SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1164

ODREĐIVANJE KORESPONDENCIJA U SLIKAMA PRIMJENOM PAŽNJE

Dominik Šarić

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1164

ODREĐIVANJE KORESPONDENCIJA U SLIKAMA PRIMJENOM PAŽNJE

Dominik Šarić

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 10. ožujka 2023.

ZAVRŠNI ZADATAK br. 1164

Pristupnik:	**Dominik Šarić (0036532348)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Određivanje korespondencija u slikama primjenom pažnje**

Opis zadatka:

Stereoskopska rekonstrukcija važan je zadatak računalnog vida s mnogim zanimljivim primjenama. U novije vrijeme posebno su zanimljivi postupci koji kombiniraju konvolucijska ugrađivanja te korespondenciju uz pomoć slojeva pažnje. Ovaj rad razmatra mogućnost primjene takvih postupaka u eksperimentima s ograničenim računskim budžetom. U okviru rada, potrebno je odabrati okvir za automatsku diferencijaciju te upoznati biblioteke za rukovanje matricama i slikama. Proučiti i ukratko opisati postojeće pristupe za stereoskopsku rekonstrukciju koji se temelje na dubokom učenju. Odabrati slobodno dostupni skup slika te oblikovati podskupove za učenje, validaciju i testiranje. Predložiti prikladnu arhitekturu za stereoskopsku rekonstrukciju primjenom slojeva pažnje. Uhodati postupke učenja modela i validiranja hiperparametara. Primijeniti naučene modele te prikazati i ocijeniti postignutu točnost. Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 9. lipnja 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1164

Određivanje korespondencija u slikama primjenom pažnje

Dominik Šarić

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 10. ožujka 2023.

ZAVRŠNI ZADATAK br. 1164

Pristupnik:	**Dominik Šarić (0036532348)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Određivanje korespondencija u slikama primjenom pažnje**

Opis zadatka:

Stereoskopska rekonstrukcija važan je zadatak računalnog vida s mnogim zanimljivim primjenama. U novije vrijeme posebno su zanimljivi postupci koji kombiniraju konvolucijska ugrađivanja te korespondenciju uz pomoć slojeva pažnje. Ovaj rad razmatra mogućnost primjene takvih postupaka u eksperimentima s ograničenim računskim budžetom. U okviru rada, potrebno je odabrati okvir za automatsku diferencijaciju te upoznati biblioteke za rukovanje matricama i slikama. Proučiti i ukratko opisati postojeće pristupe za stereoskopsku rekonstrukciju koji se temelje na dubokom učenju. Odabrati slobodno dostupni skup slika te oblikovati podskupove za učenje, validaciju i testiranje. Predložiti prikladnu arhitekturu za stereoskopsku rekonstrukciju primjenom slojeva pažnje. Uhodati postupke učenja modela i validiranja hiperparametara. Primijeniti naučene modele te prikazati i ocijeniti postignutu točnost. Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 9. lipnja 2023.

Zahvaljujem mentoru prof. dr. sc. Siniši Šegviću na pomoći, savjetima i ohrabrenju tijekom pisanja ovog rada. Zahvaljujem se obitelji i prijateljima na bezuvjetnoj potpori tijekom obrazovanja.

SADRŽAJ

1. Uvod

Računalni vid je područje koje se bavi razumijevanjem i tumačenjem vizualnih informacija iz digitalnih slika ili videozapisa. Danas se u području računalnog vida uz klasične metode koriste i metode umjetne inteligencije, pogotovo metode dubokog učenja. Neke od primjena računalnog vida bi bile u područjima autonomnih vozila, sustava nadzora, medicinskom slikanju, proširenoj stvarnosti i mnogim drugima. U ovom istraživanju fokusiramo se na problem korespondencije te nastojimo riješiti taj problem primjenom konvolucijskih ugrađivanja i slojeva pažnje. Detaljno opisujemo proces stereoskopske rekonstrukcije koji uključuje metode epipolarne geometrije, kalibracije i rektifikacije. Stereoskopska rekonstrukcija je tehnika koja nam omogućuje stvaranje trodimenzionalne reprezentacije objekata koristeći dvije ili više scena. Stereoskopska rekonstrukcija se primjenjuje u područjima računalnog vida, robotike, geodezije, kartografije i mnogim drugim područjima.

Pružamo pregled osnovnih pojmova i postupaka dubokog učenja te predstavljamo arhitekturu modela kojim se pokušava riješiti problem korespondencije na paru stereoskopskih slika. Također, opisujemo korištene podatkovne skupove koji su korišteni tijekom procesa treniranja i validacije.

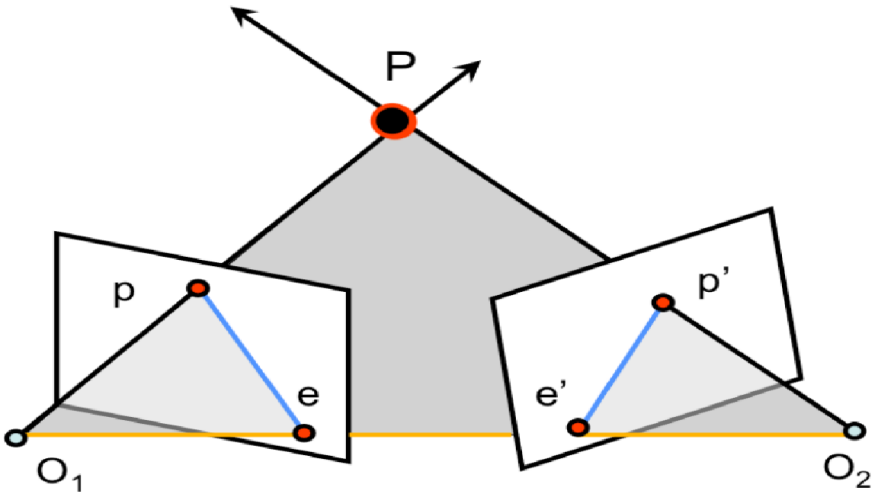
Na kraju, prikazujemo eksperimentalne rezultate i primjene razvijenog modela. Analiziramo dobivene rezultate i ocjenjujemo uspješnost modela u rješavanju problema korespondencije. Ova studija doprinosi razumijevanju i primjeni stereoskopske rekonstrukcije u računalnom vidu.

2. Steroskopska rekonstrukcija

U ovom poglavlju istražujemo tehniku stereoskopske rekonstrukcije, koja omogućuje rekonstrukciju trodimenzionalne scene temeljem analize dvodimenzionalnih projekcija iste scene. Stereoskopska rekonstrukcija predstavlja moćan alat za dobivanje dubinskih informacija i trodimenzionalnih struktura objekata, čime se otvaraju vrata brojnim primjenama u različitim područjima. Detaljnije ćemo analizirati epipolarnu geometriju, rektifikaciju i disparitet.

2.1. Epipolarna geometrija

U epipolarnoj geometriji promatramo točku P koja se nalazi u trodimenzionalnom prostoru i njenim projekcijama p i p' dobivenima iz dviju kamera gdje su O_1 i O_2 središta kamera, a e i e' epipolovi dobiveni pravcem koji siječe slikovne ravnine kamera. One zajedno čine epipolarnu ravninu koja nam je od velikog značaja. Slika 2.1 prikazuje epipolarnu ravninu koja se može opisati točkama P, O_1 i O_2

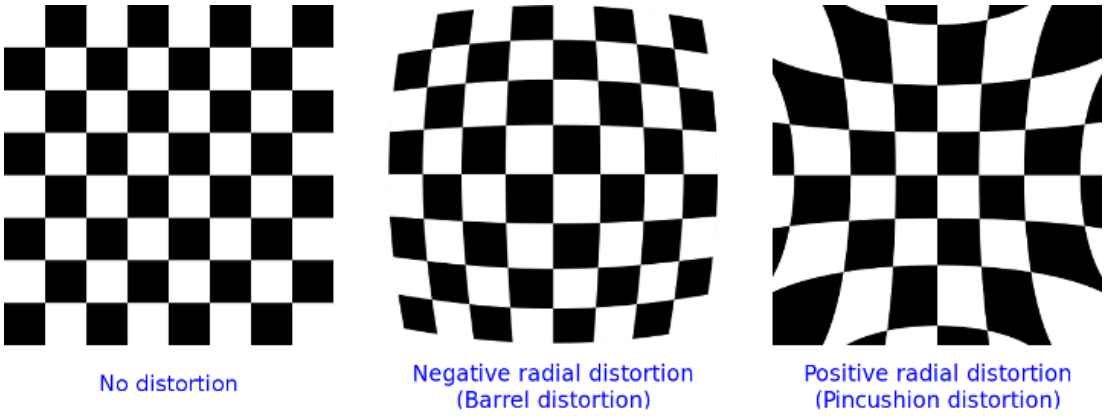


Slika 2.1: Epipolarna ravnina označena sivom bojom [1]

U epipolarnoj ravnini su sadržani i epipolarni pravci p i e, p' i e' u kojima epipolarna ravnina presijeca slikovne ravnine kamera. Glavna značajka epipolarnih pravaca je ta što ako znamo p, e i e' uz činjenicu da se nalaze na istoj epipolarnoj ravnini značajno si olakšavamo pronalazak točke p' jer se ona mora nalaziti na epipolarnom pravcu p' i e'.

2.2. Rektifikacija

Rektifikacija je proces kojim se projekcijske ravnine slika transformacijama svode na istu ravninu. Za izračun i provedbu transformacije potrebni su intrinzični i ekstrinzični parametri kamera. Ekstrinzični parametri opisuju odnos između para kamera, dok intrinzični opisuju svojstva svake kamere zasebno.
Uz pomoć ekstrinzike kamera mogu se projekcijske ravnine kamera dovesti u istu ravninu te da svi pikseli duž horizontalnog pravca prvog para odgovara istom pravcu drugog para (ista visina).



Slika 2.2: Primjer radijalne izobličenosti [2]

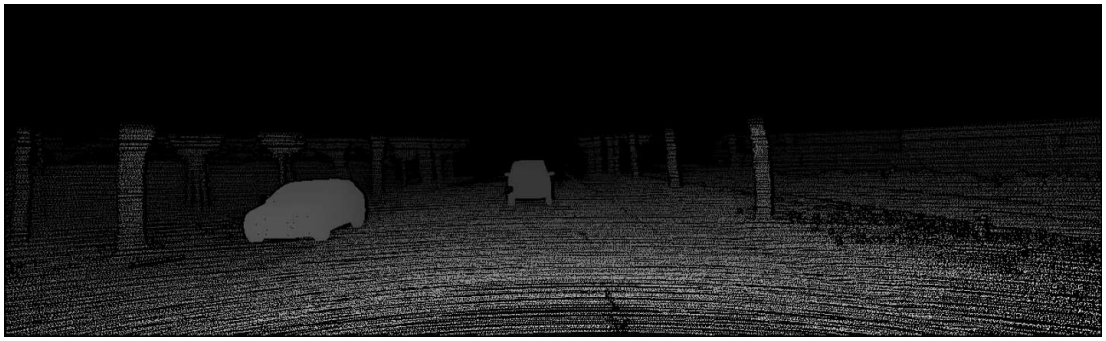
Također se moraju i otkloniti nedostatci intrinzičnih svojstava kamera zasebno kao što su nesavršenost leće, pomak senzora od centra leće i slične nesavršenosti leće. Primjer nesavršenosti leća u slici 2.2. Za određivanje tih parametara obično se koristi šahovska ploča, jer pruža vizualno jasne prednosti i olakšava proces kalibracije.

Nakon što se provede rektifikacija slika dobijemo jednake parametre intrinzičkih svojstava što nam uveliko olakšava računanje u epipolarnoj geometriji. Također u skupovima slika koje su korištene za ovaj rad postupak rektifikacije je već obavljen.

