FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Algoritmy pro generování nových pohledů aplikované na exteriérové snímky automobilů focené dronem
Student:	Bc. Matěj Latka
Vedoucí:	Ing. Jakub Novák
Studijní program:	Informatika
Obor / specializace:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	do konce letního semestru 2024/2025

Pokyny pro vypracování

Většina autobazarů pořizuje prodejní fotografie automobilů zdlouhavě a složitě na otočném talíři v interiéru s neutrálním pozadím. Cílem práce je umožnit focení automobilů i v exteriéru, potlačit negativní efekty s tím spojené, např. nehomogenní osvětlení, odlesky slunce a odrazy okolních objektů na lesklé metalíze a generovat nové pohledy na automobil nad rámec pořízených fotografií tak, aby si zákazník mohl udělat lepší představu o nabízeném automobilu podobně jako by jej prohlížel fyzicky.

1) Seznamte se s existující implementací a analyzujte ji.
2) Seznamte se s datasety automobilů focených v exteriéru pomocí dronu a vyhodnoťte jejich použitelnost.
3) Proveďte rešerši technologie NeRF a jejich modifikací, metod homogenizace osvětlení a odstranění odlesků.
4) Navrhněte a implementujte vylepšení stávajícího řešení pro generování nových pohledů na scénu.
5) Navrhněte a implementujte metodu odstranění odlesků a odrazů okolních objektů z povrchu automobilů.
6) Otestujte, vizualizujte a vyhodnoťte výsledky algoritmu na dostupných datasetech.
7) Diskutujte výsledky algoritmů a porovnejte je se stávající implementací.

Existující implementace: Vítek, Martin. Návrh a implementace algoritmů pro sběr a

---

Elektronicky schválil/a Ing. Magda Friedjungová, Ph.D. dne 7. února 2024 v Praze.

analýzu fotodokumentace vozidel s využitím kamery a neuronových sítí. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2023.
Datasety: poskytne vedoucí práce

Elektronicky schválil/a Ing. Magda Friedjungová, Ph.D. dne 7. února 2024 v Praze.

Diplomová práce

ALGORITMY PRO GENEROVÁNÍ NOVÝCH POHLEDŮ APLIKOVANÉ NA EXTERIÉROVÉ SNÍMKY AUTOMOBILŮ FOCENÉ DRONEM

Bc. Matěj Latka

Fakulta informačních technologií
Katedra aplikované matematiky
Vedoucí: Ing. Jakub Novák
9. května 2024

Odkaz na tuto práci: Latka Matěj. *Algoritmy pro generování nových pohledů aplikované na exteriérové snímky automobilů focené dronem.* Diplomová práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2024.

Obsah

Seznam obrázků

Rád bych poděkoval vedoucímu práce Ing. Jakubu Novákovi za jeho čas a cenné rady při tvorbě diplomové práce. Dále děkuji Bc. Davidu Kramnému a Bc. Danielu Pilařovi za výbornou spolupráci na projektu a poskytnutí datasetů. Díky patří i mému příteli za vytvoření prostředí pro souběžné zvládnutí studia a práce a mé rodině za podporu při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 9. května 2024

Abstrakt

Práce se zabývá tvorbou systému pro generování nových pohledů na automobily focené v exteriéru. Je analyzováno proof-of-concept řešení pro modelování a vykreslování vozidel pomocí technologie NeRF a systémy pro odstraňování odlesků. Byl navržen vlastní systém, který využívá metod hlubokého učení pro odstranění odlesků z metalízy vozidel a pokročilých architektur NeRF pro modelování vozidel ve složitých exteriérových scénách. Model pro odstraňování odlesků je schopen homogenizovat vzhled metalízy dotčené odlesky slunce, případně samotného okolí vozu. Modely pro generování nových pohledů jsou za předpokladu kvalitního odhadu parametrů kamer schopny generovat realistické nové pohledy na vůz z libovolného úhlu a vzdálenosti.

Klíčová slova automobil, COLMAP, dron, NeRF, odstraňování odlesků, segmentace, strojové učení, zpracování obrazu

Abstract

This thesis focuses on implementing a system for generating novel views of the automobile captured in the exterior scene. A proof-of-concept solution for vehicle modelling and rendering using the NeRF technology and reflection removal systems has been analyzed. A system using deep learning methods for reflection removal and advanced NeRF architectures for modelling vehicles in complex outdoor scenes has been designed. The reflection removal model can unify the appearance of a vehicle's metallic degraded by the reflection either of sunlight or the vehicle's surroundings. Given that reliable estimation of camera parameters is available, novel-view-generating models can generate realistic-looking views from arbitrary angles or distances.

Keywords car, COLMAP, drone, image processing, machine learning, NeRF, reflection removal, segmentation

Seznam zkratek

GPU	Graphics Processing Unit
HPC	High-Performance Computing
IPE	Integrated Positional Encoding
LLM	Large Language Model
MIRS	Multiple Image Reflection Separation
MLP	Multi-Layer Perceptron
MVS	Multi-View Stereo
NLP	Natural Language Processing
NR-IQA	no-reference image quality assessment
PnP	Perspective-n-Point
PSNR	Peak signal-to-noise ratio
SAM	Segment Anything Model
SIRR	Single Image Reflection Removal
SIRS	Single Image Reflection Separation
SfM	Structure from Motion
VRAM	Video RAM
YOLO	You Only Look Once

Úvod

Nedílnou součástí procesu prodeje automobilu z druhé ruky je focení prodejních fotografií. Tyto fotografie musí splňovat rozličná vizuální kritéria tak, aby byl prodejní inzerát atraktivní. Je nutné automobil zabrat ze všech směrů (přední a zadní pohled, boční a rohové pohledy), ostře a tak, aby byl automobil hlavním objektem ve scéně. Ne vždy je však možné takové snímky pořídit, ať už kvůli neznalosti prodávajícího, jeho nedůslednosti, či objektivním překážkám v prostředí, kde je automobil snímán. U exteriérových fotografií jsou dále kladeny vysoké požadavky na prostředí, ve kterém jsou fotografie pořizovány. Je vhodné, aby se na pozadí nenacházelo nic rušivého nebo neestetického, co by zbytečně odvádělo zákazníkovu pozornost, popř. zcela kazilo estetické kvality zabírané scény. Autobazary zpravidla pro tyto účely disponují místností s neutrálním pozadím a otočným talířem, na který je automobil umístěn a nafocen ze všech úhlů. Pořizování prodejních fotografií v těchto podmínkách je však kvůli pořizovacím nákladům na otočný systém drahé a neflexibilní. Práce si klade za cíl odbourat problémy s focením v exteriéru (odlesky, neestetické pozadí), vytvořit reprezentaci automobilu na základě několika pořízených snímků tak, aby bylo možné vygenerovat snímky nové, umožnit tvorbu kvalitních prodejních fotografií kdekoliv a zjednodušit tak proces prodeje.

Kapitola 1

Rešerše

Algoritmy pro tvorbu reprezentace scény a generování nových pohledů prošly v posledních letech dynamickým vývojem, kdy se alternativou klasických fotogrammetrických přístupů staly algoritmy strojového učení, zejména NeRF [1]. Na úspěch této architektury navázal další výzkum, který přišel s mnohými optimalizacemi co do kapacity modelu i výpočetní náročnosti, která umožnila efektivní nasazení těchto algoritmů i na složitějších scénách.

K odstranění odlesků se ve fotografii tradičně používá polarizační filtr, který je mj. schopen potlačit některé odlesky, zejména ty, které vznikly odrazem od lesklého nekovového povrchu. Odlesky vzniklé na povrchu automobilu jsou však odlesky jiného druhu. Tam, kde polarizační filtr nelze použít, mohou pomoct algoritmy pro odstranění odlesků.

1.1 Generování nových pohledů na scénu

Těžištěm práce je využití nasnímaných snímků pro tvorbu reprezentace zabrané scény, resp. vozidla samotného a následné generování snímků zabírajících vozidlo z nových úhlů a jiných vzdáleností.

1.1.1 Fotogrammetrie

Jednou z možností, jak takovou reprezentaci získat, je explicitně vytvořit 3D model snímané scény. Tradiční technikou pro tuto úlohu je fotogrammetrie [2], která na vstupu přijímá částečně překrývající se snímky pořízené z různých úhlů, které jako celek pokrývají celou scénu. Z těchto snímků jsou extrahovány příznaky, nejčastěji SIFT [3] nebo SURF [4] deskriptory. Následně jsou hledány korespondence deskriptorů mezi jednotlivými snímky, tedy párování deskriptorů nacházejících se na různých snímcích, ale popisujících stejné místo snímaného objektu. Body odpovídající těmto korespondencím jsou následně triangulovány, tedy jsou vypočítány jejich pozice v prostoru. Z těchto bodů jsou následně aproximovány povrchy ve scéně, na které je namapována barva či textura.

1.1.2 Neural Radiance Fields

Neural Radiance Fields (NeRF) [1] představují novou technologii pro reprezentaci scény a generování nových pohledů. NeRF se odklání od explicitní reprezentace scény jako 3D modelu a namísto toho scénu reprezentuje jako spojitou funkci objemu, kterou modeluje pomocí vícevrstvé neuronové sítě (MLP). Tato funkce uchovává informaci, jaké barvy a svítivosti nabývá příslušný bod při pohledu z určitého úhlu. Pokud model dokáže správně generalizovat přes různé úhly a pozice, je schopen vytvořit realistické pohledy na celou scénu i z dříve nezabraných úhlů. Většina implementací NeRF na vstupu vyžaduje kromě snímků i pozice příslušných kamer, jelikož je potřeba znát úhel, ze kterého byla scéna v příslušném trénovacím snímku nahlížena. Nejrozšířenějším způsobem, jak tuto polohu získat, je použití softwaru COLMAP (podrobněji popsán v sekci 2.1), který kromě těchto poloh vrací i point cloud bodů zrekonstruovaných ze zabrané scény. Podrobněji je NeRF popsán v sekci 2.2.1.

Model NeRF sice pro mnohé datasety dosahoval velmi dobrých výsledků, tyto datasety však obsahovaly maximálně ohraničené scény (byť zabírané z 360°), nebo i neohraničené scény, ty však byly zabírány pouze jedním směrem (forward-facing scenes). Model NeRF++ [5] rozšiřuje využití NeRF i na snímky prostorově neohraničených scén zabraných 360° ze všech stran. Takové scény obsahují mnohem více objemu a je tak obtížnější je modelovat. NeRF++ proto zavádí nový způsob parametrizace bodů (sekce 2.2.2) vybíraných pro trénování a vykreslování. Motivací pro tuto parametrizaci je potřeba různé granularity výběru bodů. Granularita je úměrná tomu, zda příslušné body reprezentují objem nacházející se na popředí, nebo na pozadí. Objekty v popředí je potřeba reprezentovat detailně, naproti tomu scéna v pozadí tolik informace nést nemusí. Snaha o univerzální zlepšení v celém prostoru zabrané scény by na prostorově neomezených scénách vedla k přílišné výpočetní náročnosti a tím i plýtvání výpočetními zdroji a časem, jelikož v takových scénách se zejména na pozadí nachází mnoho oblastí, které pro scénu jako celek nejsou příliš důležité.

Mip-NeRF [6] se snaží řešit problémy rozostřených a aliasingem trpících pohledů vykreslených z NeRF. Jeho technika vzorkování scény v bodech podél paprsků směřujících do scény a spojování této informace do jednoho barevného pixelu je sice vhodná pro pohledy, které scénu zabírají z přibližně stejné vzdálenosti, pokud však trénovací snímky zachycují scénu z více různých vzdáleností, jsou nové pohledy rozostřené a zejména pohledy z větší dálky mohou obsahovat různé artefakty, které se na trénovacích snímcích nenacházely. Řešení používané v offline ray-tracingu, totiž vzorkování několika různých paprsků směřujících skrz totožný pixel, je vzhledem k výpočetní náročnosti NeRF (průchod hlubokou neuronovou sítí pro každý vzorek na každém paprsku) nepraktický. Mip-NeRF se inspiruje technikou mipmappingu [7], která se v počítačové grafice používá pro zrychlení vykreslování a eliminaci aliasingu. Jeho použití je podrobněji popsáno v sekci 2.2.3.

Mip-NeRF 360 [8] je nadstavba nad Mip-NeRF vytvořená pro scény, kde je objekt zájmu, ležící ve středu neohraničené scény, zabrán z 360° po kružnicové trajektorii. Zavádí tři nové koncepty: parametrizaci souřadnic v neohraničené scéně do omezeného oboru hodnot, optimalizaci průchodu neuronovou sítí, která scénu reprezentuje, a prevenci potenciálně nejednoznačného vykreslování objektů na pozadí neohraničené scény. Tyto objekty jsou totiž zachytitelné pouze malým množstvím paprsků, podél kterých NeRF vzorkuje zabíraný prostor. Tyto koncepty jsou podrobněji popsány v sekci 2.2.4.

Zip-NeRF [9] si klade za cíl zkombinovat výhody Mip-NeRF, který umí předcházet aliasingu (viz sekce 2.2.3) s výhodami optimalizovaných rychlých mřížkovitých technologií, např. Instant NGP [10], která používá 3D pyramidu mřížek s různou granularitou pro konstrukci příznaků, které jsou pak zpracovávány malou MLP, a to tak, že zavádí pyramidu inspirovanou Instant NGP do Mip-NeRF 360 (viz sekce 2.2.4). Tyto technologie však vykazují několik nekompatibilit: pyramida používaná Instant NGP je nekompabitilní s IPE v Mip-NeRF 360, jelikož deskriptory

získané v Instant NGP by byly aliasovány vzhledem k jejich souřadnicím v prostoru, což by způsobilo aliasing i ve vykreslených obrázcích. Destilace znalosti z NeRF neuronové sítě do návrhové sítě v Mip-NeRF 360 také vede k tomu, že při pohybu kamery ve scéně se může část obsahu ztratit. Jak tyto koncepty Zip-NeRF kombinuje dohromady, popisuje sekce 2.2.5.

Alternativou reprezentace zářivého pole pomocí spojité objemové funkce je reprezentace množinou 3D gaussiánů. To využívá Gaussian Splatting [11], který se snaží překonat zásadní problém popsaných NeRF modelů, a sice výpočetní náročnost a související délku trénování i inference. Ačkoliv je technologie NeRF schopna se věrně naučit i velmi složité scény, trénování může v případě Mip-NeRF 360 zabrat až 48 hodin. Vykreslování nových pohledů ve vysokém rozlišení pak může zabrat až 15 minut pro jeden obrázek. Cílem metody je umožnit v reálném čase (alespoň 30 snímků za sekundu) vykreslování pohledů ve Full HD rozlišení. Zásadním přínosem metody pro experimenty v této práci bylo snížení trénovacího času pro složité scény až na 30 minut a zrychlení generování nových pohledů až na desítky snímků za sekundu. Podrobněji je metoda popsána v sekci 2.3.

1.2 Odstraňování odlesků

V problematice odstraňování odlesků se prosadily dva různé přístupy – odstraňování odlesků s využitím více obrázků najednou (MIRS – Multiple Image Reflection Separation) nebo odstraňování odlesků s využitím pouze jediného obrázku (SIRS – Single Image Reflection Separation). MIRS využívá více snímků pro získání robustnější informace pro oddělení odlesku z původního snímku. Více takové informace může přinést i různé nastavení polarizačního filtru při pořizování jednotlivých snímků. Tyto manuální zásahy do snímací soustavy (instalace polarizačního filtru na objektiv) však není možné provádět při pořizování snímků dronem či dokonce mobilním telefonem, proto tato metoda nebyla v práci použita.

SIRS (nebo také SIRR – Single Image Reflection Removal) naproti tomu pracuje pouze s jedním obrázkem I, na který nahlíží jako na vážený součet základního snímku T a odrazu (např. při focení scény skrz sklo) R, tedy

$$I = g(T) + f(R) \tag{1.1}$$

přičemž funkce $g(\cdot)$ a $f(\cdot)$ reprezentují defekty, ke kterým dojde na základním snímku, ale zejména na odrazu, např. rozostření, zesvětlení, posun, distorze aj. Pro tento problém zatím existuje jen nemnoho datasetů, nejpoužívanější jsou např. dataset Real20 ([12], Obrázek 1.1a) nebo dataset SIR^2[13] a jeho části Postcard (Obrázek 1.1b), Solid Object (Obrázek 1.1c) a Wild (Obrázek 1.1d).

