanja.

Kao što možemo vidjeti, vrijeme odšumljivanja je jednako za bilo koji broj zraka po pikselu kod renderiranja prikaza. Posljedica toga jest da je za **za srednje količine SPP-a biti najbolji omjer utrošenog vremena i kvalitete izlaza**; neće biti potrošeno previše vremena na renderiranje, a uklanjač šuma će dobiti dovoljno kvalitetan prikaz da ga odšumi do kraja.



Slika 3.15. Usporedba kombiniranih vremena renderiranja i uklanjanja šuma na sceni "Bedroom"

3.8. Buduće nadogradnje

Dosadašnjom analizom pokazano je da je moguće ostvariti kvalitetan uklanjač šuma temeljen na dubokom učenju koji je nedvojbeno kvalitetniji od najboljih klasičnih metoda uklanjanja šuma. Međutim, implementacijom u ovom radu nismo riješili sve probleme.

Najveći problem koji je ostao u nekoj mjeri neriješen je **generalizacija**. Treniranjem modela na različitim scenama ju je poboljšalo, ali opet nije potpuno otklonilo problem. To se u principu još uvijek rješava proširivanjem skupa podataka i predprocesiranje.

Sljedeće što bi se dalo isprobati je potpuna promjena modela. Model koji sam ja koristio zasniva se na konvolucijskim neuronskim mrežama. Ranije je kao alternativna spomenuta alternativna skupina modela u području računalnog vida — vizualni transformeri (ViT-ovi). Ovakvi modeli zahtijevaju mnogo više ulaznih podataka, ali ako je podataka dovoljno, takav model može dati potencijalno kvalitetnije rezultate koji bolje generaliziraju. Najveći problem ovakvih modela jest što se zbog potrebe za iznimno velikim skupovima podataka za učenje jako sporo uče, a bolji rezultati nisu garantirani.

Kvalitetu odšumljivanja moguće je popraviti generativnim modelima. Arhitektura dubokih generativnih neuronskih mreža **GAN (Generative Adversarial Network)**

[18] osmišljena je sa specifičnim ciljem generiranja uvjerljivih izlaznih slika. Mreža se sastoji od 2 dijela: generatora i diskriminatora. Generator generira izlazne slike, a diskriminator pokušava odgonetnuti je li slika donesena na ulaz prava slika ili generirana (donesena iz generatora); generator se uči generirati uvjerljivije slike kako bi pokušao "prevariti" diskriminator, a diskriminator paralelno uči bolje kako razaznati umjetne slike generatora. Rezultat ove igre povratna je veza koja tjera generator da nauči generirati kvalitetne slike. Ovaj koncept može se koristiti za poboljšanje uklanjača šuma na način da postavimo generator kao uklanjač šuma, a diskriminator kao konvolucijsku neuronsku mrežu koja pokušava naučiti je li odšumljena slika izlaz uklanjača šuma (generatora) ili referentna slika (slika generirana s visokim SPP-om). U tom procesu uklanjač šuma naučit će generirati bolje slike kako bi prevario diskriminator.

Moguće nadogradnje su brojne. Kako područje dubokog učenja u sklopu računalnog vida još uvijek napreduje, za očekivati je da će u bliskoj budućnosti biti otkriveni novi koncepti koji bi bili korisni za ovu konkretnu svrhu.

4. Zaključak

U ovom radu istražena je izrada i analiza uklanjača šuma za slike generirane postupkom praćenja puta, temeljenog na dubokoj neuronskoj mreži arhitekture U-Net. Cilj rada nije bio postići apsolutno najbolji mogući uklanjač šuma, već sustavno analizirati utjecaj različitih komponenti sustava na konačnu kvalitetu rezultata: ulaznih značajki, funkcija gubitka, arhitekturalnih izmjena, predtreniranja i raznovrsnosti skupa za treniranje.

Pokazano je da dodatni pomoćni spremnici (difuzna refleksivnost, dubinska mapa i mapa normala) značajno pridonose kvalitetu uklanjanja šuma u odnosu na model koji koristi samo zašumljenu sliku te da se ti podatci mogu jeftino dobiti iz pogona za renderiranje praćenjem puta. Analizom različitih funkcija gubitka utvrđeno je da L1 i Charbonnierov gubitak, osobito u kombinaciji s komponentom svjesnom rubova, daju vizualno najuvjerljivije rezultate, dok gubitak temeljen na frekvencijskoj domeni u trenutačnom obliku nije pokazao konkurentne performance. Pri ispitivanju arhitekturalnih izmjena standardni konvolucijski blokovi U-Neta na promatranom su skupu podataka dali bolje rezultate od ResNet i ConvNeXt blokova, što ukazuje na to da napredniji blokovi zahtijevaju veće i raznovrsnije skupove za treniranje kako bi opravdali svoj veći kapacitet. Predtreniranje na skupu ImageNet pokazalo se korisnim za ResNet enkodere prema kvantitativnim metrikama, no vizualne razlike u odnosu na modele bez predtreniranja bile su male.