Slika 2.3: Prikaz epipolarnih linija na Kitti Stereo 2015. nakon rektifikacije

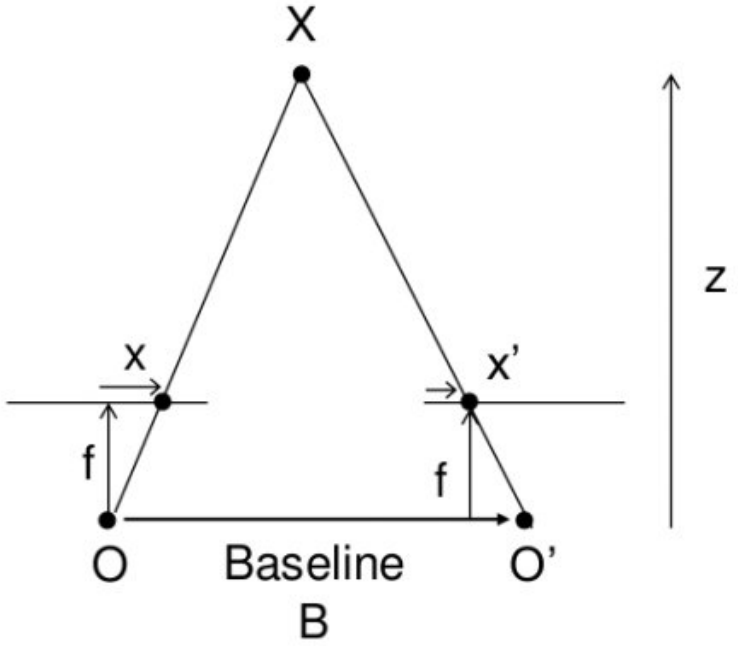
2.3. Disparitet

Disparitet je horizontalni pomak korespondencijskih točka u slikama koje su nastale iz iste scene. Disparitet možemo izračunati za svaki piksel u slici i tako dobiti mapu dispariteta (slika 2.4).



Slika 2.4: Mapa dispariteta iz Kitti Stereo 2015.

Računanje dispariteta nakon rektifikacije nam je značajno olakšano jer se parovi slika nalaze u istim ravninama, imaju jednaku žarišnu udaljenost te se korespondencijske točke nalaze na jednakim visinama. To onda implicira da na slici 2.5 možemo jednostavno primijeniti sličnost trokuta za izračun dispariteta.



Slika 2.5: Skica za izračun dispariteta [3]

Dubina scene je označena sa Z, B označava udaljenost kamera, a f označava žarišnu udaljenost.

Iz sličnosti trokuta lijeve slike imamo:

$$\frac{f}{x} = \frac{Z}{X} \tag{2.1}$$

Iz sličnosti trokuta desne slike imamo:

$$\frac{f}{x'} = \frac{Z}{X - B} \tag{2.2}$$

Izvlačimo x i x':

$$x = \frac{fX}{Z}, x' = \frac{f(X - B)}{Z} \tag{2.3}$$

Oduzimamo x i x' i dobijamo:

$$x - x' = disparity = \frac{fB}{Z} \tag{2.4}$$

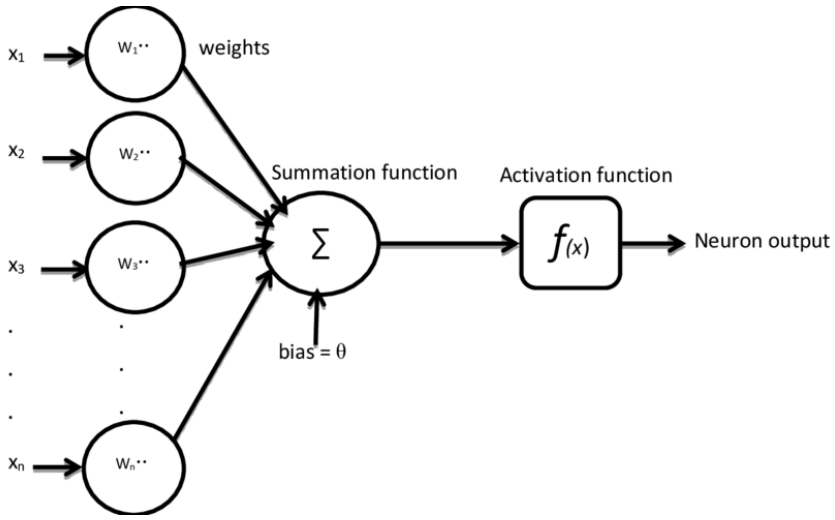
Vidljivo je da je vrijednost dispariteta obrnuto proporcionalna udaljenosti objekta od kamere. To znači da visoke vrijednosti dispariteta odgovaraju bliskim objektima, dok niže vrijednosti dispariteta odgovaraju udaljenijim objektima.

3. Duboko učenje

Duboko učenje [4] je područje umjetne inteligencije, bavi se rješavanjem problema koja nemaju definirana pravila koja bi dovela do rješenja problema već se samim algoritmom učenja nad skupom relevantnih podataka pokušava doći do tih pravila.

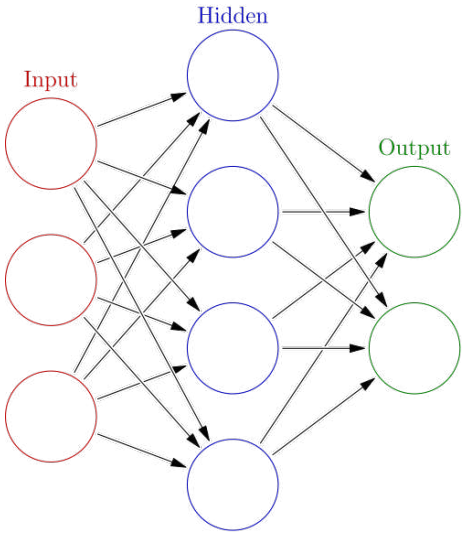
3.1. Umjetne neuronske mreže

Umjetne neuronske mreže su inspirirane biološkim neuronima i predstavljaju konektivistički pristup u području umjetne inteligencije. Osnovni element neuronske mreže je umjetni neuron koji se sastoji od težina, ulaznih i izlazne vrijednosti i aktivacijske funkcije.



Slika 3.1: Struktura umjetnog neurona [5]

Umjetni neuron informaciju temelji na numeričkim vrijednostima pa su tako i težine, ulazi i izlazi u numeričkoj formi. Ulazne vrijednosti neuron množi s pripadnim težinama te na to dodaje prag (eng. bias), onda obavlja operacija sumacije koju još naknadno propuštamo kroz aktivacijsku funkciju koja nam vraća izlaznu vrijednost umjetnog neurona.



Slika 3.2: Arhitektura primjera neuronske mreže, koju skraćeno izražavamo kao 3x4x2, sastoji se od 3 ulazna neurona, 4 skrivena neurona i 2 izlazna neurona.

Umjetne neuronske mreže su strukturirane tako da se sastoje od više umjetnih neurona, pri čemu je bitna karakteristika da u samoj mreži nema ciklusa. Tipično se neuroni agregiraju u slojeve, imamo ulazni sloj, skrivene slojeve i izlazni sloj. Najčešći tip mreže je potpuno povezana mreža u kojoj su neuroni iz jednog sloja potpuno povezani s neuronima prethodnog sloja.

3.1.1. Aktivacijske funkcije

Aktivacijske funkcije u umjetnim neuronskim mrežama su od velike važnosti jer omogućuju modeliranje nelinearnih veza između ulaznih i izlaznih podataka. Ova nelinearnost je važna jer mnogi stvarni problemi nisu linearni po svojoj prirodi. Stoga, aktivacijske funkcije koje se koriste u umjetnim neuronskim mrežama obično su nelinearne, a danas najkorištenije aktivacijske funkcije su sigmoidalna funkcija, tangens hiperbolni i funkcija zglobnice (eng. Rectified Linear Unit). Odabir aktivacijske funkcije ovisi o samom problemu koji rješavamo i prirodi podataka s kojima radimo.

– Sigmoidna funkcija

$$f(x) = \frac{1}{1 - e^{-x}} \tag{3.1}$$

– Tangens hiperbolni

$$f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{3.2}$$

– Funkcija zglobnice

$$f(x) = max(0, x) \tag{3.3}$$

3.2. Algoritam učenja

Najvažniji dio neuronskih mreža je proces učenja samih mreža. Učenje se može provoditi na tri glavna načina: nadzirano učenje, nenadzirano učenje i podržano učenje.

- **Nadzirano učenje** podatci uz značajke imaju i odgovarajuće oznake.
- **Nenadzirano učenje** podatci nisu označeni već se iz podataka pokušavaju naučiti neke pravilnosti.
- **Podržano učenje** model uobičajeno poduzima neke akcije i na temelju tih akcija model ili nagrađuje ili kažnjava.

Algoritam učenja pokušava optimizirati težine nad podatcima za učenje kako bi greška modela bila što manja. Grešku modela definiramo funkcijom gubitka. Za optimizaciju težina se uobičajeno koristi gradijentni spust.

3.2.1. Funkcija gubitka

Funkcija gubitka je mjera između predikcije model i očekivanih vrijednosti. Za optimalnu predikciju bitno je minimizirati funkciju gubitka. Odabir funkcije gubitka o visi o problemu koji rješavamo, najčešće su to srednja kvadratna pogreška (eng. mean squared error) i unakrsna entropija.

- **Srednja kvadratna pogreška**
  korisna u problemima regresije, a definiramo je kao:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i' - y_i)^2, \tag{3.4}$$

  gdje n označava broj podataka, y_i stvarnu oznaku i y_i' predikciju.

- **Unakrsna entropija**
  korisna u problemima klasifikacije, a definiramo je kao:

$$H(p, q) = -\sum_{x} p(x) \log q(x), \tag{3.5}$$

  gdje p i q označavaju vjerojatnosti.

3.2.2. Unatražna propagacija

Unatražna propagacija (eng. backpropagation) je postupak kojim se na izračunu parcijalnih derivacija optimiziraju parametri neuronske mreže. Da bi izračun parcijalnih derivacija bio učinkovit tijekom unaprijednog prolaza mrežom spremaju se međuvrijednosti pa se tijekom unazadnog prolaza efektivno od izlaza mreže do ulaza mreže mogu izračunati gradijenti. Gradijenti se računaju uz pomoć lančanog pravila i već spremljenih međuvrijednosti.

3.2.3. Gradijentni spust

Gradijentni spust je iterativni pomak u smjeru gdje funkcija najbrže opada, to nam je od koristi jer nam je cilj učenja minimizirati funkciju gubitka.
Negativni gradijent pokazuje smjer u kojem funkcija najbrže opada. Iznos pomaka se korigira hiperparametrom ϵ poznat kao stopa učenja (eng. learning rate) koji ne smije biti ni prevelik ni premali. Bilo bi idealno kad bi ϵ bio što manji, ali onda bi učenje trajalo predugo, dok za prevelik ϵ učenje divergira. U praksi bi se ovakvim jednostavnim iterativnim pomakom često zapinjalo u lokalnom minimumu funkcije. Iz tih razloga se u praksi koriste već gotove metode kao što su stohastički gradijentni spust (SGD, eng. Stochastic Gradient Descent) i Adam [6] (eng. Adaptive Moment Estimation)[7].

- **Stohastički gradijentni spust** je optimizacijski algoritam koji računa gradijente na nasumičnom odabranom podskupu za treniranje (mini-grupa). Nasumični odabir uvodi šum koji može pomoći procesu optimizacije da izbjegne lokalne minimume.
- **Adam** je optimizacijski algoritam koji koristi adaptivnu stopu učenja izračunatu procjenama prvog i drugog momenta gradijenata.
  Formulom $m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta \mathcal{L}$ opisujemo prvi moment,
  Formulom $v_t = \beta_2 v_{t-1} + (1 - \beta_2)\nabla_\theta \mathcal{L}^2$ opisujemo drugi moment,
  težine konačno ažuriramo formulom $\theta_t \leftarrow \theta_{t-1} - \frac{lr}{\sqrt{v}+\epsilon} m$.
  Obično se koristiti $\beta_1 = 0.9$, $\beta_2 = 0.999$ i $\epsilon = 10^{-8}$ [8].