(a) Dataset Real20. **(b)** Dataset SIR^2, část Postcard. **(c)** Dataset SIR^2, část Solid Object. **(d)** Dataset SIR^2, část Wild.

Obrázek 1.1 Příklady datasetů pro SIRR.

Při evaluaci SIRR modelů jsou porovnány groung-truth snímky bez odlesků a zrekonstruované snímky s odstraněnými odlesky. Používanou metrikou je PSNR (Peak signal-to-noise ratio), která je definována jako

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{\mathrm{MAX}_i^2}{\mathrm{MSE}}\right) = 20 \cdot \log_{10}\left(\frac{\mathrm{MAX}_I}{\sqrt{\mathrm{MSE}}}\right), \tag{1.2}$$

kdy MAX_I je maximální hodnota pixelu v obrázku (zpravidla 255) a MSE je střední kvadratická chyba definovaná pro dva c-kanálové obrazy I a K o velikosti $m \times n$ jako

$$\mathrm{MSE} = \frac{1}{mnc}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\sum_{k=0}^{k-1}||I_{i,j,k} - K_{i,j,k}||^2 . \tag{1.3}$$

Jednotkou PSNR je decibel. Čím vyšší PSNR, tím kvalitněji byl obrázek zrekonstruován.

Podle [14] mohou mít SIRS metody problémy ve složitých neohraničených scénách v reálném prostoru. Tomu se částečně daří předejít použitím metod hlubokého učení.

CEILNet [15] pro extrakci odrazových vrstev používá odhad toho, nakolik je příslušná část textury hladká, pro extrakci průhledové vrstvy zase používá odhad hran ve scéně. Na základě těchto odhadů pak rekonstruuje jednotlivé textury. Pracuje pouze s lokálními příznaky, nepomáhá si globální sémantickou informací, což řeší Zhang a kol. [12], kteří tuto informaci získávají syntézou vstupního snímku s deskriptory získanými jako výstupy některých vrstev předtrénované konvoluční sítě VGG-19 [16]. ERRNet [17] zase umožňuje pracovat s novým typem trénovacích dat, kdy originální obrázek I a jeho průhledová vrstva T nezabírají scénu z přesně stejného místa, ale dovoluje mezi nimi jistý posun. Všechny zmíněné sítě však odhadovaly pouze průhledovou vrstvu, nezabývaly se predikcí samotného odlesku. RAGNet [18] se naproti tomu snaží originální obrázek dekomponovat úplně, kdy nejprve predikuje odrazovou vrstvu a až pak vrstvu průhledovou. Pro její predikci využívá také informace získané predikcí odrazové vrstvy. Kompozice výsledného obrázku však není pouze prostý součet těchto vrstev. Promlouvají do něj funkce $f(\cdot)$, $g(\cdot)$ (viz rovnice 1.1), které je potřeba také odhadnout. BDN [19] začíná odhadem prostých lineárních funkcí, zatímco jiné práce [20] přechází i k funkcím nelineárním.

Na portálu Papers with Code je k dispozici porovnání různých metod pro odstraňování odlesků [21]. Porovnávány jsou hodnoty PSNR na výše zmíněných datasetech. Nejlepších hodnot na téměř všech datasetech dosahuje model DSRNet [14], který kromě průhledové a odrazové vrstvy ještě pracuje s dekompozičním reziduem, které umožňuje obě zmíněné vrstvy odhadnout daleko přesněji a zbavit se různých efektů nevysvětlitelných funkcemi $f(\cdot)$ a $g(\cdot)$ jejich přesunutím do tohoto rezidua. Model je podrobněji popsán v sekci 2.4.

1.3 Segmentace

Segmentace vozidla je poměrně obtížný úkol, který je nutné řešit pokročilými segmentačními technikami. Klasické segmentační algoritmy jsou totiž schopny segmentovat pouze na základě barvy, což pro komplexní a barevně nehomogenní strukturu vozu není dostačující. Naopak je nutno pracovat se specifickou sémantickou informací, pro jejíž extrakci jsou vhodné konvoluční neuronové sítě, které pomocí konvolučních vrstev předtrénovaných na velkých datasetech [22, 23] ze snímků extrahují generické příznaky, které následně zpracovávají a klasifikují do požadovaných tříd. Úlohu segmentace v kontextu hlubokého učení lze rozdělit na segmentaci jednotlivých objektů (jejich instancí), kdy je klasifikovány pouze pixely odpovídající detekovaným objektům ve scéně, a sémantickou segmentaci, kdy jsou klasifikovány všechny pixely ve snímku (včetně např. oblohy, trávníku a podobně).

Důležitým modelem pro instance segmentation je Mask R-CNN [24], který rozšířil do té doby nejpoužívanější konvoluční architekturu pro detekci a klasifikaci objektů Faster R-CNN [25] o novou

hlavu pro predikci segmentační masky. Takovou síť lze metodami transfer learningu [26] adaptovat na segmentaci i velmi neobvyklých objektů, například vad skleněných tyčí [27]. Zásadní problém těchto architektur je čas výpočtu – i když Faster R-CNN zavedl mnohé optimalizace oproti původním konvolučním architekturám, Mask R-CNN je schopen zpracovat pouze asi 5 snímků za sekundu.

Snahy o konvoluční architekturu schopnou detekce a klasifikace v reálném čase i např. videa s desítkami snímků za sekundu vedly k architektuře YOLO [28], jejíž novější varianty [29, 30, 31] jsou mj. schopné i instance segmentation. Podrobněji je tato architektura popsána v sekci 2.5.1.

Rozmach architektury transformerů [32] se projevil i v problematice instance segmentation. Známý je model DETR (Detection Transformer) [33], který využívá standardní konvoluční backbone pro extrakci příznaků. Ty následně umístí do jednoho vektoru, informaci o původní pozici zachová pozičním kódováním jednotlivých příznaků a takový vektor předá transformeru. Jeho výstup je předán malé síti, která predikuje detekce. Je též možné přidat hlavu predikující masky pro instance segmentation.

Závislosti na konvoluční backbone se zbavil Vision Transformer [34]. Ten pracuje přímo s obrázkem, který rozdělí na mřížku 16×16. Tyto výřezy pak embeddované předává transformeru.

Univerzálnější použití těchto architektur nabízí model GLEE [35], který umožňuje detekci, segmentaci, trackování a identifikaci téměř libovolných objektů ve scéně. Umí se také adaptovat na nové úkoly podobného typu bez potřeby dalšího trénování (zero-shot transfer). Je také možné jej integrovat do velkých jazykových modelů (LLM, Large Language Models), kterým poskytuje univerzální informace o různých objektech, které ve snímku detekuje. Tento model je podrobněji popsán v sekci 2.5.2.

Kapitola 2

Teoretické zázemí

Tato kapitola poskytuje základní teoretické informace k technikám a modelům použitým v práci. Nejprve je představen software pro odhad pozic a parametrů kamery, následně jsou popsány techniky typu NeRF od základního modelu až po pokročilé nadstavby. Nakonec jsou představeny některé modely pro odstranění odlesků a segmentaci.

2.1 COLMAP

COLMAP [36, 37] je software pro částečnou rekonstrukci 3D scény z 2D snímků. Pracuje tak, že pomocí SfM [36] nejprve extrahuje řídký (sparse) point cloud jako prvotní reprezentaci scény a pozice kamer. Body z tohoto point cloudu předá algoritmu MVS [37], který point cloud zahustí (dense point cloud).

Algoritmus SfM (Obrázek 2.1) přijímá na vstupu množinu snímků určité scény. Očekává se, že snímky budou objekt zabírat z různých úhlů, nicméně je nutné, aby se vždy alespoň dva různé snímky překrývaly. V těchto snímcích nejprve proběhne extrakce deskriptorů (COLMAP používá SIFT deskriptory [3]), ale lze použít i deskriptory jiné. Následně je proveden matching těchto deskriptorů (hledání korespondujících podobných deskriptorů z různých snímků) a jejich geometrická verifikace – odstranění sice vizuálně podobných, ale geometricky si neodpovídajících korespondencí, zpravidla je používán algoritmus RANSAC [38] či jeho modifikace. Nakonec jsou postupně odhadovány parametry kamer a rekonstruovány některé body ve scéně a to v několika krocích:

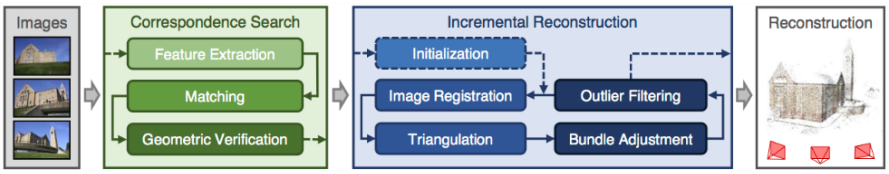
1. **inicializace**: výběr prvních dvou snímků pro rekonstrukci.
2. **přidávání dalších snímků (image registration)**: nové snímky, (resp. nové kamery) jsou do scény přidávány (registrovány) řešením Perspective-n-Point (PnP) problému (odhad relativní pozice kamery ve scéně a jejích vnitřních parametrů) s použitím 2D-3D korespondencí z již registrovaných snímků.
3. **triangulace**: nově registrovaný obrázek by již měl mít v záběru dříve viděné korespondence. Tuto množinu pak rozšiřuje triangulací dalších bodů. Nový bod může být triangulován, jestliže už kromě aktuálního snímku byl viděn v některém z dříve registrovaných snímků.
4. **bundle adjustment**: nepřesně odhadnuté pozice a parametry kamer P_c a nepřesně triangulované body X_k se mohou navzájem ovlivňovat natolik, že SfM může zcela divergovat. Proto

je potřeba využít zpřesňující informaci, kterou do scény dodávají triangulace nových bodů, pro optimalizaci těchto parametrů. To provádí algoritmus bundle adjustment [39], který nelineárně upravuje parametry P_c a X_k tak, aby byla minimalizována reprojekční chyba

$$E = \sum_j \rho_j \left(||\pi\left(P_c, X_k\right) - x_j||_2^2 \right), \tag{2.1}$$

kdy π je projekce bodů z 3D prostoru do 2D snímku a ρ_j je ztrátová funkce, která zmenšuje vliv geometricky nepřesných korespondencí.

Výstupem COLMAP je 3D point cloud a odhad vnitřních a vnějších parametrů kamery pro každý snímek.



Obrázek 2.1 Schéma algoritmu SfM. Zdroj obrázku [36].

2.2 Neural Radiance Fields

Tato sekce poskytuje teoretický základ pro základní model NeRF a jeho nadstavby, které se zaměřují na snížení výpočetní náročnosti a adaptaci na složité neohraničené scény.

2.2.1 NeRF

Trénování NeRF začíná náhodným výběrem množiny paprsků ze všech pixelů všech trénovacích snímků $\mathcal{R}$ směřujících z kamery do scény. Každý paprsek $r \in \mathcal{R}$, $r(t) = o + td$ začíná v optickém středu kamery o a po směru d směřuje do scény tak, že zároveň prochází příslušným pixelem. Na těchto paprscích jsou ve dvou fázích vybírány množiny bodů (Obrázek 2.2 a)). Nejprve je vybráno 64 N_c bodů, které se nachází na paprsku s rovnoměrnými rozestupy. Tyto body jsou vyhodnoceny a poskytnou váhy odpovídající rozložení objemu ve scéně. Tyto váhy lze interpretovat jako bodovou diskrétní hustotu pravděpodobnosti výskytu viditelného objemu ve scéně. Na základě těchto vah je pak informovaně vybráno 128 N_f bodů, které systematicky tíhnou k místům s větším množstvím objemu. Prostorová informace je pak po složkách přemapována z $\mathbb{R}$ do vícedimenzionálního prostoru $\mathbb{R}^{2L}$ podle vztahu

$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \ldots, \sin\left(2^{L-1}\pi p,\right), \cos\left(2^{L-1}\pi p\right)\right), \tag{2.2}$$

všech 192 bodů je seřazeno a jsou dále předány neuronové síti.

Ta na vstupu přijímá pětidimenzionální vektor, který kromě zakódované pozice $\mathbf{x} = (x, y, z)$ obsahuje i směr pohledu na scénu v podobě dvou úhlů (θ, φ). Výstupem sítě je vyzářená barva $\mathbf{c} = (r, g, b)$ a hustota σ (Obrázek 2.2 b)):

$$\forall t_k \in t, [\sigma_k, c_k] = \mathrm{MLP}\left(\gamma(r(t_k)); \Theta\right), \tag{2.3}$$

kdy t_k je vzdálenost od optického středu kamery, MLP je neuronová síť jako funkce a Θ jsou její parametry. Síť je optimalizována tak, aby byla reprezentace scény konzistentní přes pohledy z různých směrů.

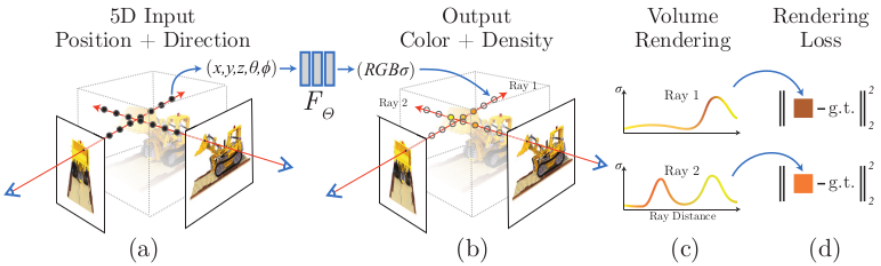
Na tato data jsou aplikovány techniky vykreslování objemu (volume rendering, [40], Obrázek 2.2 c)), které poskytnou výslednou podobu obrazu. Barva příslušného pixelu $C(r)$ je získána jako

$$\begin{aligned} C(r) &= \int_{t_1}^{t_2} T(t) \cdot \sigma\left(r(t)\right) \cdot c\left(r(t), d\right) dt \\ &\approx \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) c_i, \text{ kde } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right), \end{aligned} \tag{2.4}$$

a $\delta_i = t_{i+1} - t_i$ je vzdálenost mezi sousedními vzorkovanými body.

Celý proces včetně rovnice 2.4 je diferencovatelný a proto je možné jej optimalizovat gradientním sestupem. Ztrátová funkce je definována jako součet kvadratické chyby vykreslené a reálné barvy příslušného pixelu (Rovnice 2.5 a Obrázek 2.2 d)), kdy $\hat{C}_c$ reprezentuje barvu pixelu odpovídajícího bodu z neinformovaného výběru a $\hat{C}_f$ barvu pixelu příslušejícímu bodu z informovaného výběru.

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \left[\left|\left| \hat{C}_c(r) - C\left(r; \Theta_c, t_c\right) \right|\right|_2^2 + \left|\left| \hat{C}_f(r) - C\left(r; \Theta_f, \text{sort}\left(t_c \cup t_f\right)\right) \right|\right|_2^2 \right] \tag{2.5}$$

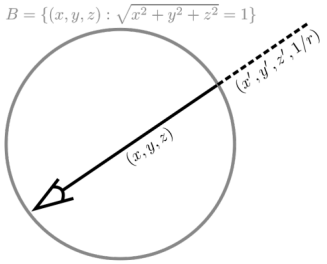


Obrázek 2.2 Schéma NeRF. Zdroj obrázku [1].