Usporedba s klasičnim algoritmom BM3D pokazala je jasnu prednost pristupa temeljenih na dubokom učenju: dok BM3D ostavlja zamućene rubove i pretjerano zatamnjuje pojedine dijelove slike, modeli temeljeni na U-Netu dobro čuvaju detalje i strukturu scene. Također je pokazano da je kvantitativna evaluacija pomoću metrika PSNR i SSIM nužna, ali nije dovoljna; vizualna usporedba ostaje ključna jer same metrike mogu propustiti suptilne pojave preostalog šuma ili nepoželjnih artefakata.

Buduće nadogradnje uključuju proširenje skupa podataka većim brojem raznovrsnih scena, istraživanje hibridnih funkcija gubitka koje bolje kombiniraju prostornu i frekvencijsku informaciju te ispitivanje arhitektura temeljenih na vizualnim transformerima i mehanizmu pažnje za potencijalno daljnje povećanje kvalitete uklanjanja šuma.

Literatura

[1] S. Bako, T. Vogels, B. McWilliams, M. Meyer, J. Novák, A. Harvill, P. Sen, T. DeRose, i F. Rousselle, "Kernel-predicting convolutional networks for denoising monte carlo renderings", u *ACM Transactions on Graphics*, sv. 36, br. 4, 2017., str. 1–14.

[2] C. R. A. Chaitanya, A. S. Kaplanyan, C. Schied, M. Salber, A. Lefohn, D. Nowrouzezahrai, i T. Aila, "Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder", *ACM Transactions on Graphics*, sv. 36, br. 4, str. 1–12, 2017.

[3] J. T. Kajiya, "The rendering equation", *ACM SIGGRAPH Computer Graphics*, sv. 20, br. 4, str. 143–150, 1986.

[4] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Gesto, R. Pacanowski *et al.*, "Mitsuba 3 renderer", 2022., https://mitsuba-renderer.org.

[5] O. Ronneberger, P. Fischer, i T. Brox, "U-net: Convolutional networks for biomedical image segmentation", *arXiv preprint arXiv:1505.04597*, 2015.

[6] K. Zhang, W. Zuo, Y. Chen, D. Meng, i L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising", *IEEE Transactions on Image Processing*, sv. 26, br. 7, str. 3142–3155, 2017.

[7] D. P. Kingma i J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[8] K. Dabov, A. Foi, V. Katkovnik, i K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering", *IEEE Transactions on Image Processing*,

sv. 16, br. 8, str. 2080–2095, 2007.

[9] Z. Wang, A. C. Bovik, H. R. Sheikh, i E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity”, *IEEE Transactions on Image Processing*, sv. 13, br. 4, str. 600–612, 2004.

[10] K. He, X. Zhang, S. Ren, i J. Sun, “Deep residual learning for image recognition”, u *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016., str. 770–778.

[11] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, i S. Xie, “A convnet for the 2020s”, u *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022., str. 11 976–11 986.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, i N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale”, *arXiv preprint arXiv:2010.11929*, 2020.

[13] J. Devlin, M.-W. Chang, K. Lee, i K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding”, *arXiv preprint arXiv:1810.04805*, 2019.

[14] A. Radford, K. Narasimhan, T. Salimans, i I. Sutskever, “Improving language understanding by generative pre-training”, 2018.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, i I. Polosukhin, “Attention is all you need”, *Advances in Neural Information Processing Systems*, sv. 30, 2017.

[16] D. Hendrycks i K. Gimpel, “Gaussian error linear units (gelus)”, *arXiv preprint arXiv:1606.08415*, 2016.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, i L. Fei-Fei, “Imagenet: A large-scale hierarchical image database”, u *IEEE Conference on Computer Vision and Pattern Recognition*, 2009., str. 248–255.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, i Y. Bengio, "Generative adversarial nets", u *Advances in Neural Information Processing Systems*, sv. 27, 2014.

Privitak A: Kod

Sav kod napravljen za ostvarenje ovog rada (uključujući među ostalim i definiciju korištene neuronske mreže, kod za njezino treniranje, evaluiranje te kod za generiranje skupa podataka za treniranje) kao i upute za njegovo korištenje nalazi se na sljedećoj poveznici: `https://github.com/spinzed/rt-denoiser`