3.2.4. Regularizacija

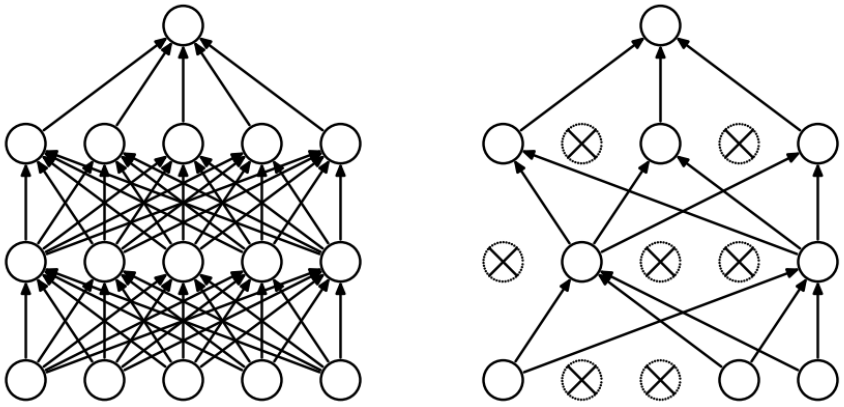
Regularizacija je tehnika koja se koristi s ciljem poboljšanja robusnosti modela na nepoznatim skupovima podataka i unapređenja generalizacije, pri čemu se ne posvećuje primarno smanjenju greške tijekom učenja. Kao regularizaciju možemo koristiti

umjetno povećanje broja podataka na primjer kod slika možemo ih zrcaliti, rotirati, izrezivati i slično.
Koristi se još i kažnjavanje norme vektora težina na primjer L1 i L2.
L1 i L2 norma se primjenjuje na težine modela, a definirane su formulama:

$$L1 = ||w||_1 = \sum_{i=0}^{n} |w_i| \tag{3.6}$$

$$L2 = ||w||_2^2 = \sum_{i=0}^{n} |w_i|^2 \tag{3.7}$$

Da bi model uzeo u obzir regularizaciju u postupcima optimizacije bitno ju je zbrojiti s funkcijom gubitka ($\mathcal{L}_{reg} = \mathcal{L} + \lambda L$).



(a) Standard Neural Net (b) After applying dropout.

Slika 3.3: Isključivanje neurona [9]

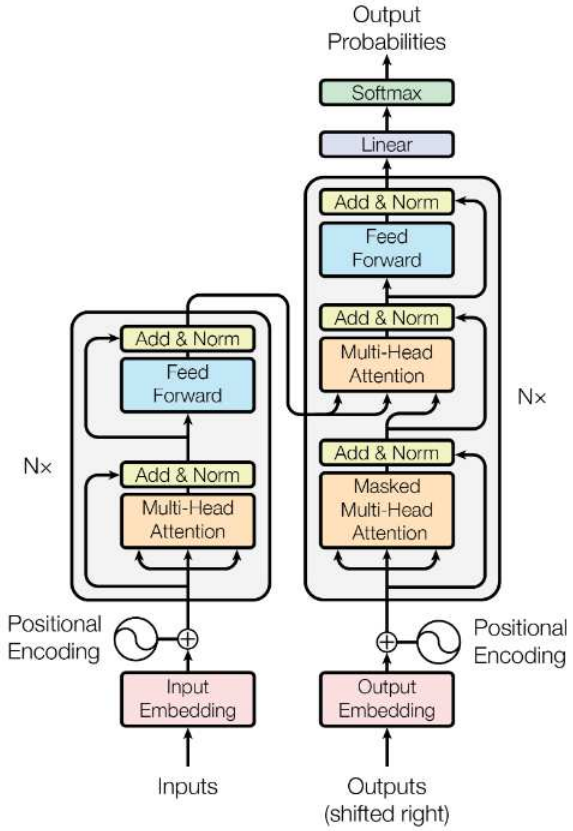
Danas se često koristi pristup regularizacije isključivanja neurona (eng. dropout), koji se temelji na isključivanju neurona s određenom vjerojatnošću tijekom faze učenja.

Još jedan primjer regularizacije je normalizacija koja se koristi na izlazima iz neurona prije same primjene aktivacijske funkcije. U implementacijama uz klasičnu normalizaciju obično se konačni rezultati normalizacije množe sa γ i zbrajaju sa β gdje su γ i β parametri modela koji se uče.

4. Korišteni neuronski modeli

4.1. Transformer

Transformer je vrsta arhitekture dubokih neuronskih modela koja je postala vrlo popularna u području obrade prirodnog jezika (NLP) i računalnog vida, koja je izuzetno uspješno predstavljena u radu "Attention Is All You Need" [10] 2017. godine. U daljnjem navođenju transformera mislit ćemo na upravo arhitekturu opisanu u navedenom radu. Glavna komponenta ove arhitekture je zasigurno sloj pažnje, a taj sloj se koristi u dekoderu i koderu modela.



Slika 4.1: Arhitektura transformera [10]

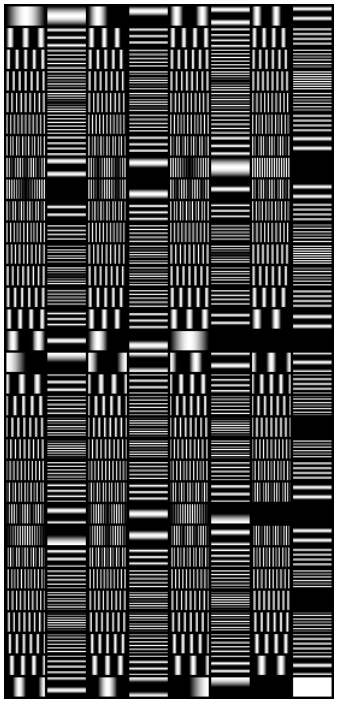
4.1.1. Ulaz

Ulaz u transformer se bitno razlikuje od ulaza u konvolucijskim modelima.
Kod konvolucijskih modela očekuju se tenzori dimenzija CxHxW, gdje je C broj kanala, H visina slike i W širina slike.
Kod transformera ulaz u model možemo najlakše objasniti na primjeru obrade prirodnog jezika, gdje je ulaz slijed riječi (tokena) i gdje se svaki token onda ugrađuje u vektorski prostor te onda dobijamo ulaz dimenzija NxE, gdje je N broj tokena, a E veličina vektora. Još jedan od problema transformera je taj što on nema informaciju o poziciji samih tokena pa se zbog toga tokenu nakon ugradnje u vektorski prostor dodaje još i vektor pozicijskog kodiranja. Pozicijsko kodiranje se može obaviti na više načina, kao što je naučeno pozicioniranje ili korištenje mrežne rešetke (eng. mesh grid) za tu svrhu.
Pozicijsko kodiranje koje mi koristimo u radu je linearno pozicijsko kodiranje, gdje svaku točku $\mathbf{x} = [x, y]$ kodiramo sljedećim formulama:

$$\mathcal{P}(\mathbf{x}) = [p_1(\mathbf{x}), p_2(\mathbf{x}, ..., p_{N/4}(\mathbf{x}))], \tag{4.1}$$

$$p_k(\mathbf{x}) = [sin(k\pi\mathbf{x}^T), cos(k\pi\mathbf{x}^T)] \tag{4.2}$$

Primijetite da p_k na izlazu daje 4 vrijednosti, zato u $\mathcal{P}$ idemo do $p_{N/4}$. U našem modelu N je 256 koji označava dimenziju ugrađenog vektora.
Slikama dodavamo pozicijsko kodiranje koje kodiramo funkcijom $\mathcal{P}$ koordinatne funkcije MeshGrid(0:1, 0:2) veličine 16 x 32 x 2.



Slika 4.2: Primjer pozicijskog kodiranja za slike veličine 256 x 32 x 16

4.1.2. Koder

Koder je jedinica modela koja određenu informaciju želi prikazati u vektorskom obliku. Može se koristiti i zasebno bez dekodera za probleme klasifikacije slično kao i koderi u konvolucijskim mrežama. Naravno koder u transformeru je drugačije strukture. Sastoji se od sloja pažnje s više glava kojem se onda dodaje rezidual te sloj normalizacije. Nakon toga ga prosljeđujemo višeslojnom perceptronu koji opet dodaje rezidual i sloj normalizacije. Bitno je za naglasiti da je ulaz i izlaz kodera jednakih dimenzija upravo zbog korištenja rezidualnog sloja.

4.1.3. Dekoder

Dekoder je jedinica koja želi iz nekog vektorskog oblika podataka izvući korisne informacije. Struktura dekodera u transformeru sastoji se od više slojeva. Prvi sloj je maskirani sloj pažnje s više glava, koji je povezan s rezidualnim slojem i normalizacijskim slojem. Nakon toga, dolazi sloj pažnje s više glava, koji također uključuje rezidualni sloj i sloj normalizacije. Konačno, izlaz iz tog sloja se prosljeđuje višeslojnom perceptronu, koji također koristi rezidualni sloj i sloj normalizacije. Ulaz i izlaz iz dekodera je i ovdje istih dimenzija zbog korištenja rezidualnog sloja.

4.1.4. Pažnja

Pažnja je najvažniji dio transformera. Pažnju se može promatrati kao mapiranje upita (eng. query) i parova ključ (eng. key), vrijednost(eng. value) na izlaz, gdje su svi navedeni vektori. Izlaz se dobiva kao težinska suma vrijednosti, gdje je težina računata s funkcijom pažnje između upita i korespondencijskog ključa. Funkcija pažnje koju koristimo je *Scaled dot-product attention*, koja se također koristi u radu "Attention Is All You Need" [11], a ista je implementirana i u PyTorch-u [12].

Scaled dot-product attention:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.3}$$

U formuli 4.3 Tenzori Q, K i V redom predstavljaju upite, ključeve i vrijednosti. Dimenzija ugrađenog vektora označena je sa d_k. Tenzori imaju dimenzije NxD, gdje je N veličina slijeda, a D je dimenzija ugrađenog vektora. Upotreba tenzora umjesto vektora ima za svrhu efikasno rukovanje grupama.

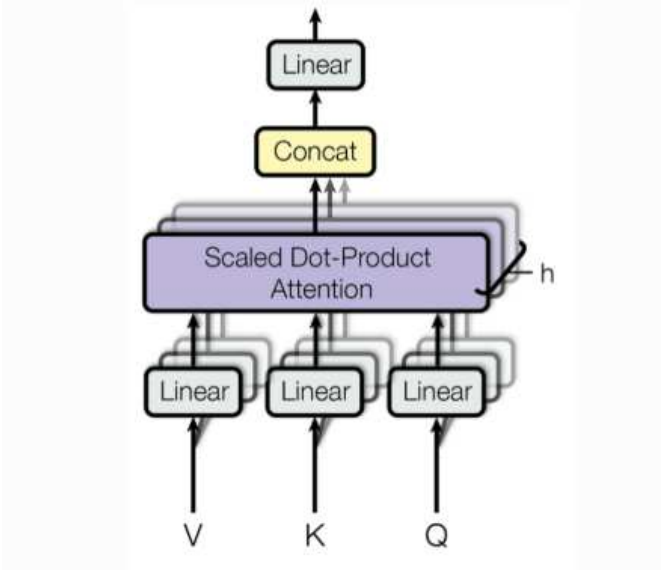
4.1.5. Pažnja s više glava

Umjesto jednostavne pažnje s ključevima, upitima i vrijednostima, pokazalo se da je korisno projicirati ključeve, upite i vrijednosti h puta, gdje bi h bio broj glava. Prvo se ključevi i upiti projiciraju u dimenzije d_k i d_k s h naučenih linearnih projekcija, gdje je d_k dimenzija vektora ključa, dok se vrijednosti projiciraju u dimenzije d_v opet s h naučenih linearnih projekcija, gdje je d_v dimenzija vektora vrijednosti, onda se na svakoj od tih h glava provodi pažnja opisana u formuli (4.3). Na kraju se izlazi tih h glava spoje i projiciraju s tenzorom W^O nazad u početni prostor dimenzija ključeva, upita i vrijednosti.

$$Multihead(Q, K, V) = Concat(head_1, ..., head_h)W^O, \tag{4.4}$$

gdje je $head_i = Attention(QW_i^Q, KW_i^K, V_i^V)$.

Dimenzije W_i^Q, W_i^K, W_i^V i W^O bi bile
redom d x d_k, d x d_k, d x d_v i hd_v x d, gdje je d originalna dimenzija.