2.2.2 NeRF++

NeRF++ dělí scénu na dvě části – vnitřní scénu reprezentovanou jednotkovou koulí a vnější scénu reprezentovanou inverzní koulí komplementární k vnitřní jednotkové kouli. Každou tuto část reprezentuje samostatný NeRF, přičemž pro získání finální barvy příslušného pixelu je zkombinován výstup obou modelů. Body $\mathbf{x} = (x, y, z)$, $r = \sqrt{x^2 + y^2 + z^2}$ patřící do vnější koule jsou reparametrizovány čtveřicí $\mathbf{x}' = \left(x', y', z', \frac{1}{r}\right)$, $x'^2 + y'^2 + z'^2 = 1$, kde (x', y', z') je jednotkový vektor se stejnou směrnicí jako vektor (x, y, z) reprezentující směr paprsku skrz kouli. Parametr $\frac{1}{r} \in \langle 0; 1\rangle$ je inverze poloměru, přičemž samotný bod nacházející se mimo kouli je definován jako $\mathbf{x} = r \cdot (x', y', z')$ (viz také Obrázek 2.3). Tato reprezentace znamená, že všechny parametry popisující příslušný bod mají dobře definovaný omezený obor hodnot, jelikož $x', y', z' \in \langle -1; 1\rangle$, $\frac{1}{r} \in \langle 0; 1\rangle$. To jednak zlepšuje numerickou stabilitu výpočtů, ale zejména určuje, že vzdálenější objekty budou mít nižší rozlišení.

Na tuto parametrizaci lze také nahlížet jako na bod v prostoru $\mathbf{x} = (x, y, z)$ promítnutý do obrazové roviny jako pixel (x', y', z'), přičemž inverze poloměru $\frac{1}{r}$ reprezentuje disparitu, tedy inverzní hloubku příslušného bodu.



Obrázek 2.3 Parametrizace inverzní koulí zavedená v NeRF++. Zdroj obrázku [5].

2.2.3 Mip-NeRF

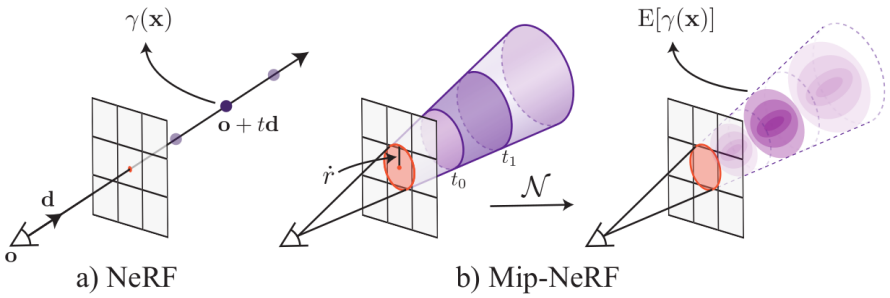
Mipmapping [7] je označení pro sekvenci obrázků, kdy každý další snímek v sekvenci je v každém směru k-krát subsamplovaný snímek předcházející. Při vykreslování scény se pro objekty nacházející se blízko využijí obrázky s vysokým rozlišením, pro objekty nacházející se dál zase obrázky s nižším rozlišením. Mip-NeRF tuto techniku využívá tak, že předfiltrované zářivé pole reprezentuje jako spojitý prostor naškálovaných pohledů. Slučuje také dvě sítě, které NeRF používal samostatně pro reprezentaci bodů z neinformovaného a informovaného výběru, do jedné, díky čemuž je model rychlejší a má jen polovinu parametrů.

Mip-NeRF se snaží předcházet aliasingu změnou vzorkování podél jednotlivých paprsků. Oproti NeRF, který z kamery do scény vysílá paprsek, Mip-NeRF vysílá kužel. Díky tomu nejsou vzorkovány body podél paprsků, ale komolé kužely uvnitř příslušného kužele (viz také Obrázek 2.4). Vzhledem k tomu, že nyní nepracujeme s bodem, ale částí prostoru, je nutné upravit původní poziční kódování (rovnice 2.2). Mip-NeRF zavádí integrované poziční kódování (integrated positional encoding, IPE), které kóduje i velikost a tvar komolého kuželu, nikoli pouze jeho pozici. Nejprve je stanoven paprsek $r(t) = o + td$ vedoucí z pozice v rovině kamery o směrem d dovnitř scény. Dále stanovena množina intervalů vzdáleností od počátku paprsku $T_i = [t_i; t_{i+1})$. Tyto vzdálenosti definují řezy kuželem, ze kterých jsou sestaveny jednotlivé komolé kužely vzorkující příslušnou část objemu ve scéně. Pro zjednodušení výpočtu jsou tyto komolé kužely aproximovány vícedimenzionálním gaussiánem. Pro každý takový gaussián je vypočítán průměr a kovarianční matice $(\mu, \Sigma) = r(T_i)$. Na tyto hodnoty je aplikováno IPE, které kóduje pozici, velikost a tvar postupně $n + 1$ vybraných komolých kuželů podle následujícího vztahu

$$\gamma(\mu, \Sigma) = \left\{ \begin{bmatrix} \sin\left(2^l \mu\right) \exp\left(-2^{2l-1} \mathrm{diag}\left(\Sigma\right)\right) \\ \cos\left(2^l \mu\right) \exp\left(-2^{2l-1} \mathrm{diag}\left(\Sigma\right)\right) \end{bmatrix} \right\}_{l=0}^{L-1}. \tag{2.6}$$

Takto zakódované informace jsou předány neuronové síti, která vrací hustotu σ_k a barvu c_k. Vykreslování probíhá podle rovnice 2.4.

Vzorkování v NeRF probíhalo ve dvou fázích – neinformované a informované, přičemž každý typ vzorků měl svou vlastní neuronovou síť. Takový postup byl nutný proto, že model byl schopen se naučit vzhled scény pouze pro jedno rozlišení. Mip-NeRF však díky použití komolých kuželů a IPE, které kóduje mj. informaci o rozlišení, umožňuje použití jediné neuronové sítě pro všechny vzorky. Tato síť je následně používána pro renderování. Díky jednoduššímu modelu je možné zjednodušit i ztrátovou funkci



Obrázek 2.4 Porovnání vzorkování prostoru podle NeRF a Mip-NeRF. Na obrázku a) je demonstrováno vzorkování NeRF pomocí bodů na paprsku. Na obrázku b) je ilustrováno vzorkování Mip-NeRF pomocí komolých kuželů a jejich aproximace vícedimenzionálními gaussiány s uplatněním pozičního kódování γ. Zdroj obrázku [6].

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \left[\lambda \left|\left| \hat{C}_c(r) - C(r; \Theta, t_c) \right|\right|_2^2 + \left|\left| \hat{C}_f(r) - C(r; \Theta, t_f)) \right|\right|_2^2 \right] \tag{2.7}$$

např. je použita pouze jedna množina parametrů neuronové sítě Θ. Ztrátová funkce pro neinformované vzorky je navíc převážena hyperparametrem λ, který autoři experimentálně stanovili na $\lambda = 0{,}1$. Na rozdíl od NeRF, kde první neuronová síť dostane 64 neinformovaných vzorků a druhá seřazených 64 neinformovaných a 128 informovaných vzorků, dostane Mip-NeRF nejprve 128 neinformovaných vzorků a následně 128 informovaných vzorků.

Implementace Mip-NeRF je založena na JaxNeRF [41], což je reimplementace NeRF, s použitím knihovny JAX [42]. JAX je designován jako HPC framework pro numerické výpočty a strojové učení.

2.2.4 Mip-NeRF 360

Mip-NeRF 360 parametrizuje scénu tak, aby bylo objektům v popředí přiznáno více kapacity modelu a objektům v pozadí kapacity méně, tedy aby přidělená kapacita modelu odpovídala disparitě bodu. Již např. NeRF++ (sekce 2.2.2) zavedl parametrizaci souřadnic bodů inverzní koulí. Mip-NeRF 360 však namísto souřadnic bodů musí reparametrizovat gaussiány. Nejprve je zavedena transformace souřadnic $f(x) : \mathbb{R}^3 \to \mathbb{R}^3$ a její lineární aproximace

$$f(x) \approx f(\mu) + J_f(\mu)(x - \mu), \tag{2.8}$$

kde $J_f(\mu)$ je Jakobián funkce f v bodě μ. Na gaussián popsaný středem μ a kovarianční maticí σ je tato transformace aplikována jako

$$f(\mu, \Sigma) = \left(f(\mu), J_f(\mu) \Sigma J_f(\mu)^\top \right). \tag{2.9}$$

Zároveň Mip-NeRF 360 upravuje způsob neinformovaného vzorkování a adaptuje ho na neohraničné scény. Namísto vzorkování v pravidelných intervalech přes vzdálenost, vzorkuje v pravidelných intervalech přes disparitu. Pro tento účel zavádí invertibilní mapování mezi euklidovskou vzdáleností bodu t a normalizovanou vzdáleností bodu s

$$s \triangleq \frac{g(t) - g(t_n)}{g(t_f) - g(t_n)}, t \triangleq g^{-1}\left(s \cdot g(t_f) + (1-s) \cdot g(t_n)\right), \tag{2.10}$$

kde $g(\cdot)$ je invertibilní funkce. Toto mapování vrací normalizované vzdálenosti $s \in [0;1]$, které lze namapovat do intervalu $[t_n; t_f]$, kdy t_n je vzdálenost roviny určující počátek scény (blíž kameře) a t_f je vzdálenost roviny určující konec scény (dál od kamery). Dosazením $g(x) = \frac{1}{x}$ určujeme transformaci ze vzdálenosti na disparitu.

Původní NeRF modeloval zejména scény s jedním objektem v popředí a maskovaným bezstrukturovým pozadím, popř. scény, jejichž záběry směřovaly vždy jen jedním směrem (forward-facing scenes). Naproti tomu 360° neohraničené scény nesou mnoho detailů na objektech v popředí i v pozadí a kapacitu NeRF je proto potřeba navýšit zvýšením poču neuronů ve skrytých vrstvách. Pro uspokojivou rekonstrukci je také potřeba až několikanásobek trénovacích snímků. To vše vedlo při některých experimentech na složitějších scénách až ke 40násobnému prodloužení trénovacícho času NeRF. Inferenci dále zpomaluje dvojité (informované a neinformované) vzorkování použité v NeRF i Mip-NeRF. Namísto trénování jedné sítě s použitím vzorků různého rozlišení Mip-NeRF 360 zavádí dvě separátní sítě: návrhovou síť a NeRF síť. Návrhová síť predikuje hustotu příslušného objemu τ. Tato hustota je transformována na váhy $\hat{w}$ podle rovnice

$$\hat{w} = \left(1 - \mathrm{e}^{-\tau_i(t_{i+1}-t_i)} e^{-\sum_{i'<i} \tau_{i'}\left(t_{i'+1}-t_{i'}\right)}\right), \tag{2.11}$$

které jsou použity pro výběr nových intervalů, které jsou poslány do NeRF sítě. Ta vygeneruje své vlastní váhy w a barvu c, ze které je pak vykreslen výsledný obrázek. Návrhová síť je relativně malá a přitom velmi často používaná, zatímco velká NeRF síť je použita méně často. Výsledkem je násobné zvýšení kapacity modelu (15×) a výrazně menší prodloužení trénovacího času (2×). Pouze NeRF síť se snaží o reprodukci barevných pixelů v obrázku, návrhová síť je optimalizována tak, aby omezila obor hodnot vah w generovaných NeRF sítí po neinformovaném vzorkování. Predikční schopnost NeRF sítě je tedy destilována do návrhové sítě. Rozdíl mezi jednou sítí v Mip-NeRF a dvěma sítěmi v Mip-NeRF 360 ilustruje obrázek 2.5.

Optimalizace návrhové sítě vyžaduje novou ztrátovou funkci, která se snaží o konzistenci histogramu $\left(\hat{t}, \hat{w}\right)$ produkovaného návrhovou sítí a histogramu (t, w) produkovaného NeRF sítí, přičemž $\hat{t}, t$ jsou vzdálenosti a $\hat{w}, w$ jsou odvozené váhy. Biny těchto histogramů však nemusí být stejné. Ztrátová funkce je tedy založena na myšlence, že její hodnota je nulová, pokud oba histogramy vychází z jediné distribuce objemu ve scéně, a nenulová, vychází-li ze dvou různých distribucí. Nejprve je definována funkce, která sčítá všechny váhy návrhové sítě, které se nacházejí v intervalu T

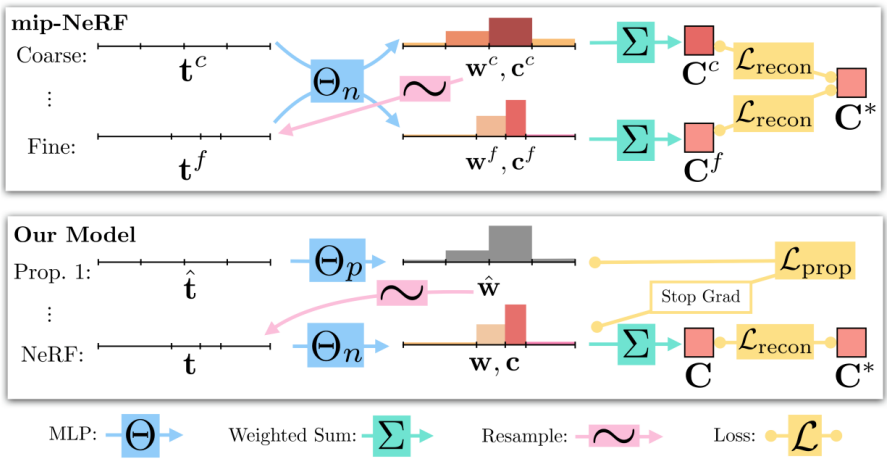
$$\text{bound}\left(\hat{t}, \hat{w}, T\right) = \sum_{j: T \cap \hat{T}_j \neq \emptyset} \hat{w}_j. \tag{2.12}$$

Jsou-li dva histogramy navzájem konzistentní, musí platit

$$\forall\left(T_i, w_i\right) \in (t, w) : \; w_i \leq \text{bound}\left(\hat{t}, \hat{w}, T_i\right). \tag{2.13}$$

Ztrátová funkce penalizuje část histogramu, která porušuje tuto nerovnost, kdy hodnoty příslušných binů histogramu převyšují hodnotu funkce bound:

$$\mathcal{L}_{\text{prop}}\left(t, w, \hat{t}, \hat{w}\right) = \sum_i \frac{1}{w_i} \max\left(0, w_i - \text{bound}\left(\hat{t}, \hat{w}, T_i\right)\right)^2 \tag{2.14}$$



Obrázek 2.5 Porovnání Mip-NeRF a Mip-NeRF 360. Mip-NeRF používá jednu síť s parametry Θ_n pro získání hustot objemu (resp. vah) w_c, w_f a barev c_c, c_f. Na ty je aplikován volume rendering (rovnice 2.4), jsou získány finální pixelové barvy C^c a C^f, které jsou zkombinovány do výsledné barvy C^*. Obě sítě jsou optimalizovány rekonstrukční ztrátovou funkcí (rovnice 2.7). Naproti tomu Mip-NeRF 360 používá návrhovou síť s parametry Θ_p pro získání odhadu hustoty objemu, resp. váhy $\hat{w}$ a NeRF síť s parametry Θ_n pro získání hustoty objemu w a barvy c, na které je aplikován volume rendering. Návrhová síť je optiamlizována samostatně a pouze NeRF síť je optimalizována rekonstrukční ztrátovou funkcí. Zdroj obrázku [8].

Ačkoliv se při trénování NeRF používá mnoho trénovacích obrázků, problém generování nových realistických pohledů je stále velmi poddefinovaný. Většina architektur NeRF zvládne rekonstruovat trénovací snímky, ale rozumné pohledy z nových úhlů zvládou rekonstruovat jen některé. Původní NeRF používal regularizaci v podobě přičítání gaussovského šumu do hlavy predikující hustotu objemu. To sice zredukovalo některé nevysvětlitelné artefakty (obláčky) ve vygenerovaných snímcích, nicméně pro komplexní neohraničené scény to není dostačující.