Slika 4.3: Pažnja s više glava

4.2. Konvolucijske neuronske mreže

Za modeliranje dubokih neuronskih mreža, konvolucijski pristup je postao standardna i neizostavna tehnika u području računalnog vida. Bitno je naglasiti da se u konvolucijskim mrežama osim ključnih konvolucija u samim modelima mogu koristiti i slojevi sažimanja ili potpuno povezani slojevi. Konvolucijske mreže su se pokazale posebno korisnima u problemima računalnog vida jer su za razliku od potpuno povezanih modela robusnije na prostornu invarijantnost, prepoznavaju razne slikovne uzorke te su parametarski puno efikasnije nego potpuno povezani modeli.

4.2.1. Konvolucija

Konvolucija se opisuje formulom

$$y[i,j] = \sum_{m=0}^{m_{max}} \sum_{n=0}^{n_{max}} h[m,n] \cdot x[i-m, j-n] \tag{4.5}$$

, gdje x označava originalnu sliku, a h jezgru same konvolucije. Sama jezgra konvolucije je tipično 3x3, 5x5 i 7x7 dimenzija, gdje se još u to treba nadodati i veličina kanala (C) i tako onda dobijemo tenzor kojem mijenjamo parametre tijekom učenja. Konvolucije su tipično korisne za izvlačenje linija, naglih prelaza i vrhova u slikama, a kod dubokih naučenih modela mogu izvući i bitnije značajke. Bitno je naglasiti da se jezgra u uobičajenim algoritmima alata za duboko učenje ne zaokreće i tako ne obavlja klasičnu operaciju konvolucije već zapravo operaciju unakrsne korelacije.

$$\begin{bmatrix} 5 & 12 & 13 & 10 \\ 15 & 4 & 12 & 9 \\ 3 & 6 & 9 & 8 \\ 13 & 10 & 4 & 6 \end{bmatrix} \circledast \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 61 & 83 \\ 50 & 66 \end{bmatrix}$$

Slika 4.4: Primjer konvolucije

U uobičajenim alatima imamo još i parametre za pomak (eng. stride) i nadopunjavanje (eng. padding). Pomak označava samo korak jezgre nad ulaznim tenzorom. Dok nadopunjavanje proširuje ulazni tenzor nulama ili nekim drugim brojem, kako bi se ili korigirala izlazna veličina ili smanjio gubitak informacije na vrhovima slike.

Pomak i nadopunjavanje imaju utjecaj na izlazne dimenzije ulazne slike opisane sljedećim formulama za visinu (H) i širinu (W).

$$H_{out} = \lfloor \frac{H_{in} + 2 \cdot padding - kernelsize}{stride} + 1 \rfloor \tag{4.6}$$

$$W_{out} = \lfloor \frac{W_{in} + 2 \cdot padding - kernelsize}{stride} + 1 \rfloor \tag{4.7}$$

4.2.2. Sažimanje

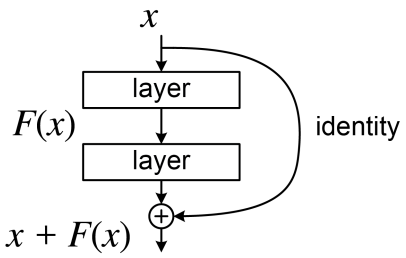
Slojevi sažimanja su slični slojevima konvolucije s tim da oni ne koriste parametre već primjenjuju neku operaciju nad slikom koristeći jezgru. Neke od operacija su sažimanje srednjom vrijednošću, maksimalnom vrijednošću ili nekom od normi na primjer L2. Sažimanje se koristi za grupiranje prostorno bliskih značajki i smanjivanje slikovne rezolucije. U praksi se uglavnom odabiru veličine jezgre 2x2 i koraka 2 koje efektivno smanjuju slikovnu rezoluciju za faktor 2. Također se često u zadnjim slojevima koriste jezgre koje su iste veličine kao i ulazna rezolucija (globalno sažimanje) koje onda efektivno daju na izlazu jednu vrijednost što je korisno za prijelaz na potpuno povezane mreže.

$$\begin{bmatrix} 5 & 12 & 13 & 10 \\ 15 & 4 & 12 & 9 \\ 3 & 6 & 9 & 8 \\ 13 & 10 & 4 & 6 \end{bmatrix} \longrightarrow \begin{bmatrix} 15 & 13 \\ 13 & 9 \end{bmatrix}$$

Slika 4.5: Maksimalna vrijednost kao primjer sažimanja.

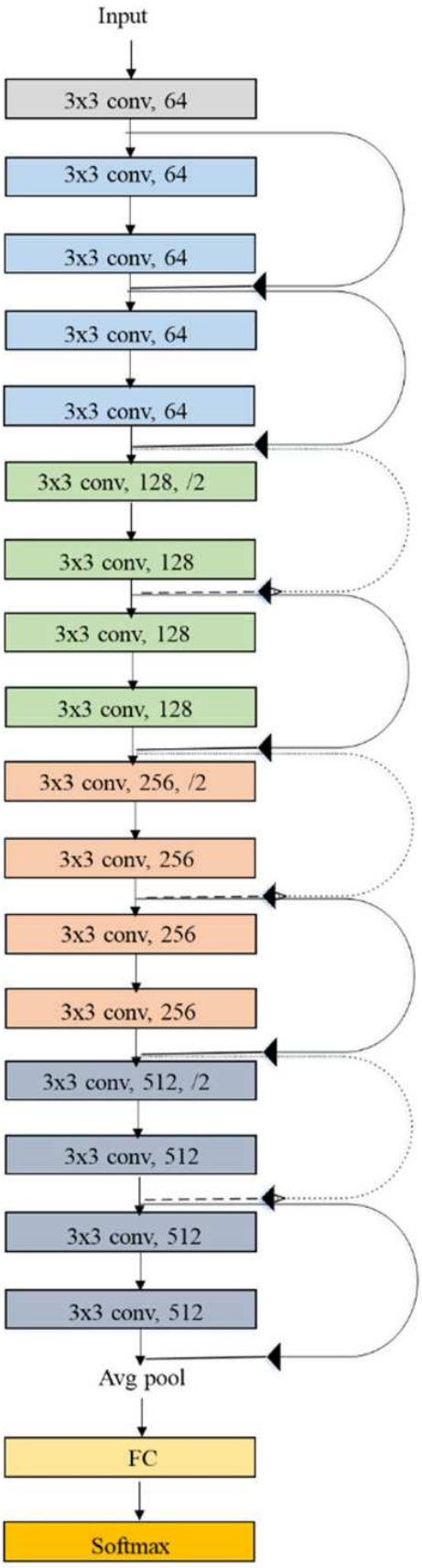
4.2.3. ResNet-18

Duboka konvolucijska mreža ResNet-18 [13] koristi rezidualne veze. U ovoj sekciji detaljnije ćemo opisati njenu arhitekturu jer je koristimo kao koder slika u implementaciji povezanoj s problemom koji se istražuje u ovom radu.
Glavne prednosti modela s rezidualnim vezama su te što mogu lako naučiti funkcije identiteta i lakše razrješavaju problem nestajućeg gradijenta, što je od velike koristi u jako dubokim modelima koji imaju sličnih problema.
Taj problem modeli s rezidualnim vezama rješavaju rezidualnim blokovima.
Rezidualni blok se sastoji od 2 konvolucijska sloja, gdje se na izlazu drugog sloja pribraja ulaz prvog sloja i tako održava referencu na sami identitet ulaza..



Slika 4.6: Rezidualni blok [13]

Arhitektura ResNet-18 se temelji na konceptu rezidualnih blokova i koristi model sa 18 slojeva. U nastavku je priložena slika 4.7 u kojoj su rezidualne veze označene strelicama.



Slika 4.7: ResNet-18 arhitektura [14]

5. Metoda i programska izvedba

5.1. Formulacija problema

Neka je $x \in [0,1]^2$ normalizirana koordinata upita u slici I, kojoj želimo pronaći korespondencijsku točku $x' \in [0,1]^2$ u slici I'. Gledat ćemo na rješenje problema kao pronalazak najboljih parametara Φ za parametarsku funkciju $\mathcal{F}_\Phi(x|I,I')$ gdje minimiziramo:

$$\underset{\Phi}{\operatorname{argmin}} \underset{(x,x',I,I')\sim\mathcal{D}}{\mathrm{E}} \mathcal{L}_{corr} + \mathcal{L}_{cycle}, \tag{5.1}$$

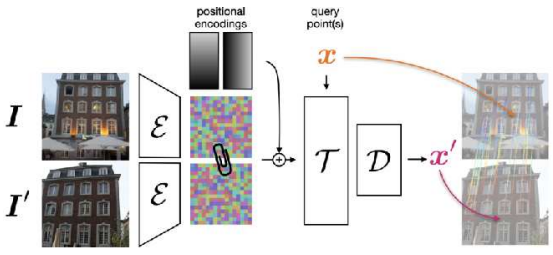
$$\mathcal{L}_{corr} = \|x' - \mathcal{F}_\Phi(x|I,I')\|_2^2, \tag{5.2}$$

$$\mathcal{L}_{cycle} = \|x - \mathcal{F}_\Phi(\mathcal{F}_\Phi(x|I,I')|I,I')\|_2^2 \tag{5.3}$$

U prikazanoj jednadžbi, $\mathcal{D}$ označava skup podataka za treniranje, $\mathcal{L}_{corr}$ je greška korespondencije i $\mathcal{L}_{cycle}$ je ciklička konzistentnost korespondencije.

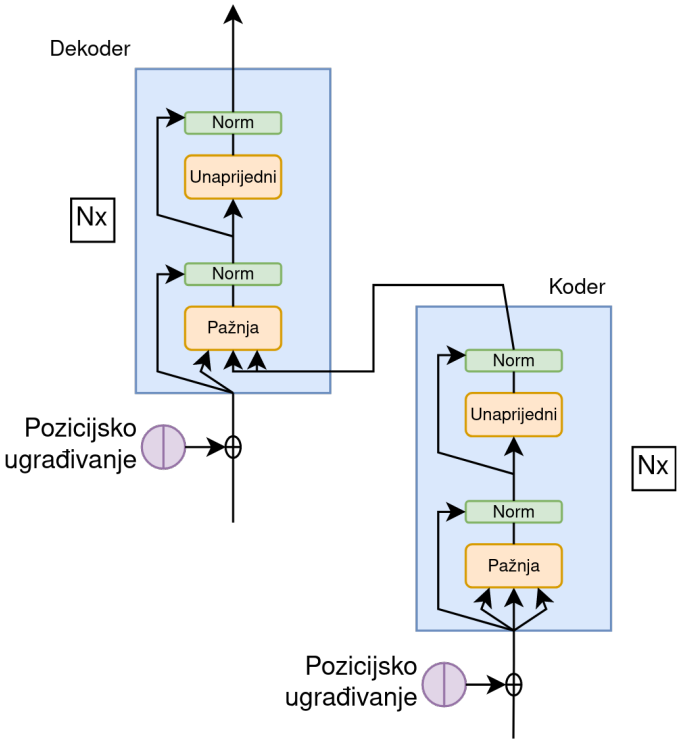
5.2. Implementacija

Za implementaciju navedene funkcije $\mathcal{F}_\Phi$ i rješavanje problema korespondencije koristimo metodu COTR [11] koja ostvaruje korespondenciju kombiniranjem klasičnih konvolucijskih ugrađivanja i slojeva pažnje. Slika 5.1 prikazuje arhitekturu modela.