V takových scénách totiž kromě výskytu obláčků může dojít k tzv. kolapsu pozadí, kdy je pozadí scény vymodelováno poloprůhlednými obláčky objemu nacházejícího se nikoliv na pozadí, ale v popředí scény blízko kamery. Proto Mip-NeRF 360 navrhuje nový způsob regularizace jako ztrátovou funkci

$$\begin{aligned}\mathcal{L}_{\text{dist}}(s, w) &= \iint_{-\infty}^{\infty} w_s(u) w_s(v) - |u - v| d_u d_v \\ &= \sum_{i,j} w_i w_j \left| \frac{s_i + s_{i+1}}{2} - \frac{s_j + s_{j+1}}{2} \right| + \frac{1}{3} \sum_i w_i^2 (s_{i+1} - s_i)\end{aligned} \tag{2.15}$$

kdy $w_s(u) = \sum_i w_i \mathbb{1}_{[s_i, s_{i+1}]}(u)$. Tato funkce je integrálem vzdáleností mezi všemi dvojicemi bodů podél příslušného paprsku naškálovanými vahami w, které vrací NeRF síť. Tato ztrátová funkce je minimalizována položením $w = 0$. Pokud toto není možné, je minimalizována konsolidací vah do co nejmenší oblasti. První člen funkce minimalizuje vážené vzdálenosti mezi všemi dvojicemi středů intervalů, druhý člen minimalizuje váženou velikost každého intervalu.

Kombinace ztrátové funkce rekonstrukce obrázku $\mathcal{L}_{\text{recon}}$ (viz Obrázek 2.5), ztrátové funkce návrhové sítě $\mathcal{L}_{\text{prop}}$ a regularizace $\mathcal{L}_{\text{dist}}$ dává výslednou ztrátovou funkci

$$\mathcal{L} = \mathcal{L}_{\text{recon}}(C(t), \hat{C}) + \lambda \mathcal{L}_{\text{dist}}(s, w) + \sum_{k=0}^{1} \mathcal{L}_{\text{prop}}\left(s, w, \hat{s}_k, \hat{w}_k\right) \tag{2.16}$$

zprůměrovanou přes všechny paprsky v trénovacím batchi. Parametr λ byl stanoven na 0,01. Návrhová síť obsahuje 4 vrstvy s 256 neurony a NeRF síť obsahuje 8 vrstev s 1024 neurony. Nejprve návrhová síť pracuje s dvěma množinami 64 vzorků a vyprodukuje dvě množiny dvojic $\{(\hat{s}_0, \hat{w}_0)\}$ a $\{(\hat{s}_1, \hat{w}_1)\}$. Následně je vybráno 32 vzorků pro NeRF síť, která vrátí množinu dvojic $\{(s, w)\}$.

2.2.5 Zip-NeRF

Problém aliasingu Instant NGP deskriptorů řeší Zip-NeRF zavedením nového způsobu antialiasingu. Převádí komolé kužely použité v Mip-NeRF 360 na množinu izotropických gaussiánů (gaussiánů se zjednodušenou kovarianční maticí $\Sigma = \sigma^2 I$). Nejprve je anisotropický komolý kužel převeden na body aproximující jeho tvar s tím, že každý takový bod je uvažován jako izotropický gaussián se směrodatnou odchylkou $\sigma_j = \frac{\dot{r}t_j}{\sqrt{2}}$. Body aproximující komolý kužel s poloměrem rt, kde t je vzdálenost od počátku paprsku, jsou vybrány pomocí multisamplingu [43]. Ten produkuje množinu vzorků uspořádaných do šestiúhelníku. Vzorky svírají se středem kužele úhly $\theta = \left[0, \frac{2\pi}{3}, \frac{4\pi}{3}, \frac{3\pi}{3}, \frac{5\pi}{3}, \frac{\pi}{3}\right]$. Tyto body x_j pak tvoří středy izotropických gaussiánů se standardní odchylkou $\sigma_j = 0{,}5 \cdot \frac{rt_j}{\sqrt{2}}$. Standardní odchylka σ_j je použita pro potlačení vysokých frekvencí vzorkované funkce. Izotropické gaussiány jsou převáženy koeficientem nepřímo úměrným tomu, nakolik příslušný gaussián pasuje do příslušného políčka 3D mřížky z Instant NGP. Je-li gaussián mnohem větší než políčko, které má aproximovat, je pravděpodobně nespolehlivý a jeho váha by měla být snížena koeficientem

$$\omega_{j,l} = \operatorname{erf}\left(\frac{1}{\sqrt{8\sigma_j^2 n_l^2}}\right), \omega_{j,l} \in [0, 1], \tag{2.17}$$

kde n reprezentuje počet uzlů 3D mřížky v každé dimenzi a

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \mathrm{d}t \tag{2.18}$$

je Gaussova chybová funkce.

Dále je potřeba vyřešit možnost ztráty části obsahu scény při pohybu kamery, tzv. z-aliasing. Jeho příčinou je ztrátová funkce návrhové sítě (rovnice 2.14), která vrátí stejnou hodnotu pro částečný i úplný překryv binů histogramu. Penalizuje pouze případ, kdy se nepřekrývají vůbec. Proto Zip-NeRF tuto ztrátovou funkci nahrazuje funkcí, která je spojitá vzhledem ke vzdálenosti od obrazové roviny. Detailně ji popisuje Barron a kol. v [9], sekci 3.

2.2.6 CamP

CamP (Camera Preconditioning for Neural Radiance Fields) [44] je technika, která umožňuje společně s NeRF optimalizovat i parametry kamery. Snaží se tím zpřesnit odhady pozic kamer získané např. pomocí COLMAP. Nejprve zkoumá, jak tento problém ovlivňují různé parametrizace kamer. Zjišťuje, že u nejrozšířenějších typů parametrizací je úloha jejich optimalizace špatně

podmíněná, tedy malé změny vstupních parametrů mají velký vliv. Proto navrhuje výpočet transformace, která eliminuje korelace mezi jednotlivými parametry a normalizuje jejich efekt. Tuto funkci používá pro zlepšení podmíněnosti celého optimalizačního problému. Funkce je zapsána jako $k \times k$ matice, kde k je počet parametrů kamery. Tato matice je aplikována na parametry kamery předtím, než jsou předány modelu NeRF (autoři článku použili Zip-NeRF).

Článek rozšiřuje optimalizaci NeRF definovanou v sekci 2.2.1 o parametry kamery $\Phi_i \in \mathbb{R}^k$, kde k je počet parametrů kamery. Ztrátovou funkci popisuje jako $\mathcal{L}(D, \Theta, \Phi)$, kde Φ jsou parametry kamer. Linearizovanou optimalizaci parametrů kamer definuje jako $\Phi = \Phi^0 + \Delta\Phi$, kde je optimalizováno reziduum parametrů kamer $\Delta\Phi$ vzhledem k počátečním hodnotám Φ^0. Parametry kamer definují projekci Π, kde $\Pi(x, \Phi) : \mathbb{R}^3 \to \mathbb{R}^2$, která promítá 3D bod x do 2D pozice pixelu p.

Změna podmíněnosti je definována následovně: Buď $u \in \mathbb{R}^k$ parametr optimalizovaný tak, aby byla minimalizována ztrátová funkce $f(u) : \mathbb{R}^n \to \mathbb{R}$. Systém je přepodmíněn výpočtem matice pro změnu podmíněnosti $P \in \mathbb{R}^{k \times k}$, výpočtem $v = Pu$ a minimalizací ztrátové funkce $f(P^{-1}v)$. Po nalezení optimálního řešení v^* je odvozeno optimální řešení u^* jako $u^* = P^{-1}v^\star$. Změna podmíněnosti tohoto optimalizačního problému je definována jako změna citlivosti projekční funkce $\Pi(x; \Phi)$ na její vstupní parametry. Buď $\Pi^n(\Phi) : \mathbb{R}^k \to \mathbb{R}^{2m}$ rozšířená projekční funkci jako posloupnost funkcí Π v bodech $\{x_j\}_{j=1}^m$. Je zkoumáno, jak se projekce těchto bodů změní vzhledem k jednotlivým parametrům kamery. V libovolném bodě Φ^0 v prostoru parametrů kamery lze vliv každého parametru na každý projektovaný bod $p_j = \Pi(x_j; \Phi)$ zapsat jako jakobián

$$\left.\frac{\mathrm{d}\Pi^m}{\mathrm{d}\Phi}\right|_{\Phi=\Phi^0} = \mathrm{J}_\Pi \in \mathbb{R}^{2m \times k}. \tag{2.19}$$

rs. člen matice $\Sigma_\Pi = J_\Pi^\top J_\Pi \in \mathbb{R}^{k \times k}$ je roven

$$\sum_{l=1}^{m} \left(\frac{\mathrm{d}p_l}{\mathrm{d}\Pi[r]}\right)^\top \left(\frac{\mathrm{d}p_l}{\mathrm{d}\Pi[s]}\right). \tag{2.20}$$

Na diagonále matice Σ_Π je pak průměrný pohyb, který způsobí změna r. parametru $\Phi[r]$ a mimo diagonálu pak korelace těchto pohybů mezi parametry $\Phi[r]$ a $\Phi[s]$. Cílem je nalézt matici pro změnu podmíněnosti P takovou, že dosazením $P^{-1}\tilde{\Phi} = \Phi$ do projekční funkce $\Phi^m(P^{-1}\tilde{\Phi}) = \tilde{\Pi}^m(\tilde{\Phi})$ je získána matice $\Sigma_{\tilde{\Pi}} = \mathrm{J}_{\tilde{\Pi}}^\top \mathrm{J}_{\tilde{\Pi}}$, která je rovna jednotkové matici I_k, přičemž $\mathrm{J}_{\tilde{\Pi}} = \mathrm{J}_\Pi P^{-1}$. Tento problém má nekonečně mnoho řešení, přičemž autoři zvolili

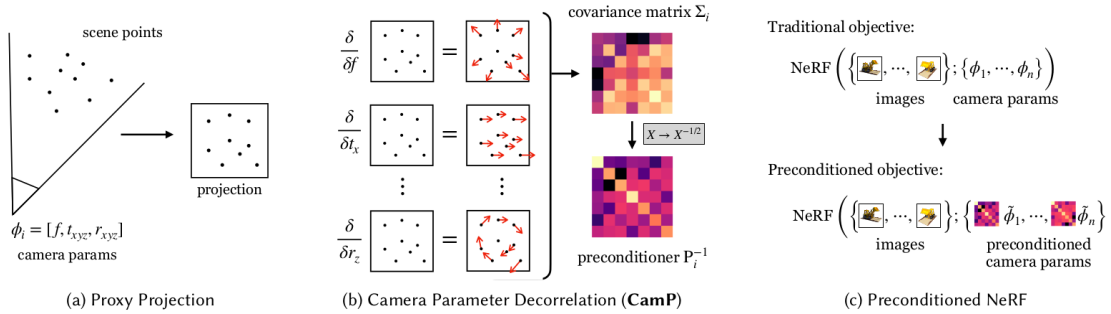
$$P^{-1} = \Sigma_\Pi^{-\frac{1}{2}} = \left(\mathrm{J}_\Pi^\top \mathrm{J}_\Pi\right)^{-\frac{1}{2}}. \tag{2.21}$$

Celá metoda je ilustrována na Obrázku 2.6.

Navržený postup předpokládal, že každý snímek má svou vlastní kameru a ty jsou optimalizovány nezávisle. V mnohých případech je však k získání všech snímků v datasetu použita jediná kamera (a objektiv), proto jsou vnitřní parametry sdílené. CamP toto řeší rozšířením ztrátové funkce o člen $\mathcal{L}_{\text{shared}}$, který minimalizuje rozptyl těchto sdílených parametrů. Tento člen je do funkce přidán v podobě malých hodnot přičtených k diagonále kovarianční matice Σ_Π.

2.3 Gaussian Splatting

Na rozdíl od klasického NeRF, který optimalizuje scénu jako funkci v pětidimenizionálním prostoru, Gaussian Splatting reprezentuje zářivá pole ve scéně jako množinu 3D gaussiánů. Metoda



(a) Proxy Projection (b) Camera Parameter Decorrelation (**CamP**) (c) Preconditioned NeRF

Obrázek 2.6 Na obrázku a) je ilustrován výběr množiny bodů $x_j{}_{j=1}^{m}$ viditelných příslušnou kamerou. Na těchto bodech je na obrázku b) vypočítána kovarianční matice Σ_i (rovnice 2.20) derivací projekcí vzhledem k jednotlivým parametrům kamery. Z té je pak spočítána matice pro změnu podmíněnosti P_i^{-1}. Obrázek c) ilustruje reformulaci optimalizačního problému z přímé optimalizace parametrů kamery $\{\Phi_i\}$ na optimalizaci latentních parametrů $\{\tilde{\Phi}\}_i$. Při výpočtu ztrátové funkce jsou latentní parametry převedeny na skutečné parametry $\Phi_i = P^{-1}\tilde{\Phi}_i$ a ty jsou pak předány NeRF. Zdroj obrázku [44].

na vstupu přijímá kamery kalibrované pomocí COLMAP SfM [36], konkrétně sparse point cloud, který SfM produkuje, a kterým metoda inicializuje gaussovskou reprezentaci scény, konkrétně pozice středů, kovarianční matice a průhlednost α. Barva, jejíž hodnota závisí na směru pohledu na scénu, je reprezentována sféricko harmonickými (SH) koeficienty [45].

3D gaussián $G(x)$ je definován středem μ a 3D kovarianční maticí Σ (obojí ve světovém souřadnicovém systému)

$$G(x) = \exp\left(-\frac{1}{2}(x)^\top \Sigma^{-1}(x)\right). \tag{2.22}$$

Takto definované gaussiány jsou pak násobeny koeficientem průhlednosti α, resp. neprůhlednosti, jelikož $\alpha = 0$ znamená zcela průhledný a $\alpha = 1$ neprůhledný gaussián. Pro účely vykreslení výsledného obrázku je však nutné provést projekci 3D gaussiánu do dvoudimenzionální obrazové roviny. Toho lze docílit s použitím transformační matice ze světového do kamerového souřadnicového systému W, pro převedení kovarianční matice gaussiánu do tohoto souřadnicového systému pomocí vztahu $\Sigma' = JW\Sigma W^\top J^\top$, kdy J je jakobián afinní aproximace příslušné projektivní transformace. Vynecháním třetího řádku a sloupce matice Σ' dostaneme 2×2 kovarianční matici pro rovinu. Prostá optimalizace kovarianční matice Σ pomocí gradientního sestupu tak, aby gaussiány správně reprezentovaly zářivé pole, není možná, jelikož kovarianční matice musí být pozitivně semidefinitní, což v průběhu gradientního sestupu nelze zaručit. Proto byla kovarianční matice vyjádřena jako konfigurace elipsoidu pomocí škálovací matice S a rotační matice R jako $\Sigma = RSS^\top R^\top$. Aby mohly být obě komponenty optimalizovány nezávisle, byl zvlášť uložen škálovací vektor s a rotační kvaternion q.