Slika 5.1: $\mathcal{E}$ označava konvolucijsko ugrađivanje, $\mathcal{T}$ je transformer, dok je $\mathcal{D}$ oznaka za višeslojnu umjetnu neuronsku mrežu [11]

Prvo uzmemo par slika kojem promijenimo prostorne veličine na dimenzije 256 x 256, te ih onda kodiramo u mape značajki uz pomoć već naučene rezidualne konvolucijske mreže. Za rezidualnu konvolucijsku mrežu koristimo ResNet-18 [13] koja je već naučena na više milijuna slika iz skupa podataka ImageNet [15]. Ne koristimo potpuni prolaz kroz rezidualni konvolucijski model već izvlačimo samo mape značajki iz trećeg konvolucijskog bloka. Taj sloj daje izlazne dimenzije 16x16x256. Nakon toga, dodajemo jedan konvolucijski sloj s jezgrom veličine 1 i istim brojem značajki. Konačno, za svaku sliku dobivamo dimenzije 16x16x256. Tijekom učenja ne učimo ResNet-18, već samo taj zadnji konvolucijski sloj. Ovaj dio odgovara elementu $\mathcal{E}$ u slici 5.1. Nakon toga spajamo mape značajki jednu uz drugu (16x32x256) i dodajemo im pozicijsko kodiranje. Uz dobiveni par slika koje ispeglamo u vektore imamo još i točke za koje želimo dobiti korespondenciju koje kodiramo istim pozicijskim kodiranjem kao i mape značajki te ih onda zajedno sa spojenim mapama značajki proslijedimo transformeru. Pozicijsko ugrađivanje je potrebno zbog implementacije pažnje uz pomoć skaliranog matričnog množenja 4.3 (eng. scaled dot-product). Zbog kojeg transformer nema informaciju o poziciji ulaza, već samo težinsku informaciju između upita i ključa. Da bismo riješili taj problem koristimo linearno pozicijsko ugrađivanje 4.2 kojim odgovarajući par x, y koordinata ugrađivamo u vektor značajki dimenzija 256 po formulama navedenim u 4.1 i 4.2. Pozicijsko ugrađivanje primijenimo na koordinate prostornih dimenzija spojenih mapi značajki (16 x 32 x 2), koje nam za svaku dvodimenzijsku koordinatu vrati vektor ugrađivanja veličine 256 (16 x 32 x 256). Sličnu stvar radimo i s točkama kojima tražimo korespondenciju, gdje su one već izražene dvodimenzionalnim vektorima koje odgovaraju njihovim pozicijama u slikama pa uz ugrađivanje dobijemo vektore dimenzija 256. Broj korespondencijskih točaka je proizvoljan, ali je u ovom radu tijekom učenja iz svih točaka koje su dobro definirane disparitetom nasumično birano 100 točaka.



Slika 5.2: Arhitektura transformera, sastoji se od kodera i dekodera koji imaju N slojeva (Nx).

Transformer je građen od 6 slojeva kodera i dekodera. Svaki od slojeva kodera sadrži sloj pažnje s 8 glava, a svaki sloj dekodera isto 8 glava, ali ne sadrži sloj pažnje već sloj unakrsne pažnje da bi se spriječila komunikacija između korespondencijskih točaka (želimo da model razmatra svaku točku nezavisno). Dekoder transformera je dio modela koji je zapravo zaslužan za pronalazak korespondencijskih točaka, no te točke su u visoko dimenzijskim koordinatama (N x 256, N označava broj točaka, 256 je dimenzija ugrađivanja). Konačno da bi koordinate sveli na dvodimenzijske koordinate koristimo dekoder koji je implementiran troslojnom umjetnom neuronskom mrežom s 256 značajki u svakom sloju.

5.3. Korišteni alati

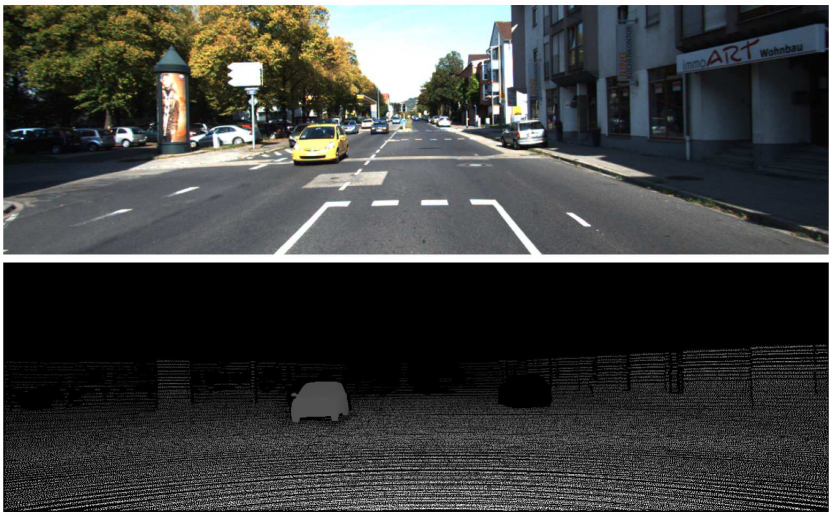
Programska implementacija je obavljena u programskom jeziku Python i u Jupyter bilježnicama. Za implementaciju ovog modela korišten je PyTorch okvir[12] koji ima potporu za grafičke procesorske jedinice (CUDA[16]). Samo treniranje i validacija modela se izvršavala na Google Colab-u[17] uz dodatak Google Drive-a na koji smo postavili sami izvorni kod implementacije u programskom jeziku Python te onda učitavali i trenirali uz pomoć Jupyter bilježnice na Google Colab-u. Google Colab nudi

grafičku karticu T4 koja ima ovisno o dostupnim resursima oko 15GB radne memorije. Podatkovni skup Kitti Stereo 2015 [18] je u vlasništvu Karlsruhe Institute of Technology and Toyota Technological Institute koji je javno dostupan. Za pripremu podataka koristili smo biblioteku Numpy. Numpy je izuzetno brz zahvaljujući implementacijama pisanim u C-u i strojnom kodu. Također koristimo i OpenCV[19] za obradu i pripremu slika.

6. Podatkovni skupovi

6.1. Skup podataka KITTI 2015

U ovom radu korišten je podatkovni skup KITTI 2015 [18] za učenje.
Sastavljen je od 200 parova slika kojima su izračunate mape dispariteta. Slike imaju razlučivosti od 374 x 1238 i 375 x 1242 piksela, dok se maksimalni disparitet kreće u rasponu od 62 do 230 piksela, s prosječnom vrijednosti od 83 piksela. Slike u podatkovnom skupu su već unaprijed kalibrirane i rektificirane. Točnost dispariteta na razini piksela se prihvaća uz maksimalno odstupanje od 3px. Podatkovni skup smo podijelili na 80% za učenje i 20% validaciju. Naknadno da bismo mogli par slika uz odgovarajuće korespondencije proslijediti modelu bitno je da su nam slike oblika 256 x 256 x 3, te da izračunamo korespondencije iz zadane mape dispariteta.



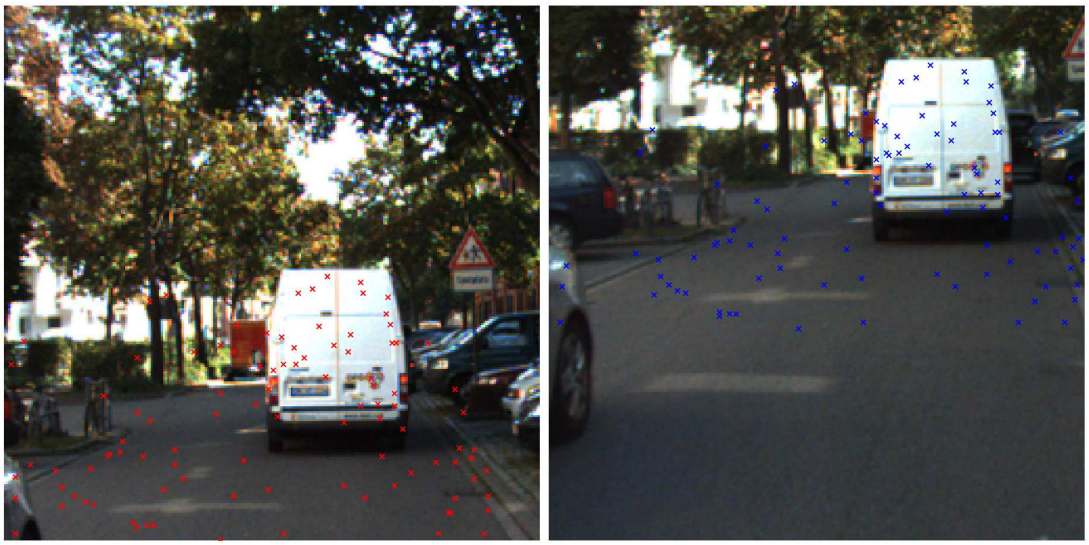
Slika 6.1: prikaz lijevog okna s odgovarajućom mapom dispariteta.

6.1.1. Predprocesiranje

Da bismo osigurali ispravne dimenzije slika i odgovarajuće korespondencijske točke obavljamo sljedeće transformacije. Koristimo 2 pristupa:

1. pristup (Ciljano izrezivanje)
    - Prvo uzmemo sve valjane točke iz mape dispariteta, točku smatramo valjanom ako joj je iznos dispariteta veći od 0.
    - Nakon toga pogledamo gdje se nalazi najsjevernija i najjužnija valjana točka te potom izrežemo par slika na istim razinama (efektivno iz slika izbacimo nebo).
    - Skaliramo valjane točke iz lijeve slike na interval [0.0, 0.5], dok njihove korespondencije na desnoj slici stavljamo na interval [0.5, 1.0].
    - Skaliramo slike na dimenzije 256 x 256.
    - Proberemo 100 nasumično izabranih točaka.
2. pristup (Nasumično izrezivanje)
    - Prvo uzmemo sve valjane točke iz mape dispariteta, točku smatramo valjanom ako joj je iznos dispariteta veći od 0.
    - Potom izrežemo nasumično izabran prozor veličine 256 x 256 iz lijeve slike.
    - Izračunamo prozor koji sadrži sve točke na desnoj slici, te nasumično iz tog područja izaberemo prozor dimenzija 256 x 256.
    - Skaliramo valjane točke iz lijeve slike na interval [0.0, 0.5], dok njihove korespondencije na desnoj slici stavljamo na interval [0.5, 1.0].
    - Proberemo 100 nasumično izabranih točaka.

Slika 6.2: Par slika KITTI 2015 s korespondencijskim točkama koji dobijemo primjenom 1.pristupa



Slika 6.3: Par slika KITTI 2015 s korespondencijskim točkama koji dobijemo primjenom 2.pristupa

6.2. Skup podataka Middlebury 2014

Podatkovni skup Middlebury 2014 [20] se sastoji od više parova slika i odgovarajućih mapa dispariteta. Slike imaju razne razlučivosti između 1908 x 2800 i 2000 x 2952 piksela, dok im se maksimalni disparitet kreće u rasponu od 200 do 634 piksela, s

prosječnom vrijednosti 316 piksela. Te nad ovim skupom ne provodimo učenje već ga koristimo za evaluaciju već naučenog modela. Da bi parove slike iz ovog skupa mogli koristiti za predikcije moramo ili skalirati slike na dimenzije 256 x 256 ili prolaziti kroz sliku isječak po isječak, gdje je isječak dimenzija 256 x 256.

Slika 6.4: lijevo okno s odgovarajućom mapom dispariteta

7. Eksperimentalni rezultati

7.1. Metrike

U ovoj sekciji ćemo navesti metrike koje koristimo za validaciju i tijekom učenja. Za učenje modela koristimo srednju kvadratnu pogrešku (eng. mean squared error), dok za validaciju koristimo PCK (eng. percent of correct keypoints) i AEPE (eng. average endpoint error).

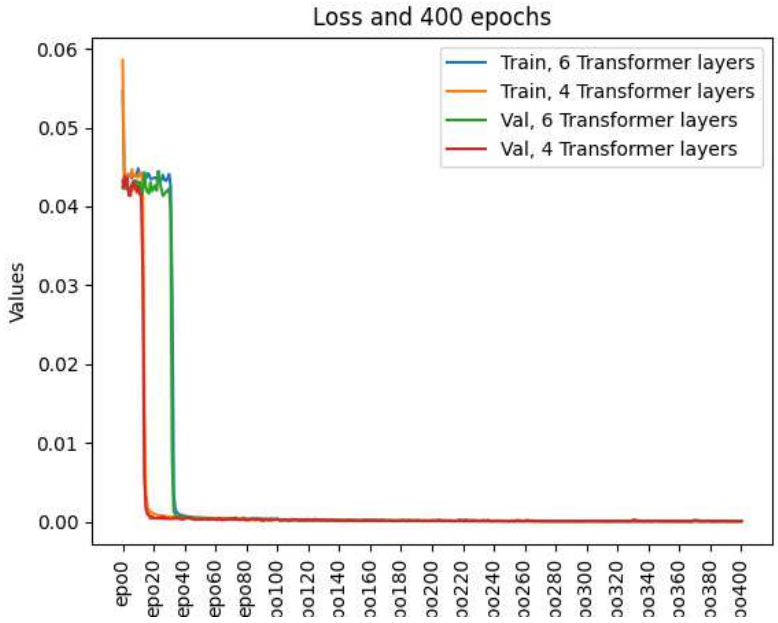
- **PCK** omjer točnih korespondencija kroz ukupan broj korespondencija, točku smatramo točnom ako se nalazi na udaljenosti manjoj od nekog praga. U ovom radu koristimo pragove od 1px, 3px i 5px.
- **AEPE** prosječna udaljenost između korespondencijskih točaka i pravih vrijednosti točaka. Izražavamo je u pikselima.

7.2. Rezultati

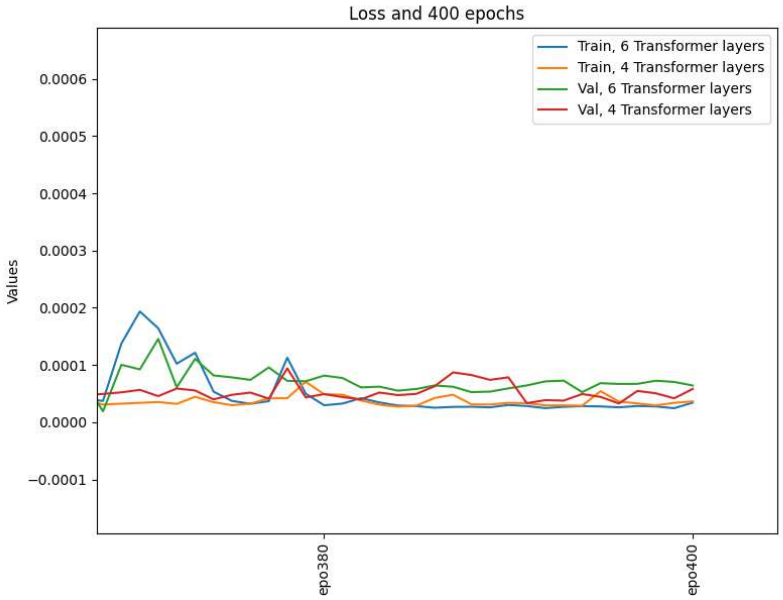
Eksperimentirali smo s dva modela koji se razlikuju u broju slojeva kodera i dekodera u transformeru, točnije jedan model sa 6 slojeva i jedan s 4 sloja i u koderu i dekoderu transformera. Inicijalno smo trenirali model nad KITTI 2015 kojem smo parove slika i korespondencija transformirali 1. pristupom već navedenim u prijašnjem poglavlju. Funkcija gubitka je već navedena srednja kvadratna greška. Model sa 6 slojeva ima 27,668,778 parametara, dok model s 4 sloja ima 22,408,490 parametara. Osnovica modela ResNet-18 koju koristimo kao koder $\mathcal{E}$ iz slike 5.1 ima 11,689,512 parametara kojih ne učimo. Učenje se provodi kroz 400 epoha i veličinom grupe 4, ukupno učenje traje oko 4 sata i obavlja se na Google Colab platformi.
Kao optimizator koristimo Adam sa stopom učenja 10^{-4} i betama
($\beta_1 = 0.9$, β_2 = 0.999).

U 7.1 možemo vidjeti kako se funkcija gubitka naglo spusti nakon 20 epoha za

model s 4 sloja u koderu i dekoderu i nakon 40 epoha u modelu sa 6 slojeva u koderu i dekoderu, nakon toga je funkcija gubitka stabilna s vrlo malim pomacima, dok se sve vrijeme gubitak na skupu za validaciju bitno ne odmiče od gubitka na podatcima za treniranje, slika 7.2 prikazuje gubitak u zadnjih 20 epoha.

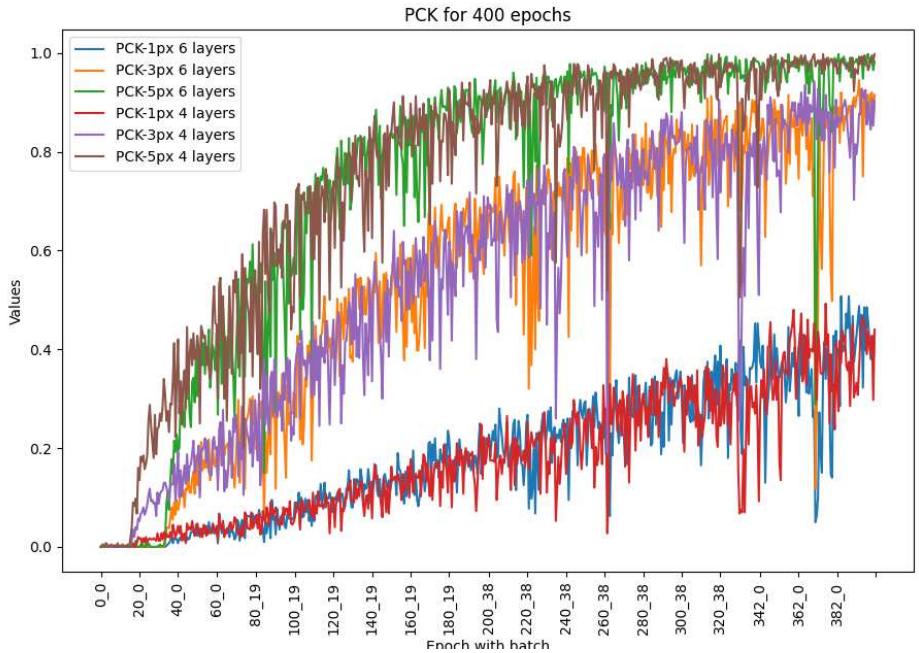


Slika 7.1: Funkcija gubitka (mean squared error) kroz 400 epoha



Slika 7.2: Funkcija gubitka zumirana na zadnjih 20 epoha

Tijekom procesa učenja također pratimo PCK metrike za udaljenosti od 1 piksela, 3 piksela i 5 piksela. Ove metrike se mjere na nasumično odabranim točkama iz skupa valjanih točaka, što rezultira prisutnošću značajne količine šuma na grafovima.



Slika 7.3: PCK metrika (eng. percentage of correct keypoints)

Nakon treniranja prikazujemo točnosti modela za metrike PCK-1px, PCK-3px, PCK-5px, AEPE i srednju kvadratnu grešku 7.2.

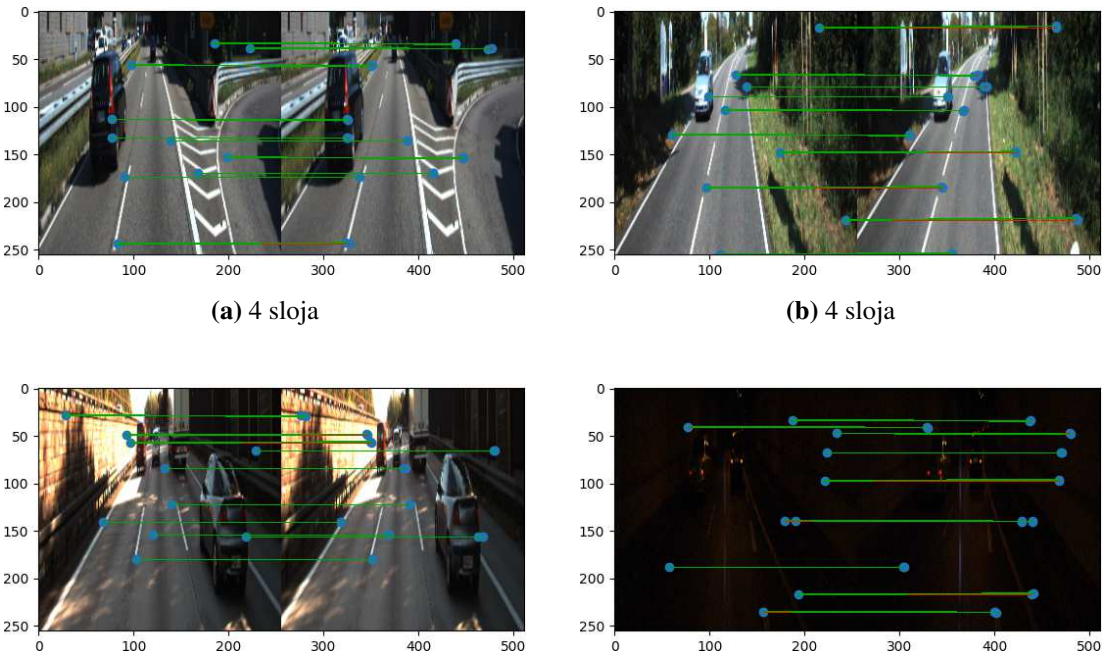
	PCK-1px	PCK-3px	PCK-5px	AEPE (px)	Loss
Train 4	17.94 %	76.79 %	96.85%	2.3486	1.60E-05
Val 4	13.15 %	56.80 %	79.00%	3.8843	5.72E-05
Train 6	51.56 %	89.52 %	97.67%	1.5433	9.39E-06
Val 6	40.83 %	74.08 %	82.53%	3.3794	6.46E-05

Tablica 7.1: rezultati modela sa 4 i 6 slojeva u transformeru na Kitti 2015

Vidimo da model sa 6 slojeva daje bolje rezultate od modela s 4 sloja, a pogotovo u metrici PCK-1px.

U slikama 7.4 prikazujemo zelenim linijama predikcije modela, dok crvenim linijama prikazujemo točne korespondencije između lijeve i desne slike na podatkovnom skupu Kitti 2015. Gornje 2 slike pripadaju modelu s 4 sloja kodera i dekodera, dok donje 2 slike pripadaju modelu sa 6 slojeva kodera i dekodera. Vidimo da zelene linije

uglavnom preklapaju crvene što indicira da model dobro pronalazi korespondencije. Naravno treba uzeti u obzir da su slike skalirane na 256 x 256 i iz tih razloga se smanjuje prostorna razlika između disparitetnih piksela, također slike su rektificirane što dodatno pojednostavljuje problem pronalaska korespondencija.



(a) 4 sloja

(b) 4 sloja

(c) 6 slojeva

(d) 6 slojeva

Slika 7.4: predikcije korespondencija s odgovarajućim korespondencijama

Nadalje provodimo eksperiment u kojima je model prilagođen stereoskopskom zadatku na način da na izlazu gledamo samo x koordinatu. Dovoljno nam je gledati samo jednu koordinatu jer je po pretpostavci rektifikacije y koordinata jednaka u parovima slika.

	PCK-1px	PCK-3px	PCK-5px	AEPE (px)	Loss
Train 4	30.34 %	89.19 %	98.36 %	2.1386	1.16E-05
Val 4	19.60 %	65.37 %	82.05 %	3.7595	5.18E-05
Train 6	67.80 %	93.29 %	98.41 %	1.3261	6.63E-06
Val 6	54.08 %	77.00 %	84.53 %	3.2186	6.57E-05

Tablica 7.2: rezultati modela sa 4 i 6 slojeva u transformeru na skupu podataka Kitti 2015, uz pretpostavku rektifikacije

Vidimo da su rezultati eksperimenta bolji od modela bez pretpostavke rektifikacije što je i za očekivati. Također možemo zaključiti da naš model još uvijek radi greške

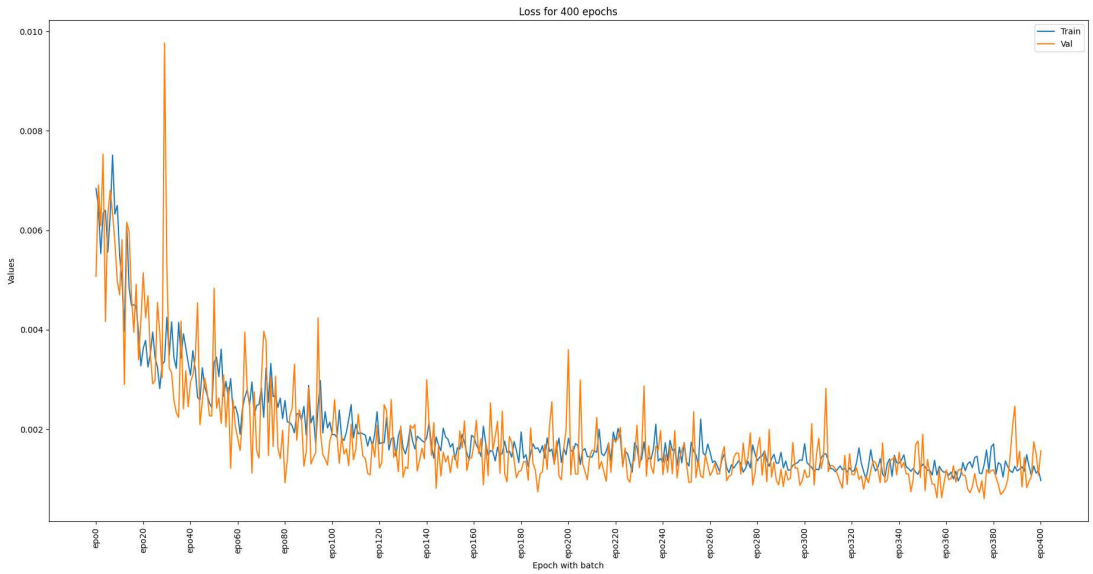
predikcije na y koordinati, iako te greške nisu značajno velike.