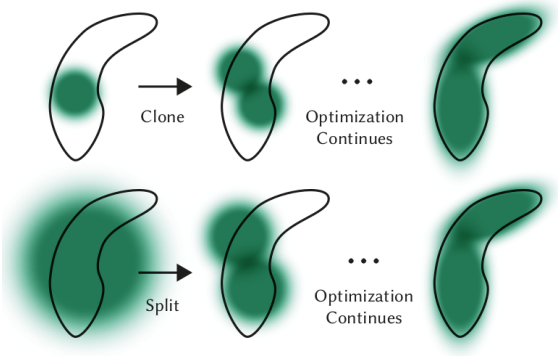
Optimalizační proces pracuje se všemi parametry 3D gaussiánů μ, Σ, α a SH koeficienty. Zároveň je optimalizována hustota gaussiánů co do jejich umístění v prostoru tak, aby dobře reprezentovaly scénu. Optimalizace je založena na iterativním procesu vykreslování a následném porovnávání vykreslených obrázků s ground-truth trénovacími snímky. Na začátku jsou gaussiány odhadnuty jako izotropické objekty s osami rovnými průměrné vzdálenosti ke třem nejbližším bodům ze SfM pointcloudu. Ztrátová funkce je definována jako

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}, \tag{2.23}$$

kdy $\mathcal{L}_1$ je po složkách absolutní hodnota rozdílu hodnot optimalizované proměnné a ground-truth, $\lambda = 0{,}2$ a

$$\mathcal{L}_{\text{SSIM}}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{2.24}$$

přičemž μ_x, μ_y jsou středy gaussiánů, σ_x^2, σ_y^2 příslušné rozptyly, σ_{xy} kovariance a c_1, c_2 konstanty zajišťující numerickou stabilitu [46]. Po inicializaci gaussiánů probíhá vždy 100 iterací jejich zahušťování, pak jsou odstraněny neužitečné průhledné gaussiány, tedy gaussiány s $\alpha < \epsilon_\alpha$. Při zahušťování je potřeba gaussiány doplnit do prázdných oblastí s chybějícími geometrickými deskriptory (podrekonstruované oblasti) a do oblastí, kde jednotlivé gaussiány zabírají příliš velkou oblast (přerekonstruované oblasti). Experimentálně bylo zjištěno, že takové gaussiány mají velké poziční gradienty, což odpovídá tomu, že tyto gaussiány reprezentují nepříliš dobře rekonstruované regiony, proto se je optimalizační proces snaží posunout a rekonstrukci tak zlepšit. Z tohoto důvodu jsou zahušťovány ty gaussiány, jejichž poziční gradient je větší než $\tau_{\text{pos}} = 0{,}0002$. Malé gaussiány v podrekonstruovaných oblastech jsou zkopírovány a tato nová kopie je posunuta ve směru pozičního gradientu. Velké gaussiány v přerekonstruovaných oblastech jsou nahrazeny dvěma menšími gaussiány zmenšenými konsantou $\phi = 1{,}6$. Jejich pozice jsou vybírány vzorkováním z původního gaussiánu jako z hustoty pravděpodobnosti. V podrekonstruovaných oblastech je tedy zvětšováno množství objemu reprezentovaného gaussiány, zatímco v přerekonstruovaných oblastech je stejné množství objemu rozděleno do více gaussiánů (viz také Obrázek 2.7). Zároveň je nutno zabránit tvorbě obláčků v blízkosti jednotlivých kamer. Proto je každých 3 000 iterací hodnota α všech gaussiánů sražena blízko 0. Následné optimalizační kroky zvyšují α pro gaussiány, které jsou pro reprezentaci scény nezbytné a zbytné gaussiány s nízkým $\alpha < \epsilon_\alpha$ jsou pak pravidelně odstraňovány.

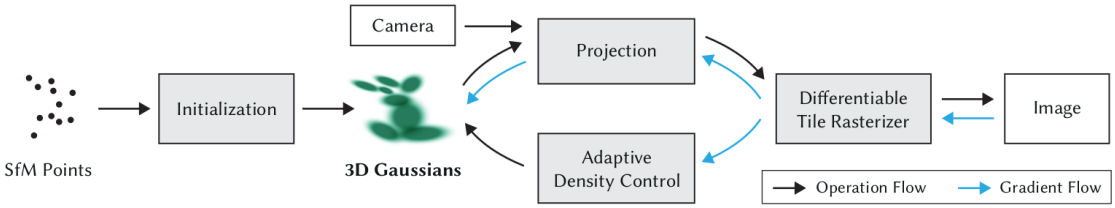


Obrázek 2.7 Schéma zahušťování gaussiánů. V horní řadě je ilustrováno zdvojnásobení objemu reprezentovaného původním gaussiánem do dvou gaussiánů a následné rozložení gaussovsky reprezentovaného objemu do tvaru skutečného objektu ve scéně. Ve spodní řadě je ilustrováno rozdělení velkého množství objemu v původním gaussiánu do dvou gaussiánů, které se pak také přizpůsobí skutečnému tvaru objektu ve scéně. Zdroj obrázku [11].

Hlavním cílem metody je umožnit vykreslování nových pohledů v reálném čase. Proto byl zaveden dlaždicovitý rasterizér gaussiánů umožňující efektivní zpětný chod gradientů přes libovolný počet spojených gaussiánů. Nejprve je scéna rozdělena na 16×16 dlaždic a následně dochází k výběru gaussiánů pro finální vizualizaci. Jsou vybrány pouze gaussiány, jejichž 99% konfidenční interval protíná příslušný komolý kužel jako výřez pohledu na scénu z příslušného políčka. Vybrané gaussiány jsou pak instanciovány tolikrát, kolik dlaždic protínají a každé instanci je přiřazen klíč, kombinující hloubku v prostoru pohledu a identifikátor příslušné dlaždice. Tyto instance jsou následně seřazeny podle těchto klíčů pomocí GPU implementace číslicového řazení [47]. Poté je

pro každou dlaždici sestaven seznam seřazených gaussiánů. Následná rasterizace je prováděna paralelně, kdy rasterizaci jedné dlaždice odpovídá jeden blok GPU vláken. Každý blok nejprve načte gaussiány do své sdílené paměti a následně pro každý pixel akumuluje barvu a průhlednost α postupným procházením seznamu. Jakmile příslušné vlákno dosáhne cílové (ne)průhlednosti $\alpha = 1$, zastaví se. Vlákna příslušející každé dlaždici jsou v pravidelných intervalech kontrolována a jakmile jsou všechny pixely saturovány, výpočet pro příslušný blok končí. Během zpětné propagace gradientu je potřeba zrekonstruovat sekvenci všech bodů (resp. gaussiánů), které přispěly k výpočtu barvy příslušného pixelu během dopředného chodu. Proto jsou znovu procházeny seznamy seřazených gaussiánů, nyní pozpátku. Koeficienty pro výpočet gradientu jsou získány jako podíl finální akumulované průhlednosti α a hodnoty α příslušného gaussiánu.

Schéma inicializace gaussiánů, jejich optimalizace a následné vykreslování je znázorněno na Obrázku 2.8.



Obrázek 2.8 Schéma Gaussian Splatting. Zdroj obrázku [11].

2.4 DSRNet – metoda pro odstraňování odlesků

DSRNet (Dual-stream Semantic-aware network with Residual Connection) [14] kromě dekompozice obrazu na základní snímek a odraz pracuje také s residuem této dekompozice, konkrétně nahlíží na snímek I jako na

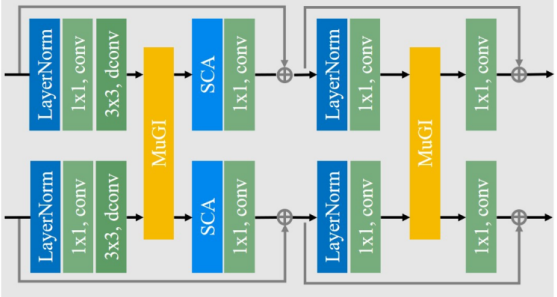
$$I = \tilde{T} + \tilde{R} = T + R + \Phi(T, R), \tag{2.25}$$

kde T a R jsou ground-truth základního snímku a odrazu, $\tilde{T} = g(T)$ a $\tilde{R} = f(R)$ jsou odleskem i jiným způsobem degradované T a R které jsou viditelné kamerou a $\Phi(T, R) = I - T - R$ je residuum této dekompozice. Jelikož funkce $\Phi(\cdot, \cdot)$ může být libovolná, pracuje s ní DSRNet jako s předmětem optimalizace. Residuum umožňuje dosáhnout kvalitnější a čistší predikce $\tilde{T}$ a $\tilde{R}$, jelikož obsáhne všechny komponenty nad rámec prosté lineární dekompozice I.

Vstupem do sítě je vstupní obrázek I a pyramida příznaků, kterou z I extrahuje VGG-19 [16]. Tyto příznaky jsou hierarchicky agregovány sítí DSFNet (Dual-Stream Pyramid Fusion Network, viz Obrázek 2.10, část DSFNet). Vstupní obrázek je zde rozkopírován na dva, tyto snímky slouží jako základ pro odhad průhledové vrstvy $\hat{T}$, resp. pro odhad odrazové vrstvy $\hat{R}$. Každá úroveň deskriptorové pyramidy je nejprve zpracována MuGI blokem (viz Obrázek 2.9). Žlutě označené části MuGI určují, které části zpracovávaných deskriptorů jsou relevantní pro průhledovou, resp. odrazovou vrstvu. Jsou implementovány jako

$$\begin{cases} \hat{F}_T = \mathcal{G}_1\left(F_T\right) \circ \mathcal{G}_2\left(F_R\right) \\ \hat{F}_R = \mathcal{G}_1\left(F_R\right) \circ \mathcal{G}_2\left(F_T\right), \end{cases} \tag{2.26}$$

kde $F_T, F_R \in \mathcal{R}^{H_1 \times W_1 \times C_1}$ představují obrazové deskriptory získané z $\tilde{T}$ a $\hat{R}$ a $\hat{F}_T, \hat{F}_R \in \mathcal{R}^{H_2 \times W_2 \times C_2}$ představují výstup bloku. $\mathcal{G}_1$ a $\mathcal{G}_2$ jsou funkce určující, která část deskriptorů je relevantní pro



Obrázek 2.9 Schéma MuGI bloku. Zdroj obrázku [14].

příslušnou vrstvu. DSRNet je implementuje tak, že $\mathcal{G}_1(\cdot)$ vezme první polovinu vrstev feature map a $\mathcal{G}_2$ druhou polovinu, tedy $H_1 = H_2$, $W_1 = W_2$ a $C_2 = \frac{C_1}{2}$.

Deskriptory zpracované MuGI blokem jsou pak přeškálovány na rozlišení odpovídající vyšší úrovni pyramidy, spojeny s deskriptory v této vrstvě a dohromady opět zpracované MuGI blokem. Zpracování pokračuje až na nejvyšší úroveň pyramidy, kde jsou deskriptory spojeny se vstupním obrázkem.

Takto zpracované deskriptory jsou postoupeny síti DSDNet (Dual-Stream Fine-Grained Decomposition Network, viz Obrázek 2.10, část DSDNet), která sestává z MuGI, downsamplovacích a upsamplovacích bloků sestavených obdobně jako architektura U-Net [48]. Dekompoziční reziduum je pak odhadnuto sítí LRM (Learnable Residue Module, viz Obrázek 2.10, část LRM).

Během trénování síť minimalizuje rekonstrukční ztrátovou funkci

$$\mathcal{L}_{\text{rec}} = \left|\left| I - \left(\hat{T} + \hat{R}\right) - \Phi\left(\hat{T}, \hat{R}\right)\right|\right|_1. \tag{2.27}$$

Kromě ní DSRNet zavádí také další ztrátové funkce: pixelovou ztrátovou funkci $\mathcal{L}_{\text{pix}}$ zajišťující konzistenci zrekonstruovaných snímků a jejich horizontálních a vertikálních gradientů vůči příslušné ground-truth

$$\mathcal{L}_{\text{pix}} = \left|\left|\hat{T} - T\right|\right|_2^2 + \left|\left|\hat{R} - R\right|\right|_2^2 + \alpha\left(\left|\left|\nabla\hat{T} - \nabla T\right|\right|_1 + \left|\left|\nabla\hat{R} - \nabla R\right|\right|_1\right), \tag{2.28}$$

percepční ztrátovou funkci zajišťující vizuální konzistenci predikcí a ground-truth a zároveň bránící přílišným penalizacím drobných odchylek ve světelnosti snímků

$$\mathcal{L}_{\text{per}} = \sum_i \omega_i \left|\left|\phi_i\left(\hat{T}\right) - \phi_i\left(T\right)\right|\right|_1, \tag{2.29}$$

kde $\phi_i(\cdot)$ označuje výstup $i.$, $i \in \{2, 7, 12, 21, 30\}$ vrstvy VGG-19 jako extraktoru deskriptorů. Parametry ω kombinují deskriptory z různých vrstev. Poslední ztrátová funkce $\mathcal{L}_{\text{exc}}$ posiluje nezávislost jednotlivých gradientů:

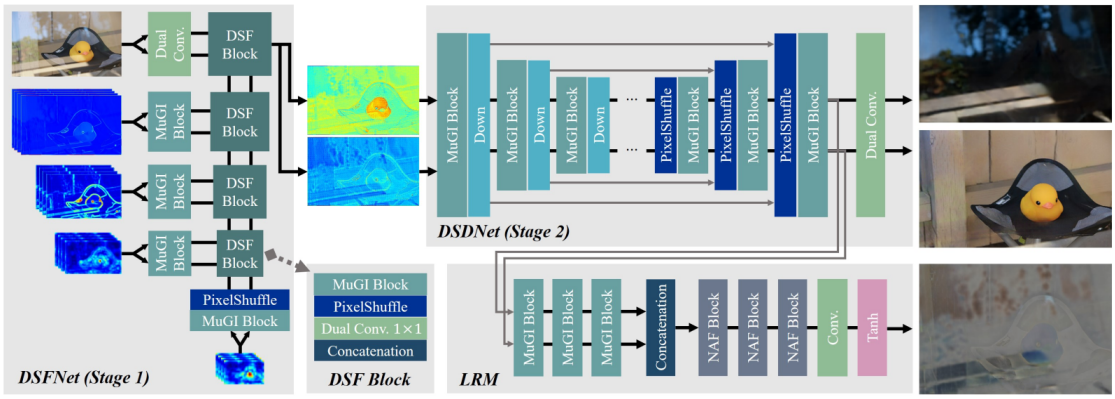
$$\mathcal{L}_{\text{exc}} = \frac{1}{N}\sum_{n=0}^{N-1}\left|\left|\Psi\left(\hat{T}^{\downarrow n}\hat{R}^{\downarrow n}\right)\right|\right|_2^2, \text{ kde } \Psi\left(\hat{T}, \hat{R}\right) = \tanh\left(\eta_1\left|\nabla\hat{T}\right|\right) \circ \tanh\left(\eta_2\left|\nabla\hat{R}\right|\right) \tag{2.30}$$

a $\hat{T}^{\downarrow n}$, $\hat{R}^{\downarrow n}$ reprezentují 2^nkrát provedené škálování snímku na menší rozlišení. η_1 a η_2 jsou normalizační faktory.

Výsledná ztrátová funkce celé sítě $\mathcal{L}_{\text{all}}$ kombinuje výše popsané ztrátové funkce takto:

$$\mathcal{L}_{\text{all}} = \mathcal{L}_{\text{pix}} + \beta_1 \mathcal{L}_{\text{per}} + \beta_2 \mathcal{L}_{\text{exc}} + \beta_3 \mathcal{L}_{\text{rec}}, \tag{2.31}$$

kdy $\beta_1 = 0{,}01$, $\beta_2 = 1$ a $\beta_3 = 0{,}2$.



Obrázek 2.10 Schéma DSRNet. Zdroj obrázku [14].

2.5 Metody pro instance segmentation

Tato sekce popisuje vybrané metody hlubokého učení pro segmentaci instancí objektů v obraze s důrazem na výpočetní rychlost a přesnost.

2.5.1 YOLOv8

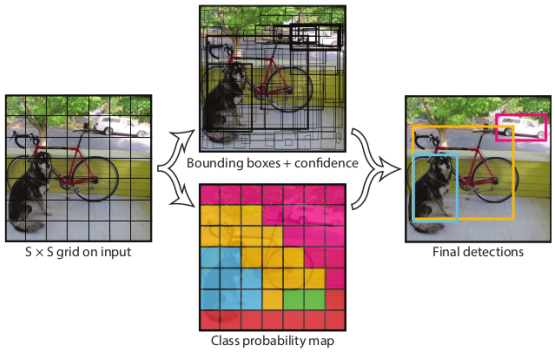
YOLO [28] je konvoluční neuronová síť určená pro detekci a klasifikaci objektů v reálném čase (až 45 snímků za sekundu). Nejprve je obrázek zpracován 24konvolučními vrstvami (extrakce příznaků) a 2MLP vrstvami (nelineární zpracování příznaků a jejich transformace do výstupního formátu). Výstupem sítě je tensor o velikosti $S \times S \times (B \cdot 5 + C)$. Snímek je totiž rozdělen na mřížku o velikosti $S \times S$. Detekovaný objekt je přiřazen příslušnému políčku mřížky tehdy, pokud se v něm nachází střed tohoto objektu. Pro každé pole v mřížce je predikováno B bounding boxů (x, y, w, h) a konfidenčního skóre, které vyjadřuje, nakolik si je model jistý polohou detekovaného objektu. Zároveň je pro každé políčko (bez ohledu na počet bounding boxů v příslušném políčku) predikována množina C podmíněných pravděpodobností $\mathrm{P}(c = c_i|\text{object})$, tedy C podmíněných pravděpodobností, že pokud se v příslušném políčku nachází nějaký objekt, patří s touto pravděpodobností do třídy c_i. Výstupní formát ilustruje Obrázek 2.11.

Na architekturu YOLO pak navázal další výzkum a postupně byly odvozeny další verze, které se zaměřovaly na další optimalizaci rychlosti, výpočetní náročnosti, na maximalizaci přesnosti, zjednodušení práce s modelem a přidání nových možností, např. instance segmentation. Velmi používané modely jsou YOLOv5 [30]a YOLOv8 [31]. I přes širokou známost těchto modelů autoři k těmto modelům nezveřejnili žádnou vědeckou publikaci, nicméně mnoho článků [49, 50] nabízí jejich shrnutí.

YOLOv5 pracuje s třífázovou architekturou, kdy se jednotlivé části nazývají páteř (backbone), krk (neck) a hlava (head). Jako backbone pro extrakci příznaků slouží architektura CSPNet [51].

Extrahované příznaky si následně převezme neck v podobě PANet (Path Aggregation Network) [52], která příznaky z jednotlivých vrstev spojuje dohromady, přičemž zachovává jejich sémantickou a prostorovou informaci. Takto spojené příznaky pak doputují do head, která provede finální detekci objektu. Implementace YOLOv5 [30] již také umožňuje provádět instance segmentation.

YOLOv8 jako backbone používá FPN (Feature Pyramid Network) [53]. Zbytek architektury je velmi podobný, došlo pouze k drobným změnám v architektuře prostřední části (necj). YOLOv5 i YOLOv8 používají mozaikovitou augmentaci dat [29], která náhodně vybírá 4 snímky z příslušného datasetu, náhodně je ořízne a spojí do výsledného trénovacího obrázku.

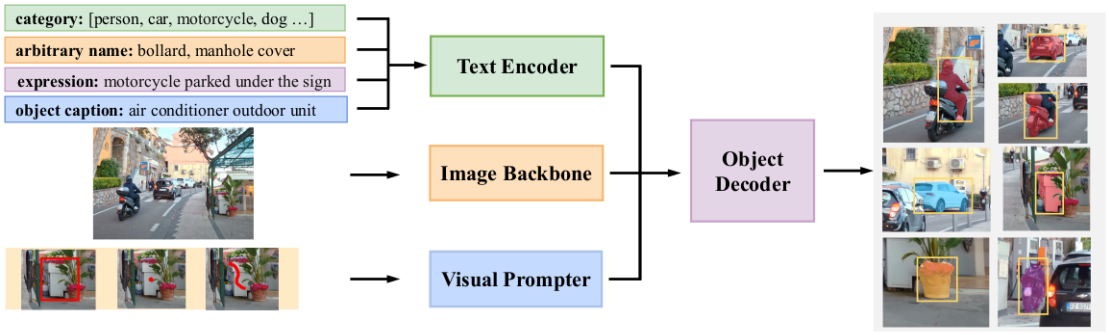


Obrázek 2.11 Schéma YOLOv1. Obrázek vlevo ukazuje rozdělení snímku na $S \times S$ mřížku. Obrázek nahoře ukazuje predikci bounding boxů a jejich konfidenčního skóre (tloušťka boxu) pro auto, kolo a psa. Obrázek dole ukazuje predikci pravděpodobností tříd v příslušném políčku mřížky. Obrázek vpravo ukazuje finální predikce (prahování bounding boxů konfidenčním prahem). Zdroj obrázku [28].

2.5.2 GLEE

GLEE [35] je model pro detekci, segmentaci a trackování libovolných objektů ve scéně. Model byl trénován na 5milionech obrázků z datasetů používaných pro různé benchmarky. Použity byly datasety s anotacemi bounding boxů a názvů tříd (Objects365 [54]), případně masek a názvů tříd (COCO [22]) nebo datasety poskytující popis obrázku formou volného textu, např. Visual Genome [55]. Díky tomu je model schopen velmi generické reprezentace sledovaných objektů, což mu umožňuje pracovat s novými daty nebo adaptaci na úkol, který se v trénovacích datech nevyskytoval, bez nutnosti dodatečného trénování (zero-shot transfer).

Model sestává z textového enkodéru, který kóduje popis vstupních dat, popř. úkolu, který má model vykonat (např. „největší auto ve snímku“), dále backbone sítě pro extrakci obrazových příznaků a vizuálního prompteru (vizuální prompt je označení objektu, který chceme detekovat, popř. segmentovat nebo trackovat, viz také Obrázek 2.12). Tyto vstupy jsou integrovány v dekodéru založeném na MaskDINO [56], který extrahuje příslušné objekty ze scény. Jeho výstupní emgeddingy zpracovávají tři detekční hlavy, jedna pro masky, druhá pro bounding boxy a třetí pro třídy objektů.



Obrázek 2.12 Schéma GLEE. Zdroj obrázku [35].

Kapitola 3

Analýza

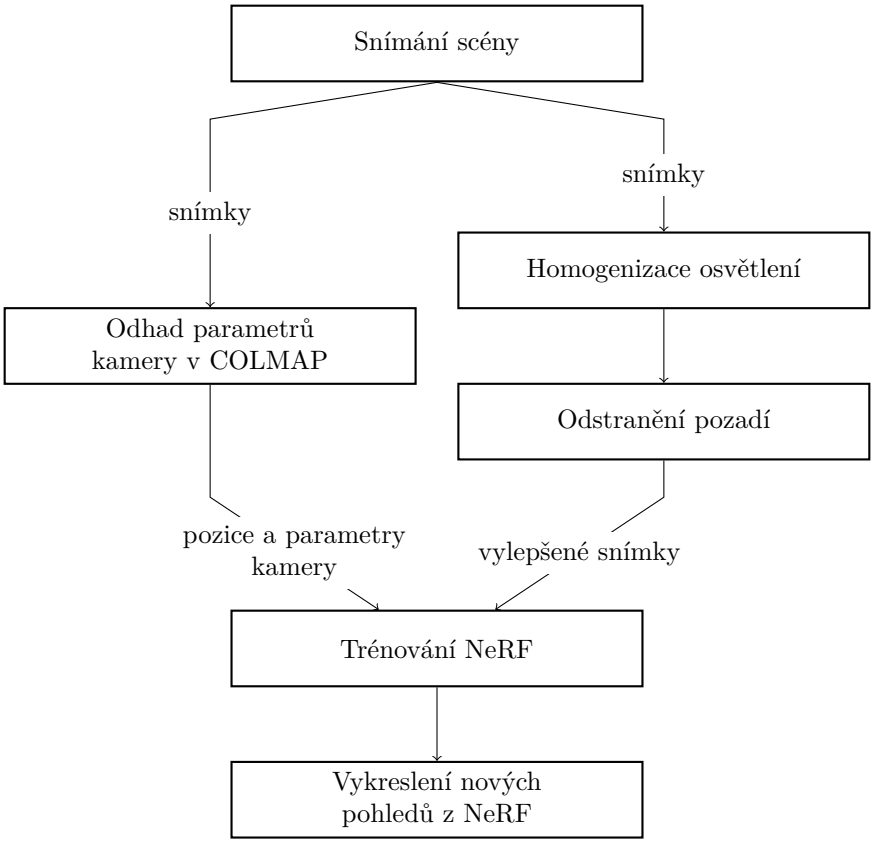
Práce navazuje na výsledky předchozí diplomové práce [57], která vytvořila prototyp řešení pro modelování a vykreslování automobilů pomocí NeRF na základě pořízených snímků, kdy byla nejprve provedena homogenizace osvětlení ve snímcích, následně byly automobily segmentovány a nakonec modelovány pomocí NeRF. Tato práce navrhuje vylepšení některých původních komponent, například zavádí nový model pro segmentaci vozu a nahrazuje původní model NeRF v Instant NGP třemi pokročilejšími modely.

3.1 Existující řešení

Původní práce [57] zkoumala metodiku snímání vozu. Bylo zjištěno, že vozidlo je vhodné nasnímat dokola v několika výškových úrovních, přičemž optimální počet snímků pro jednu výškovou úroveň je 20 – 25. Zároveň bylo zjištěno, že je vhodné mít tři úrovně snímání, nejnižší v úrovni kol a nejvyšší nad úrovní střechy vozidla. Nejvyšší vrstva je však potřebná pouze pro generování pohledů na automobil zvrchu, což je však velmi neobvyklé, jelikož člověk běžné výšky si auta zvrchu neprohlíží, tedy ani prodejní inzeráty těmito snímky nedisponují.

Pro odhad pozic kamery byl použit algoritmus COLMAP. Práce zmiňuje možnosti použití lepšího algoritmu pro extrakci příznaků, jelikož hladká textura metalízy vozidla je nevhodným povrchem pro hledání robustních deskriptorů. COLMAP tak v mnohých datasetech nacházel referenční body v oblasti kol a SPZ. Pokud scéna neměla dostatečně heterogenní pozadí, nebyly pozice kamer odhadnuty správně, což vedlo např. ke splynutí přední a zadní části auta a NeRF pak generoval snímky, kde přední či zadní pohledy obsahovaly prvky přídě i zádě vozidla najednou.

Práce analyzovala mnohé algoritmy pro homogenizaci osvětlení. Pro porovnání použila NR-IQA (no-reference image quality assessment) metriky [58], ze kterých však nebylo možné jednoznačně určit, která z testovaných metod vedla ke zlepšení vizuální kvality obrazu. Jako závěr bylo stanoveno, že vzhledem k výpočetní náročnosti a nejednoznačným výsledkům je vhodné zvolit metodu ekvalizace histogramu, která nemá vysoké výpočetní nároky. Chybí však podrobnější diskuze výsledků této operace na reálných datech, práce poskytuje pouze jeden snímek, ze kterého je patrné, že přesvětlené oblasti jsou sice tmavší, avšak dříve méně osvětlené oblasti jsou příliš tmavé.



Obrázek 3.1 Schéma systému navrženého v původní implementaci [57].

Práce testovala dva algoritmy pro odstranění pozadí – konvoluční neuronové sítě architektury U2-Net [59] a model SAM (Segment Anything Model) [60]. U2-Net determinuje objekt k segmentaci automaticky, SAM je sice schopen téhož, pro zajištění spolehlivějších detekcí mu však autor ještě předložil bounding box největšího automobilu nalezeného modelem YOLO.

Pro modelování vozu byl použit fotogrammetrický software Meshroom [61] a Instant NGP [10], což je implementace NeRF optimalizovaná pro GPU. Takto byl NeRF trénován na původních snímcích (automobil i pozadí) se vzorkováním celého prostoru scény, dále na původních snímcích, avšak s omezením vzorkování prostoru, byla vzorkována pouze oblast, kde se nacházel automobil, což v důsledku znamenalo odstranění pozadí a nakonec na snímcích s vysegmentovaným automobilem a odstraněným pozadím. Výsledky prokázaly nevhodnost fotogrammetrického softwaru, který si nebyl schopen poradit s průhlednými částmi vozu, zejména s čelním sklem. NeRF fungoval nejlépe pro segmentovaná data z důvodu menší kapacity modelu a absence adaptace na neohraničené scény (viz sekce 2.2.2, 2.2.3).

3.2 Návrhy vylepšení

Vzhledem k popsaným nedostatkům algoritmu COLMAP by bylo vhodné otestovat alternativní algoritmy pro zisk obrazových příznaků, případně COLMAP inicializovat apriorní znalostí pozic kamer získaných z 3D poloh snímače v příslušném okamžiku. Takto získané polohy by následně byly pouze zpřesňovány triangulací deskriptorů získaných pomocí COLMAP. Tyto experimenty však nejsou součástí této práce.

Je velmi obtížné zavést metriku pro vyhodnocení homogenizace osvětlení, popř. jiných algoritmů zpracování obrazu aplikovaných na použité snímky. Jeden z důvodů je, že ani není možné dobře formulovat, jak by měl výsledek vypadat, a co pro to udělat. Proto bylo upuštěno od dalšího zkoumání a vyhodnocování metod homogenizace osvětlení a namísto toho byly zkoumány algoritmy pro odstranění odlesků. Odlesky způsobené přímým slunečním svitem na lesklou metalízu, popř. odlesky okolního prostředí jsou něco, co ruší model automobilu, který by měl obsahovat pouze automobil samotný bez vlivů okolního prostředí. Odlesk samotný je dobře definovaný a zároveň je dobře definovaná podoba automobilu bez vlivu příslušného odlesku. Tato úloha je taktéž předmětem aktivního výzkumu (viz sekce 1.2) a existují také použitelné implementace (sekce 2.4).

Práce zmiňuje, že segmentace pomocí U2-Net není zcela spolehlivá, model totiž neví, co je nejdůležitější objekt ve scéně, a co má být tedy segmentováno. U2-Net je tedy vhodný pouze na některé exteriérové scény, problematická jsou např. další auta ve scéně, popř. jiné prominentní objekty, které by U2-Net mohl vysegmentovat také, případně auto zcela ignorovat a namísto něj segmentovat onen prominentní objekt. Experimentálně bylo zjištěno (viz sekce 5.4), že ani SAM není zcela spolehlivý, zejména v případě, že selže detekční model, který segmentačnímu modelu dává bounding box vozidla k segmentaci. YOLO je totiž optimalizováno zejména na rychlost, což nevyhnutelně vede k jistým kompromisům v přesnosti. To se experimentálně projevilo tak, že ani nejnovější YOLOv8 nebylo schopno vozidlo detekovat při pohledu z některých úhlů, nebo bylo-li na automobilu zvlášť velké množství odlesků. Proto byly v této práci využity pokročilejší modely pro instance segmentation, byť za cenu jistého snížení rychlosti. Rozdíl však na jeden snímek činí maximálně stovky milisekund, což je vzhledem k celkové výpočetní náročnosti všech komponent dohromady, která činí střední desítky minut až nižší desítky hodin v závislosti na použitém modelu pro vykreslování nových pohledů, zanedbatelné.

Původní práce použila pouze základní architekturu NeRF a zcela ignorovala další výzkum, který v této oblasti proběhl, zejména adaptaci modelů na neohraničené scény zabrané z 360° (sekce 2.2.2 a 2.2.3) nebo alternativní reprezentace objemu ve scéně (sekce 2.3), což umožňuje dramatické

zrychlení trénování i vykreslování nových pohledů, přičemž je stále možné reprezentovat složitou neohraničenou scénu. Proto byly testovány pokročilé architektury NeRF.

3.3 Datasety a metodika snímání

Byly pořízeny tři typy datasetů:

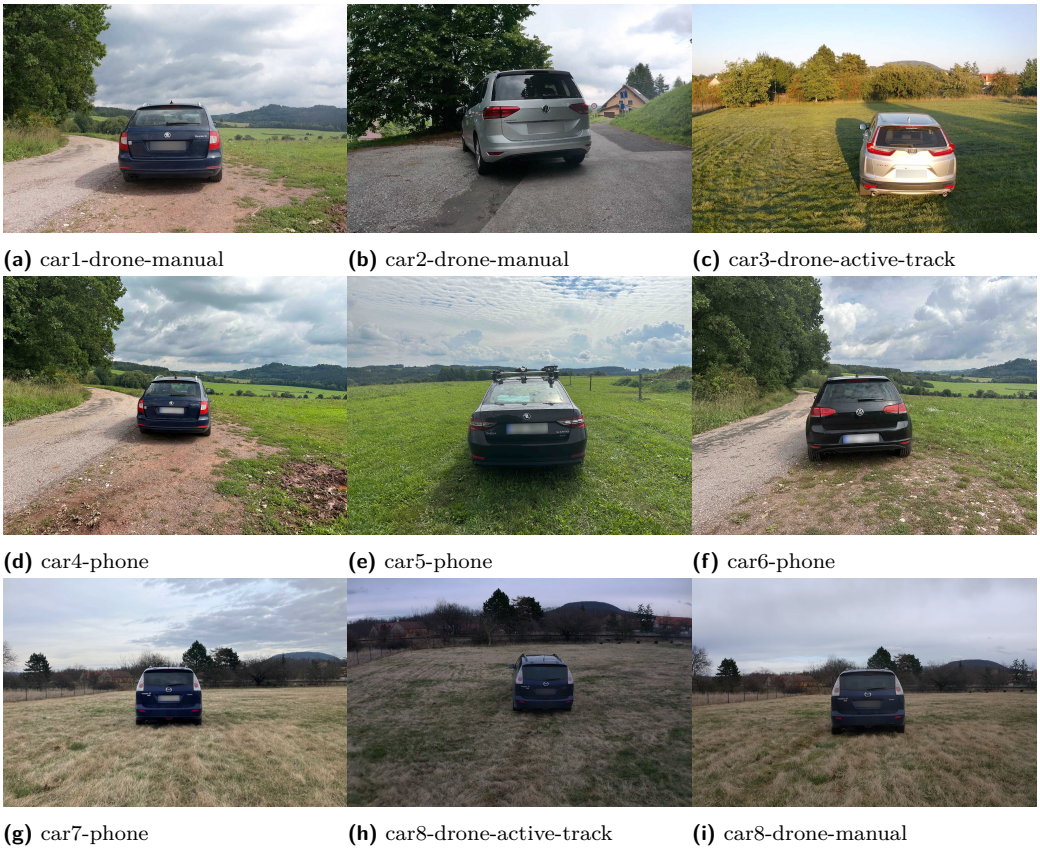
- reálný dataset pořízený dronem,
- reálný dataset pořízený mobilním telefonem,
- syntetický dataset vytvořený v Blenderu [62].