Model sa 6 slojeva zbog boljih rezultat i više korištenih parametara dodatno treniramo i na skupu Kitti 2015 nad kojim koristimo 2. pristup predprocesiranja koji smo već opisali. Treniramo ga dodatnih 400 epoha s istim hiperparametrima kao kod 1. pristupa. Model smo također trenirali i iz nule, ali rezultate tog eksperimenta nećemo prikazivati jer se bitno ne razlikuju od rezultata modela koji je dodatno istreniran, dok nam je model koji je dodatno istreniran koristan u primjeni.

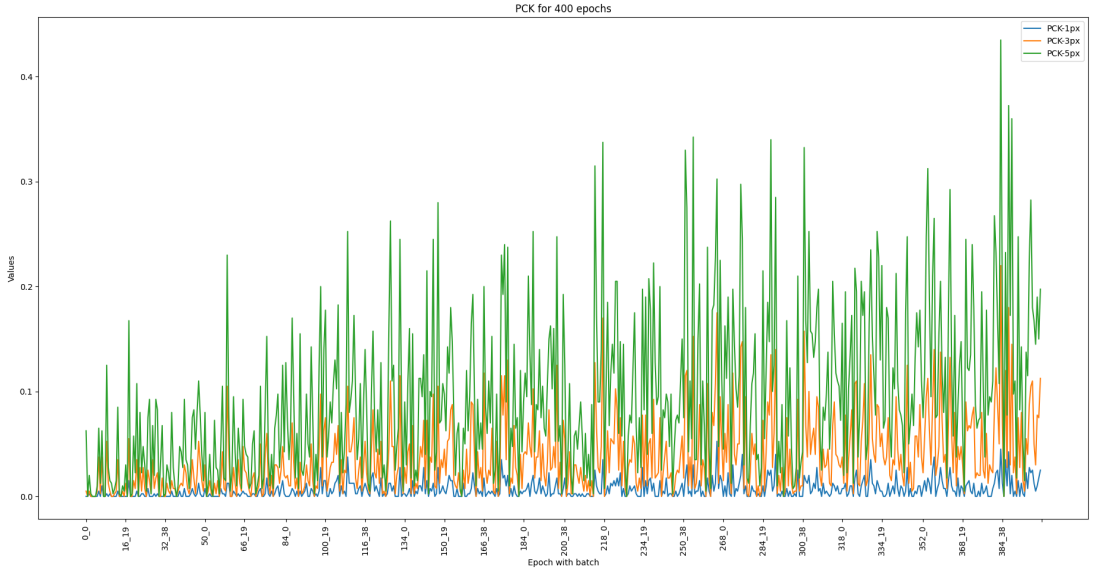
	PCK-1px	PCK-3px	PCK-5px	AEPE (px)	Loss
Train	2.33 %	11.82 %	25.31%	10.483	3.90E-04
Val	1.48 %	10.43 %	25.13%	11.198	6.90E-04

Tablica 7.3: rezultati modela korištenjem 2.pristupa na Kitti 2015

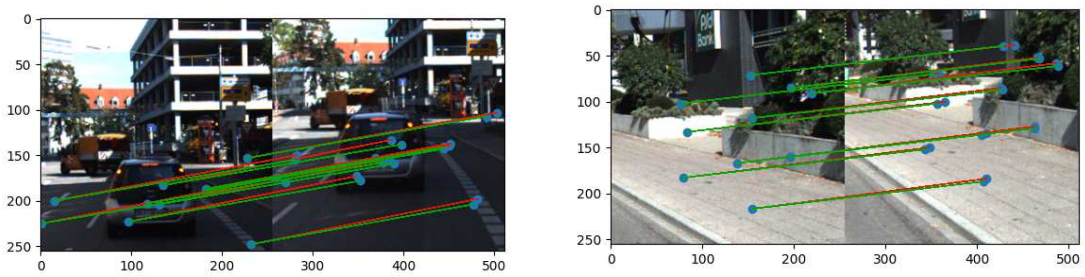
U priloženoj tablici 7.3 možemo vidjeti da je AEPE metrika oko 10px-11px što znači da se model jako muči naučiti korespondencije u slikama koje predprocesiramo 2. pristupom. Mogući razlog tome je taj što u parovima slika Kitti 2015 ima dosta točaka na ravnim površinama što je težak zadatak i za ljudsko oko dok sami detalji na slikama nisu toliko jasni. U nastavku su priloženi i grafovi funkcije gubitka i metrike PCK tijekom učenja. Graf metrike PCK također se evaluira nad nasumično odabranim točkama iz skupa valjanih točaka, što također rezultira šumom vidljivim u slici 7.6.



Slika 7.5: Funkcija gubitka 2. pristupa kroz 400 epoha



Slika 7.6: PCK metrika (eng. percentage of correct keypoints) 2. pristupa kroz 400 epoha



Slika 7.7: predikcije korespondencija s odgovarajućim korespondencijama u 2. pristupu

Iz slike 7.7 je vidljivo da predikcija korespondencije još uvijek vizualno odgovara prostoru gdje bi prava korespondencija trebala biti, jedino pri tome radi veću grešku nego kod slika 1. pristupa koji ima i jednostavniji korespondencijski problem za riješiti. Iako ovaj pristup daje lošije rezultate, kad bismo uspjeli dobiti dobre rezultate onda bi nam u primjeni ovaj pristup bio puno korisniji od pristupa gdje skaliramo cijele slike.

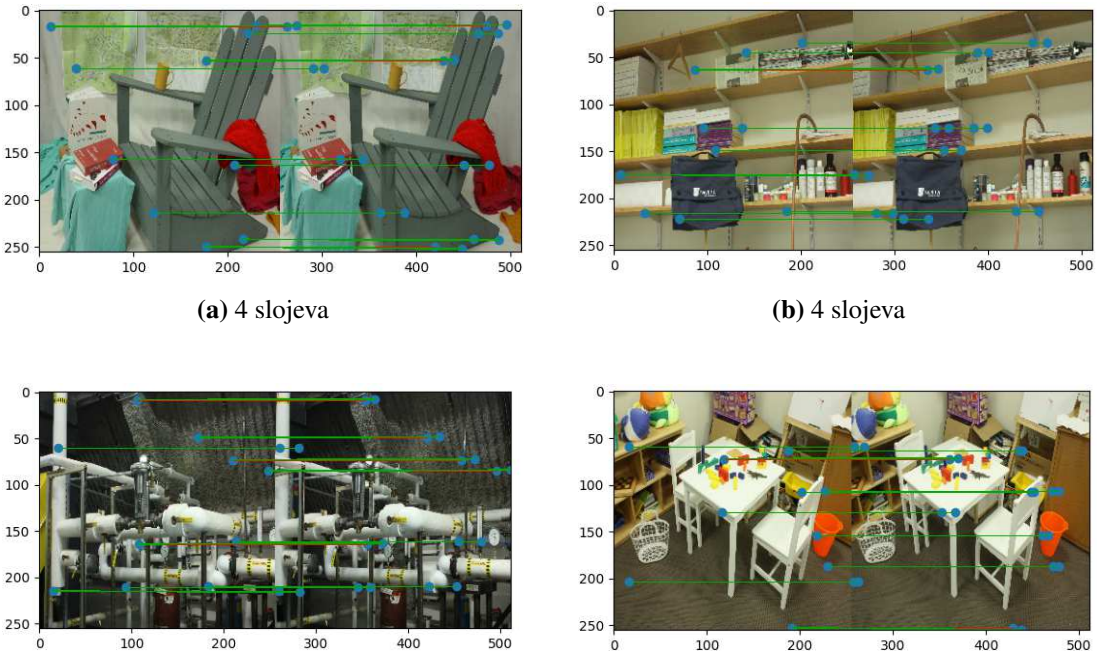
Model nadalje evaluiramo na skupu podataka Middlebury 2014. Evaluiranjem modela na skupu podataka Middlebury ćemo dobiti uvid u generalizaciju modela na promjeni domene. Domena je izmijenjena zbog korištenja različitih kamera, vrijednosti disparitеta, žarišnim vrijednostima kamera, udaljenost kamera (eng. baseline) i relevantnih pojava. Model evaluiramo na način da slike iz skupa Skup podataka Middlebury 2014

predprocesiramo koristeći sličan pristup kao i 1. pristup predprocesiranja skupa KITTI 2015. Međutim, umjesto rezanja slika, samo ih skaliramo na isječke veličine 256 x 256 piksela. Iz rezultata evaluacije 7.4 vidimo da model ima značajno lošije rezultate nego na evaluacijskom skupu iz Kitti 2015. Još uvijek pogađa neke točke u PCK 1px, 3px i 5px metrikama, greška metrike AEPE je značajno veća. Što je i očekivano zbog same izmjene domene.

	PCK-1px	PCK-3px	PCK-5px	AEPE (px)	Loss
Middlebury 4	1.9 %	5.6 %	13.3 %	16.4712	7.98E-04
Middlebury 6	5.1 %	17.50 %	23.90 %	15.9864	8.55E-04

Tablica 7.4: rezultati modela s 4 i 6 slojeva u transformeru na Middlebury 2014

U slikama 7.8 vidimo Middlebury 2014 za evaluaciju modela. Također kao i u prijašnjoj slici zelenim linija su označene korespondencijske predikcije, dok su crvenom linijom označene točne korespondencije. Na slikama vidimo da korespondencije uglavnom premaše x koordinatu dok je y koordinata točna, što implicira da model koristi vrijednost dispariteta za određene točke koji je učen iz Kitti 2015, te zbog toga radi ovu vrstu greške na Middlebury 2014 skupu. Skaliranost slika naravno utječe na vrijednost dispariteta i to je možda jedan od efekata koji je imao negativnu posljedicu na generalizaciji nad skupom Middlebury 2014.



(a) 4 slojeva

(b) 4 slojeva

(c) 6 slojeva

(d) 6 slojeva

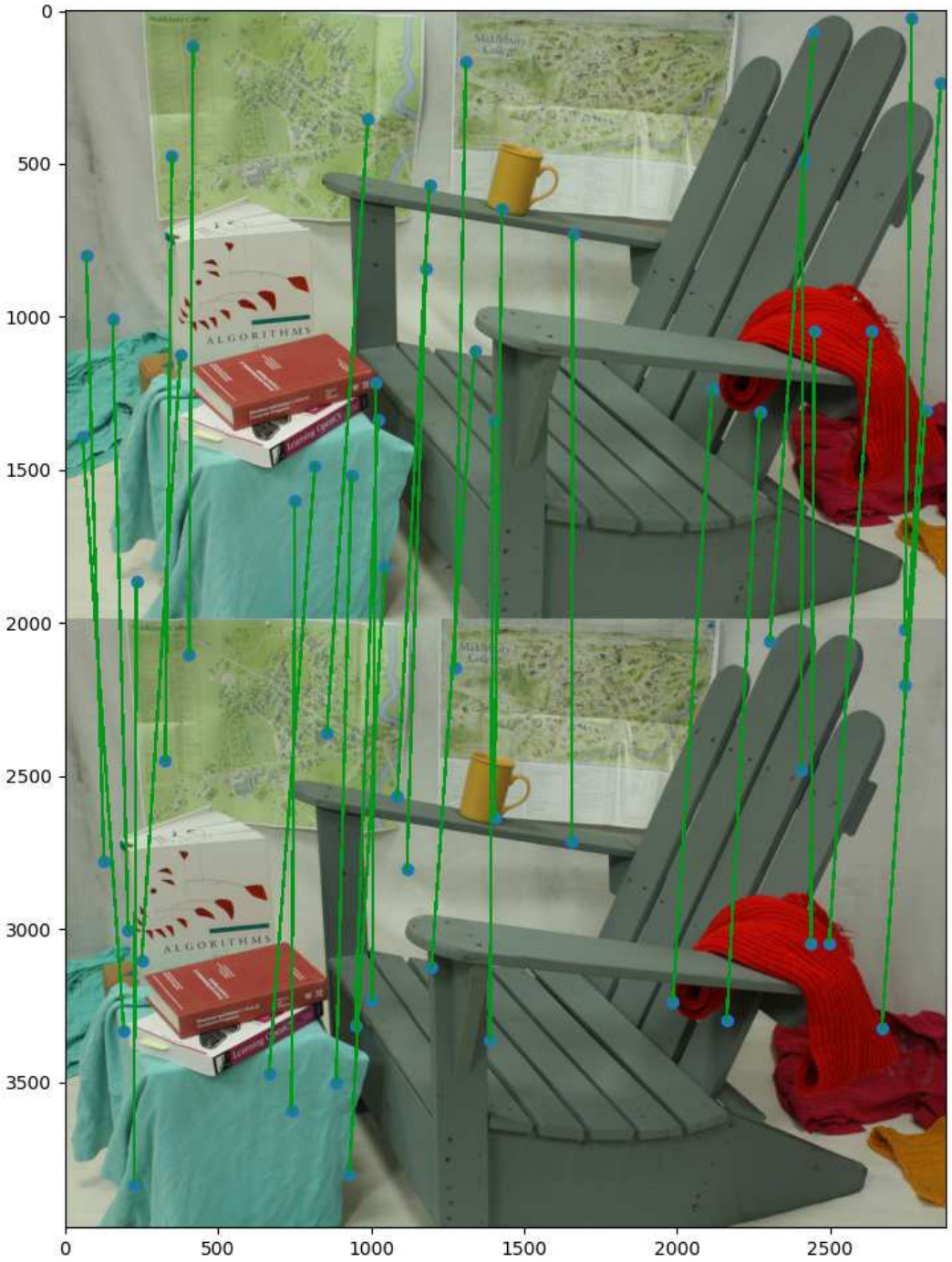
Slika 7.8

7.3. Primjena

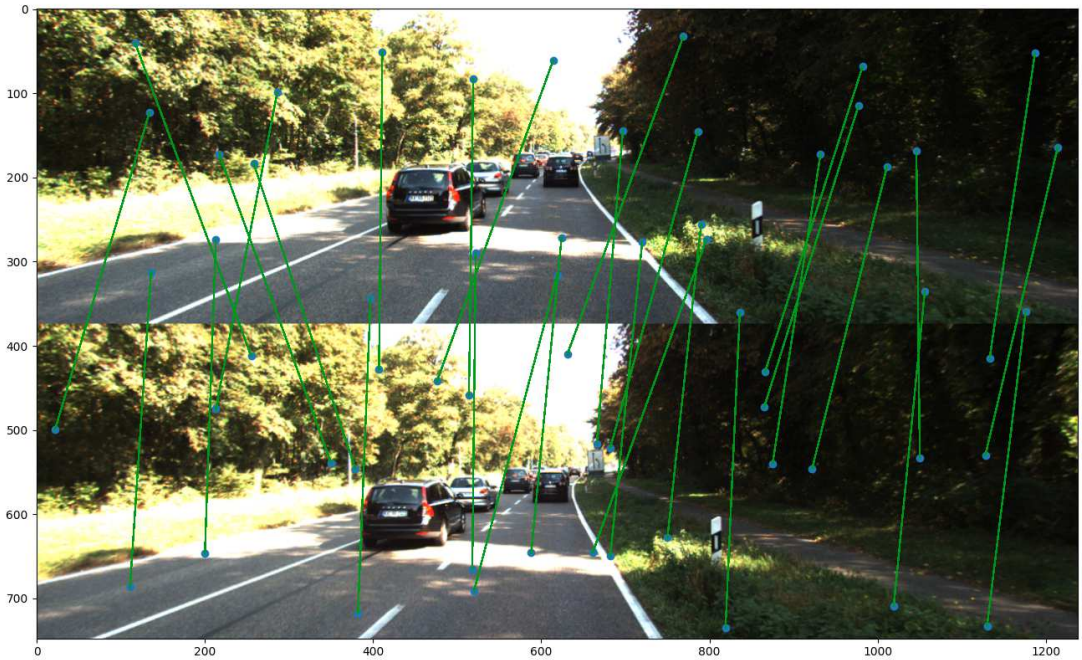
U ovom poglavlju opisujemo primjenu modela nad ulaznim slikama koje mogu biti proizvoljnih dimenzija. Da bismo mogli primijeniti model koji kao ulaz prima par slika dimenzija 256 x 256, moramo na neki način originalnu sliku podijeliti na nekoliko područja sastavljena od 256 x 256 isječaka po kojima ćemo onda tražiti korespondencije. No javlja se problem u tome što mi ne znamo unaprijed gdje trebamo tražiti korespondencije na drugom paru slika, te zbog toga za svaki ulazni prozor iz prvog para slika moramo pretraživati više prozora na drugom paru slika. Da bismo problem dodatno pojednostavnili, pretpostavit ćemo da je par slika kalibriran i rektificiran.

Prvo što radimo u pretraživanju korespondencija je da pronađemo područje na prvom paru slika koje pretražujemo i to jednostavno tako da pogledamo u kojem području su zadane točke upita. Nakon toga to područje podijelimo na isječke dimenzija 256 x 256 tako da uzmemo u obzir i mali odmak od rubova (8px) za svaki isječak. Za svaki isječak koji uzmemo iz prvog para slika uzmemo 5 isječaka iz drugog para slika. Uzmemo isječak na istoj visini i širini, te 4 ostala koja su posmaknuta po horizontalnoj osi redom -64px, -32px, 32px i 64px. Nad svim parovima isječaka (1-1, 1-2, 1-3, 1-4, 1-5) računamo korespondencije i njihovu odgovarajuću cikličku korespondenciju. Mjeru sigurnosti u predikciju izražavamo kao udaljenost cikličke korespondencije koja

bi trebala biti na istim koordinatama kao i točka upita. Sigurnost u predikciju je tim veća što je udaljenost manja. Točkama upita koje se ne nalaze u isječcima postavljamo sigurnost u predikciju na beskonačno. Konačno uzimamo one točke za koje je mjera sigurnosti najbolja.



Slika 7.9: Predikcija na slici originalne veličine iz skupa Middlebury 2014



Slika 7.10: Predikcija na slici originalne veličine iz skupa KITTI 2015

8. Zaključak

U računalnom vidu se danas znatno pomiču granice napretkom dubokog učenja. U ovom radu bavili smo se stereoskopskom rekonstrukcijom korespondencijskih točaka uz pomoć slojeva pažnje. Razmatrali smo metodu pronalaska korespondencija u parovima slika koja se temelji na korespondencijskom transformeru opisanom u radu [11]. U toj metodi se koristi arhitektura iz rada DETR [21], koji također koristi konvolucijska ugrađivanja u kombinaciji sa slojevima pažnje, a bavi se detekcijom objekata. Detaljno smo proučili navedenu metodu i eksperimentalno potvrdili njenu učinkovitost putem eksperimenta. Dobiveni rezultati ukazuju na mogućnost daljnjeg napretka u učenju našeg modela. Jedan potencijalan korak naprijed bio bi pronalazak boljeg podatkovnog skupa s većom prostornom varijacijom, poput podatkovnog skupa MegaDepth [22]. Nadalje mogli bismo model i prilagoditi rješavanju stereoskopskog zadatka tako da na izlazu tražimo samo jednu koordinatu, te pretpostavili rektifikaciju i kalibraciju slike. Model kojeg smo učili ima dosta problema u razlikovanju detalja na sličnim površinama i slabu generalizaciju nad promjenom domene. Da bismo riješili gore navedene probleme budući rad može obuhvatiti proširenje problema korespondencije na slike koje nisu kalibrirane i rektificirane, što bi dalje poboljšalo sam model. Također bi trebalo primijeniti bolju metodu u samom načinu predprocesiranja slika. Na kraju bi bilo dobro se posvetiti i boljim metodama primjene samog modela nad ulaznim slikama, kao naprimjer tehnike zumiranja kojima bismo iterativnim zumiranjem nad ulaznim slikama tražili korespondencije.

LITERATURA

[1] Python OpenCV: Epipolar Geometry. https://www.geeksforgeeks.org/python-opencv-epipolar-geometry/.

[2] OpenCV. Radial distortion. https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html.

[3] OpenCV. https://docs.opencv.org/3.4/dd/d53/tutorial_py_depthmap.html.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[5] Joseph Yacim and Douw Boshoff. Impact of artificial neural networks training algorithms on accurate prediction of property values. *Journal of Real Estate Research*, 40:375–418, 11 2018.

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[7] Siniša Šegvić. Neslužbene stranice predmeta duboko učenje 1. http://www.zemris.fer.hr/ ssegvic/du/index.shtml, 2023.

[8] Pytorch documentation. https://pytorch.org/docs/stable/generated/torch.optim.Adam.html. 2023.

[9] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[11] Wei Jiang, Eduard Trulls, Jan Hosang, Andrea Tagliasacchi, and Kwang Moo Yi. COTR: correspondence transformer for matching across images. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 6187–6197. IEEE, 2021.

[12] Pytorch: An imperative style, high-performance deep learning library. `https://pytorch.org/`, 2021.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[14] Farheen Ramzan, Muhammad Usman Khan, Asim Rehmat, Sajid Iqbal, Tanzila Saba, Amjad Rehman, and Zahid Mehmood. A deep learning approach for automated diagnosis and multi-class classification of alzheimer's disease stages using resting-state fmri and residual neural networks. *Journal of Medical Systems*, 44, 12 2019.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[16] NVIDIA Corporation. *CUDA Toolkit Documentation*. NVIDIA Corporation, 2021.

[17] Google colaboratory. `https://colab.research.google.com/`, 2021.

[18] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018.

[19] Gary Bradski and Adrian Kaehler. Opencv: Open source computer vision library. *Dr. Dobb's Journal of Software Tools*, 3(3):122–125, 2000.

[20] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nesic, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In Xiaoyi Jiang, Joachim Hornegger, and Reinhard Koch, editors, *GCPR*, volume 8753 of *Lecture Notes in Computer Science*, pages 31–42. Springer, 2014.

[21] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020.

[22] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

Određivanje korespondencija u slikama primjenom pažnje

Sažetak

Pronalaženje korespondencija u parovima slika jedan je od temeljnih problema u računalnom vidu. Pronalaženje korespondencija može biti korisno u područjima poput kalibracije kamere, optičkog toka, vizualne lokalizacije i praćenja točaka. U ovom radu rješavamo problem korespondencije u parovima kalibriranih i rektificiranih slika koristeći konvolucijska ugrađivanja i modele koji koriste slojeve pažnje. Treniramo takav model na skupu podataka KITTI 2015, na kojem provodimo eksperimente i primjene.

Ključne riječi: korespondencije , slojevi pažnje, računalni vid, stereoskopska rekonstrukcija, duboko učenje.

Detecting correspondences in images using attention

Abstract

Finding correspondences in pairs of images is one of the fundamental problems in computer vision. Discovering correspondences can be useful in areas such as camera calibration, optical flow, visual localization, and point tracking. In this paper, we tackle the correspondence problem in pairs of calibrated and rectified images using convolutional embeddings and models that employ attention layers. We train such a model on a publicly available dataset KITTI 2015, on which we conduct experiments and applications.

Keywords: correspondences, attention layers, computer vision, stereoscopic reconstruction, deep learning.