Všechny datasety byly pořízeny jako 360° záběry vozidla, snímač kolem objektu opsal jeden, nebo dva kruhy.

Seznam datasetů:

- **car1-drone-manual**: dataset byl pořízen manuálně ovládaným dronem Mavic 2 [63] a obsahuje 1 oblet o 54 snímcích. Ukázka viz Obrázek 3.2a.
- **car2-drone-manual**: dataset byl pořízen manuálně dronem Mavic 2 a obsahuje 1 oblet o 93 snímcích. Ukázka viz Obrázek 3.2b.
- **car3-drone-active-track**: dataset byl pořízen dronem Mavic 2. Dron byl řízen automaticky s použitím funkce ActiveTrack [64]. Tato funkce umožňuje sledování objektu (i pohyblivého). Nejprve je nutné tento objekt manuálně vybrat v mobilní aplikaci, která s dronem interaguje, dron pak objekt automaticky sleduje z bezpečné vzdálenosti, umí jej také oblétat dokola. Pro účely datasetu byl zvolen oblet dokola za dodržení fixní vzdálenosti od objektu. Dataset obsahuje 1 oblet o 35 snímcích. ActiveTrack sice dodržel fixní vzdálenost, nicméně jako oblétávaný objekt nebyl detekován střed automobilu, nýbrž jeho levý bok. Proto je automobil na snímcích z jedné strany výrazně blíže. Ukázka viz Obrázek 3.2c.
- **car4-phone**: dataset byl pořízen mobilním telefonem a obsahuje 1 neúplný kruh o 28 snímcích. Ukázka viz Obrázek 3.2d.
- **car5-phone**: dataset byl pořízen mobilním telefonem a obsahuje 2 neúplné kruhy o 31, resp. 32 snímcích. Ukázka viz Obrázek 3.2e.
- **car6-phone**: dataset byl pořízen mobilním telefonem a obsahuje 2 neúplné kruhy o 40, resp. 49 snímcích. Ukázka viz Obrázek 3.2f.
- **car7-phone**: dataset byl pořízen mobilním telefonem a obsahuje 2 kruhy o 48, resp. 47 snímcích. Ukázka viz Obrázek 3.2g.
- **car8-drone-active-track**: dataset byl pořízen dronem Mavic 2 s použitím funkce ActiveTrack. Obsahuje 1 oblet o 58 snímcích. Ukázka viz Obrázek 3.2h.
- **car8-drone-manual**: dataset byl pořízen manuálně ovládaným dronem Mavic 2. Obsahuje 2 oblety o 73, resp. 47 snímcích. Ukázka viz Obrázek 3.2i.
- **synthetic-porsche-911**: dataset byl vytvořen v Blenderu s použitím modelu [65]. Obsahuje 2 kruhy, každý o 50 snímcích.
- **synthetic-volvo-s90**: dataset byl vytvořen v Blenderu s použitím modelu [66]. Obsahuje 4 kruhy, každý o 25 snímcích.

Snímky pořízené dronem byly obecně zachyceny ve větší výšce než snímky pořízené mobilním telefonem. To je dáno minimální letovou výškou dronu, která při použití ActiveTrack činí 2 m,



(a) car1-drone-manual **(b)** car2-drone-manual **(c)** car3-drone-active-track

(d) car4-phone **(e)** car5-phone **(f)** car6-phone

(g) car7-phone **(h)** car8-drone-active-track **(i)** car8-drone-manual

(j) synthetic-porsche-911 **(k)** synthetic-volvo-s90

Obrázek 3.2 Ukázky datasetů.

při manuálním ovládání je možné létat ve výšce nad 1 m bez omezení a nad 50 cm s vizuálním a zvukovým varováním. Při podkročení výšky 50 cm začne dron automaticky přistávat a další snímání již není možné.

3.4 Návrhy experimentů

Vzhledem k vybraným technologiím byly navrženy tři druhy experimentů:

1. **vyhodnocení použitelnosti datasetů**: dataset je pro trénování NeRF vhodný tehdy, jsou-li k dispozici dostatečně přesné odhady pozic a parametrů kamer pro každý snímek. COLMAP je schopen správně odhadnout pozice tehdy, je-li scéna zabraná dostatečným množstvím snímků tak, aby se vždy dva sousední snímky dostatečně překrývaly. Zároveň musí být ve scéně dostatek heterogenních struktur, na kterých je COLMAP schopen extrahovat dostatečné množství kvalitních deskriptorů, pomocí nichž odhad provádí (viz sekce 2.1). V tomto

experimentu je na všech datasetech spuštěn COLMAP, jsou analyzovány odhadnuté pozice i extrahovaný sparse point cloud a je vyhodnoceno, který typ datasetu je pro další zpracování vhodný.

2. **validace modelu pro odstraňování odlesků**: na všech datasetech je spuštěn předtrénovaný model pro odstraňování odlesků. Je-li výsledek nedostačující, je vytvořen vlastní dataset vozů s odlesky a bez odlesků, na kterém bude model znovu natrénován. Takový model je opět vyhodnocen na všech dostupných datasetech a je posouzeno, zda je tento přístup dostatečně robustní, zda snímky nedeformuje a kdy je vhodné jej použít.
3. **validace modelů pro generování nových pohledů**: na všech datasetech jsou natrénovány vybrané modely a následně je vyhodnocena kvalita nově vygenerovaných pohledů, se zvlášním zřetelem na úhly a vzdálenosti neobsažené v trénovacím datasetu tak, aby byla objektivně vyhodnocena učící a generalizační schopnost modelu.

Kapitola 4

Implementace

Implementace je postavená na architektuře mikroslužeb, které jsou implementovány jako Docker kontejnery. Účelem je, aby každá mikroslužba byla nezávislá na ostatních a ke svému běhu potřebovala pouze samotný dataset. Jednotlivými mikroslužbami jsou:

1. `colmap`: použití nástroje COLMAP pro odhad parametrů kamer. Podrobněji viz sekce 4.2.
2. `reflections`: segmentace vozu pomocí modelu GLEE, detekce zatmavení oken za použití modelu YOLO a odstranění odlesků z karoserie pomocí sítě DSRNet. Podrobněji viz sekce 4.3.
3. `mipnerf360`: trénování modelu Mip-NeRF 360 a generování pohledů z 9 kružnicových trajektorií v různých výškách a vzdálenostech od vozu. Podrobněji viz sekce 4.4.
4. `zipnerf`: trénování modelu Zip-NeRF a generování a generování pohledů z 9 kružnicových trajektorií v různých výškách a vzdálenostech od vozu. Podrobněji viz sekce 4.5.
5. `gaussian-splatting`: trénování modelu Gaussian Splatting a generování pohledů ze dvou různých trajektorií. Podrobněji viz sekce 4.6.
6. `postprocessing`: segmentace vozu pomocí modelu GLEE a jeho umístění do scény s neutrálním pozadím. Podrobněji viz sekce 4.7.

Každý dataset je postupně zpracován všemi kontejnery v tomto pořadí.

4.1 Architektura systému

Architektura systému byla koncipována tak, aby byl systém co nejjednodušším způsobem spustitelný. Zásadním požadavkem při práci s modely strojového učení je zachování konzistence vývojového a produkčního prostředí. Tímto a dalšími souvisejícími otázkami se zabývá paradigma MLOps [67]. Zachování této konzistence v praxi znamená zejména uchování neměnné verze frameworku, ve kterém byl model vyvinut (v případě této práce zejména JAX[42] a PyTorch [68]). S tím se však pojí požadavky na kompatibilitu Pythonu, CUDA [69], cuDNN [70], příslušné grafické karty a jejích ovladačů [71]. Zároveň je v práci použito několik různých modelů, přičemž každý byl vyvinut v jiné době a s odlišnými knihovnami. A konečně, během vývoje byla práce spouštěna na třech různých serverech, každý s jinou konfigurací. Je nabíledni, že není možné vytvořit jedno prostředí, ve kterém poběží všechny modely najednou a které bude triviálně nasaditelné na různá zařízení. Proto byl systém rozdělen na moduly `colmap`,

`reflections`, `mipnerf360`, `zipnerf`, `gaussian-splatting` a `postprocessing`. Schéma nově navrženého systému je na Obrázku 4.1.

Jednotlivé moduly byly kontejnerizovány s použitím Dockeru [72]. Většina použitých image vychází z Ubuntu, které má nainstalovanou pouze CUDA a cuDNN. Výjimkou je image pro Gaussian Splatting, který vychází z image s přeinstalovaným PyTorch. Do těchto image jsou v dalších vrstvách instalovány požadované balíčky a knihovny.

Při implementaci bylo nutno brát v potaz i vývoj samotného nástroje Docker, jehož starší verze nepodporují některá systémová volání typická např. pro Ubuntu 22.04. S ohledem na instalace Dockeru na starších a neaktualizovaných serverech je tedy nutné, aby kontejnery vycházely ze starších verzí Ubuntu, pro potřeby práce je použito Ubuntu 20.04.

Každý modul má svou stejnojmennou složku, která vždy obsahuje soubor `Dockerfile` definující sestavení příslušného image. Dále složka obsahuje soubor `main.py`, který s využitím interních implementací jednotlivých modelů nacházejících se ve složce `src` vykoná kód příslušného modulu. Složka pak může obsahovat i další soubory s dalšími částmi implementace, instalačními skripty aj. Dále implementace obsahuje složku `scripts`, kde lze souborem `run-module.sh` spustit jeden modul nad jedním datasetem. Skript přijímá název modulu, informaci o dostupných grafických kartách a název datasetu. Následně provede sestavení image a spustí příslušný kontejner. Složka `scripts` též obsahuje soubor `run-all.sh`, který přijímá název datasetu, pro který pak postupně spustí všechny moduly.

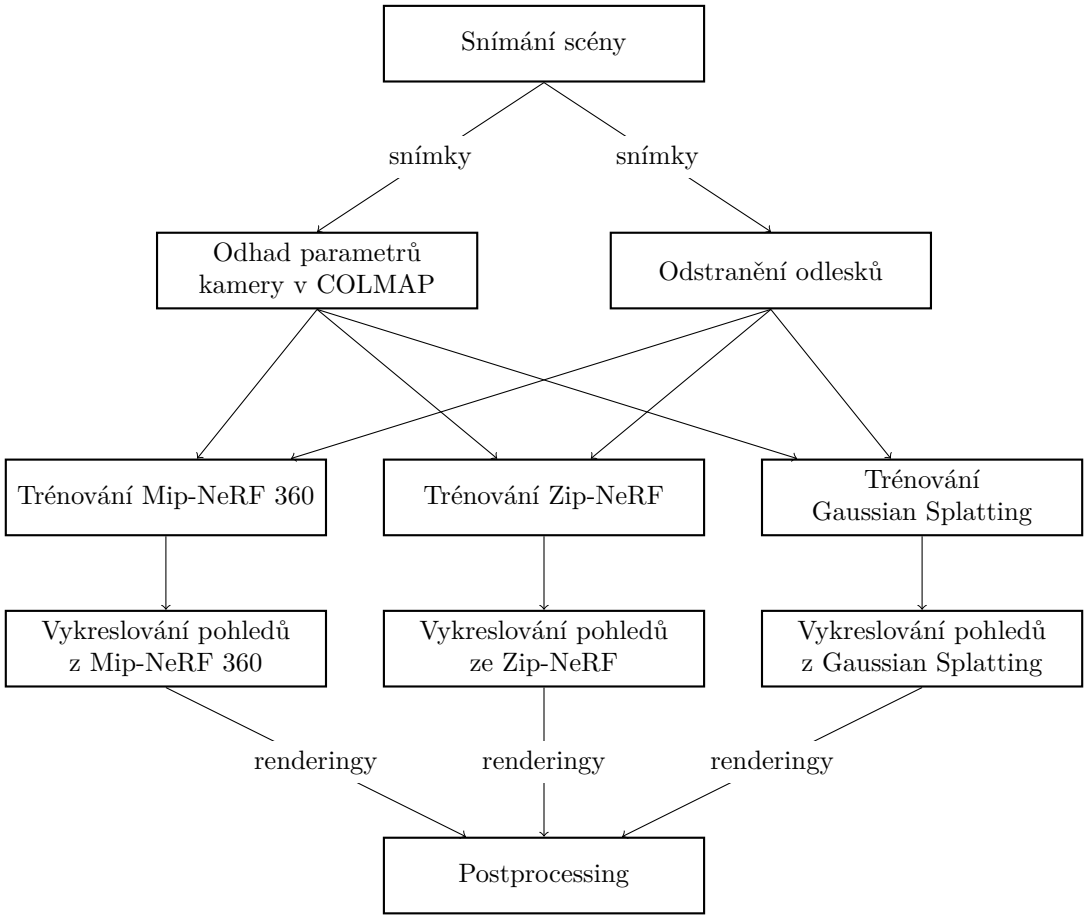
Datasety je potřeba umístit do složky `data-raw`. Každý dataset musí být v samostatné složce, jejíž název bude odpovídat názvu datasetu předávaného do `run-module.sh` nebo `run-all.sh`. Tento dataset musí obsahovat složku `images` se snímky scény s automobilem, pro který chceme vygenerovat nové pohledy.

4.2 COLMAP

Tento modul provádí odhad parametrů kamer ze snímků pomocí nástroje COLMAP (sekce 2.1). Nejprve zkopíruje dataset ze složky `data-raw` do složky `data-processed`, aby nebyla narušena integrita původních dat. Následně provede extrakci deskriptorů, poté vybere náhodně dva snímky, pomocí triangulace lokalizuje deskriptory viditelné z obou těchto snímků a poté postupně přidává do scény další snímky (image registration) a trianguluje další deskriptory. Nakonec je provedena korekce extrahovaných parametrů kamer a pozic triangulovaných deskriptorů pomocí bundle adjustment. Extrahované parametry kamery spolu a polohy triangulovaných bodů jsou uloženy do souborů.

Jelikož je výběr iniciální dvojice snímků náhodný, může se stát, že není možné do scény registrovat všechny snímky [73]. Se zbylými snímky pak COLMAP začne odznova a registruje je do další scény. Tyto scény jsou pak procesovány utilitou `model_merger`, která hledá společné deskriptory mezi dvěma scénami a ty pak spojuje dohromady.

Na extrahované parametry a snímky je pak aplikováno odstranění distorze (utilita `image_undistorter`). Nakonec jsou vytvořeny 2krát, 4krát a 8krát změnešené verze vstupních snímků, které mohou být použity jako vstup do modelů pro generování nových pohledů (zejména Mip-NeRF 360 a Zip-NeRF), pokud by pro použití standardních snímků nedostačovala dostupná VRAM. Výstupní snímky jsou uloženy do složky `orig_images`, resp. do složek `orig_images_2`, `orig_images_4` a `orig_images_8`. Parametry kamer a další výstupy COLMAP jsou uloženy ve složce `sparse/0`. Všechny tyto složky se nachází ve složce příslušného datasetu.



Obrázek 4.1 Schéma nově navrženého systému.

Obrázek 4.2 Ukázka reálného datasetu pro odstraňování odlesků. Zleva originální snímek, průhledová vrstva s odstraněnými odlesky a reflexní vrstva obsahující výhradně odstraněné odlesky.

4.3 Odstranění odlesků

V tomto modulu jsou z originálních i zmenšených obrázků odstraněny odlesky. Nejprve je vysegmentován samotný automobil a je provedena detekce a segmentace jeho skel, která jsou poté zatmavena. Na segmentovaný vůz se zatmavenými skly (tedy zejména na metalízu) je aplikováno odstranění odlesků. Do snímku s odstraněnými odlesky je potom vrácena původní podoba skel a pozadí, které bylo odstraněno při segmentaci. Výsledné snímky jsou uloženy do složky `images`, resp. do složek `images_2`, `images_4` a `images_8`.

Pro trénování modelu odstraňující odlesky byly vytvořeny dva datasety: reálný dataset ze 147 snímků pořízených dronem, ze kterých byly odlesky odstraněny manuálně. Syntetický dataset byl vytvořen kompozicí datasetu snímků automobilů a snímků přírody, které byly zesvětleny a spojeny se snímkem automobilu tak, aby reprezentovaly odlesk okolního prostředí. Takto vzniklo celkem 5600 snímků.

4.3.1 Odstraňování odlesků v GIMP

Reálný dataset byl vytvořen na základě datasetu `car1-drone-manual` a `car2-drone-manual`. Tyto snímky byly upraveny v programu GIMP [74], zejména s použitím klonovacího razítka [75] ve ztmavovacím módu. Pro potřeby modelu DSRNet pak byla pro každý snímek vytvořena trojice originálního snímku, průhledové vrstvy s odstraněnými odlesky a reflexní vrstvy obsahující výhradně odstraněné odlesky (Obrázek 4.2).

Odlesky nebyly odstraňovány z oblasti oken, jelikož zde se k odleskům přidává ještě průhled oknem dovnitř do vozidla, případně i opětovný průhled ven skrz další okno. Odstranění odlesků z takového povrchu s použitím klonovacího razítka je extrémně obtížné, zvlášť při důrazu na realistický vzhled snímku s odstraněnými odlesky. Proto byla okna vozidel namísto odstraňování odlesků ztmavena, což si vyžádalo použití dalšího modelu.

4.3.2 Algoritmus pro detekci a zatmavení oken

Pro účely zatmavení oken byl využit transfer learning, kdy je model předtrénovaný pro určitý úkol, použit pro úkol nový [26]. Například konvoluční síť pro detekci určité skupiny objektů je použita na detekci jiného objektu. Výhodou je, že vrstvy pro extrakci příznaků, natrénované na původním, zpravidla velmi rozsáhlém datasetu o milionech snímků, jsou použity i pro extrakci příznaků z obrázků v novém datasetu, který je pak použit pouze pro dotrénování detekčních hlav tak, aby se adaptovaly na nový úkol. Zde byl pro detekci a segmentaci skel aut použit model YOLOv8 [50], (podrobněji viz sekce 2.5.1) . Díky přístupné implementaci [31] je model velmi snadno použitelný. Dataset pro dotrénování modelu byl sestaven z většiny pořízených datasetů

Obrázek 4.3 Snímek s anotovanými maskami skel. Anotace jsou reprezentovány vyplněnými žlutými polygony.

(kromě datasetů `car8-drone-active-track` a `car8-drone-manual`) a čítá 620 snímků. Ty byly anotovány v programu COCO Annotator [76], bylo vytvořeno 3099 anotací (Obrázek 4.3).

Na těchto snímcích byl natrénován model YOLOv8. Trénování probíhalo na minibatchích 4snímků, které byly zmenšeny tak, aby delší strana měla maximálně 1280 pixelů. Maximální počet epoch byl stanoven na 1000, zároveň byl ale aktivován early stopping, který trénování zastavil po 50 epochách bez zlepšení na validačním datasetu. Model byl trénován na grafické kartě NVIDIA A100-SXM4-40GB po 342 epoch. Trénování a validace trvaly dohromady přibližně 16 hodin.

Takto natrénovaný model byl použit pro segmentaci skel aut. Jednotlivé segmentované části byly rozmazány s použitím mediánového filtru [77]. Následně byla analyzována jejich světelnost, totiž průměrná jasová hodnota těchto oblastí převedených do šedotónu. Podle té byla určena konstanta z intervalu od 0,01 pro nejsvětlenší až po 0,5 pro nejtmavší okna. Touto konstantou byly vynásobeny hodnoty pixelů v příslušné části. Ty byly nakonec opět rozmazány mediánovým filtrem. Výsledná podoba zatmavených oken je vidět např. na Obrázku 4.2.

4.3.3 Tvorba syntetického datasetu

Vzhledem k tomu, že odstraňování odlesků v GIMP bylo poměrně zdlouhavé, nebylo by tímto způsobem možné včas vytvořit dostatek snímků pro přetrénování DSRNet. Proto bylo přistoupeno k tvorbě syntetického datasetu, a to obdobným způsobem jako při trénování DSRNet popsaném v [14]:

1. byly vybrány dva snímky, snímek T představuje průhledovou vrstvu, snímek R představuje odlesk.
2. na snímek R byla aplikována 2D konvoluce s použitím 2D Gaussova filtru, jehož velikost byla zvolena z množiny $\{5, 7, 9, 11\}$ přičemž výběr větších filtrů byl pravděpodobnější. Rozptyl filtru byl taktéž zvolen náhodně jako reálné číslo z intervalu $[2; 5]$.
3. oba snímky byly zesvětleny vynásobením konstantou, přičemž T byl vynásoben náhodně vybraným reálným číslem $\alpha \in [0{,}8; 1{,}0]$ a R byl vynásoben náhodně vybraným reálným číslem $\beta \in [0{,}4; 1{,}0]$
4. výsledný obraz byl s pravděpodobností 0,7 sestaven jako $I = \alpha T + \beta R - \alpha\beta T \cdot R$ a s pravděpodobností 0,3 jako $I = T + R$.

Obrázek 4.4 Ukázka syntetického datasetu pro odstraňování odlesků. Zleva originální snímek, průhledová vrstva s odstraněnými odlesky a reflexní vrstva obsahující výhradně odstraněné odlesky.

Pro tvorbu syntetického datasetu byl použit dataset snímků automobilů [78, 79] a dataset snímků přírodních scenérií [80, 81] s tím, že přírodní scenérie byly přidávány jako odrazová vrstva ke snímkům aut podle algoritmu výše. Auto bylo vysegmentováno (viz sekce 5.4), byla detekována okna a na masku auta (mimo oblast oken) byl aplikován odlesk přírodní scenérie. Zároveň byla na snímky aplikovaná augmentace, kdy byly snímky náhodně orotovány o úhel 90, 180 nebo 270 stupňů. Jedna z vytvořených trojic snímků je zobrazena na Obrázku 4.4.

4.4 Mip-NeRF 360

V tomto modulu probíhá trénování modelu Mip-NeRF 360 [82] a generování nových pohledů z tohoto modelu. Nejprve jsou načtena trénovací data, primárně ze složky `images`, pokud je však model konfigurován pro práci s menšími obrázky, je použita složka s přísušným zmenšovacím faktorem. Autoři stanovili délku trénování na 250 000 kroků, to však zabere přibližně 24 hodin. Experimentálně bylo zjištěno, že v případě kvalitního odhadu pozičních parametrů kamery model produkuje kvalitní výstupy již po 25 000 - 50 000 krocích trénování. Jelikož se jedná o výrazný odklon od doporučení autorů, byl implementován early stopping, který sleduje, nakolik se vždy po 100 krocích zlepšuje trénovací PSNR a pokud se ani po 1000 krocích PSNR nezvětší alespoň o 0,01, je trénování ukončeno. Takto je zabráněno nedotrénování modelu a zároveň je trénování dostatečně brzo ukončeno. Každých 1000 kroků a následně po ukončení trénování je uložen aktuální stav modelu do složky `mipnerf360_checkpoints`, přičemž z úsporných důvodů jsou zachovány vždy pouze 2 poslední checkpointy.

Po natrénování modelu ihned následuje generování nových pohledů. To je implementováno tak, aby se mohlo flexibilně přizpůsobit požadavku na vygenerování pohledu z téměř libovolného úhlu. Základem je generování elipsovité trajektorie pohledů, kdy je nejprve vypočítán střed scény. Následně je vypočítána elipsa přibližně opisující pozice kamer ve scéně. Nakonec je vypočítána výška, ze které bude příslušný pohled pořízen, přičemž ta osciluje mezi 10. a 90. percentilem výšky trénovacích pohledů.

Tato implementace byla doplněna o možnost škálování excentricity generované elipsy (a tím generovat pohledy poněkud přiblížené či oddálené od původní roviny snímání) a také zafixování výšky pohledu na 10., 50. a 90. percentil výšky trénovacích pohledů. Pro každou z těchto 9 konfigurací je vygenerováno 30 snímků, které jsou pak spojeny do videa. Výsledky jsou ukládány do složek `mipnerf360_render_xy<>_z<>`, kdy na první vynechané místo je doplněna konstanta pro škálování excentricity a na druhé místo je doplněn indikátor výšky, a to „varying“ pro oscilující výšku, „low“ pro 10., „middle“ pro 50. a „high“ pro 90. percentil výšky.

4.5 Zip-NeRF

V tomto modulu probíhá trénování modelu Zip-NeRF [83] a generování nových pohledů z tohoto modelu. Zip-NeRF je implementován velmi podobně jako Mip-NeRF 360. Early stopping je implementován se stejným nastavením jako u Mip-NeRF 360. Během trénování Zip-NeRF jsou také optimalizovány parametry kamer (viz sekce 2.2.6). Generování nových pohledů je implementováno podobně jako u Mip-NeRF 360 – jsou generovány na 9 trajektoriích o různé vzdálenosti od vozu a v různé výšce.

4.6 Gaussian Splatting

V tomto modulu probíhá trénování modelu Gaussian Splatting [84] a generování nových pohledů z tohoto modelu. Během trénování je provedeno 30 000 kroků, po 7000. a 30 000. kroku je aktuální stav modelu uložen. Následně jsou vygenerovány dvě skupiny pohledů, nejprve pohledy z pozic trénovacích snímků. Tyto pozice jsou následně mírně posunuty ve všech třech osách a použity pro vygenerování dalšího datasetu pro zjištění generalizačních schopností modelu.

4.7 Postprocessing

V tomto modulu jsou postupně zpracovány všechny snímky vykreslené moduly Mip-NeRF 360, Zip-NeRF a Gaussian Splatting. Z těchto snímků je pomocí modelu GLEE (viz sekce 2.5.2) vysegmentován automobil a umístěn do neutrálního pozadí [85]. Takto vytvořené snímky jsou umístěny do samostatných složek. V případě neúspěšné segmentace je uloženo pouze neutrální pozadí bez automobilu.

Kapitola 5

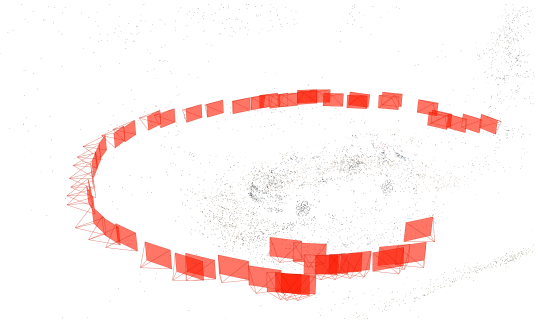
Experimenty a výsledky

Zásadní prerekvizitou pro generování nových pohledů je kvalitní odhad pozic kamer. Proto byl na datasetech nejprve spuštěn COLMAP a bylo vyhodnocováno, zda byl v datasetu dostatek snímků s dostatečným překryvem, tak, aby odhad COLMAPu odpovídal skutečnosti. Ze snímků pak byly odstraněny odlesky a na každém datasetu pak byly natrénovány postupně tři modely pro generování nových pohledů – Mip-NeRF 360, Zip-NeRF a Gaussian Splatting, u kterých pak bylo vyhodnoceno, zda se scénu dokázaly naučit a zda byly schopny vygenerovat realistické pohledy z nových úhlů a odlišných vzdáleností. Z vygenerovaných pohledů pak bylo vozidlo vysegmentováno a umístěno do scény s neutrálním pozadím.

5.1 COLMAP

Odhady pozic kamer a extrahovaný sparse point cloud byl vizualizován v grafickém rozhraní COLMAP [86]. Na každém snímku jsou červeně zobrazeny odhadnuté pozice kamer a roviny jednotlivých snímků. Dále jsou vizualizovány body řídkého point cloudu, které byl COLMAP schopen triangulovat. Pro robustnost algoritmu je důležité, aby byl COLMAP schopen extrahovat maximální množství příznaků přímo na automobilu a ne jinde, tak, aby byl algoritmus nezávislý na exteriéru, ve kterém byly snímky auta pořízeny. Pro jednotlivé reálné datasety byly výsledky následující:

- **car1-drone-manual**: zde COLMAP správně odhadnul pozice kamer, které tvoří jeden neúplný kruh (Obrázek 5.1). Zároveň je patrné, že většinu důležitých triangulovaných bodů našel přímo na automobilu, a to zejména na kolech, kapotě a oknech. Jen některé body byly nalezeny na vozovce před autem.

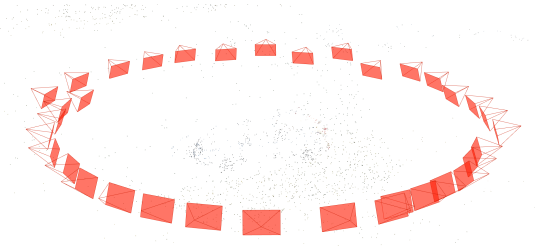


Obrázek 5.1 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `car1-drone-manual`.

- **car2-drone-manual**: zde COLMAP správně odhadnul pozice kamer, které tvoří jeden kruh (Obrázek 5.2). Snímků je však mnohem více a také se lépe překrývají. Dostatečné množství bodů bylo nalezeno na automobilu, nezanedbatelná část se jich však nachází na vozovce nabo na okolních stromech.

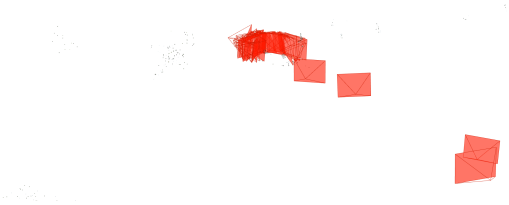
Obrázek 5.2 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `car2-drone-manual`.

- **car3-drone-active-track**: zde COLMAP správně odhadnul pozice kamer, které tvoří jeden kruh (Obrázek 5.3). Snímků je méně, překryv je horší. Také je patrné, že byl dataset pořízen s využitím ActiveTrack, jelikož jsou snímky umístěny výrazně výše než u předchozích manuálně pořízených datasetů. Triangulovaných bodů je také výrazně méně, potěšitelně se ovšem drtivá většina z nich nachází na automobilu (zejména v okolí kol a přední SPZ), okolí k rekonstrukci přispělo jen velmi málo.



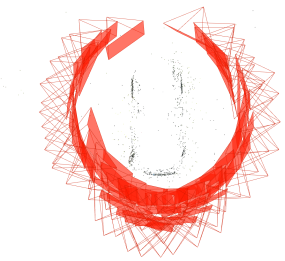
Obrázek 5.3 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `car3-drone-active-track`.

- **car4-phone**: zde COLMAP zcela selhal (Obrázek 5.4), zejména z důvodu naprosto nedostatečného překryvu snímků, kterých sice nebylo o mnoho méně než u předchozího datasetu, nicméně na rozdíl od snímků pořízených dronem s ActiveTrack, nyní byly snímky pořízeny mobilním telefonem z výrazně menší vzdálenosti, proto nemají dostatečný překryv.



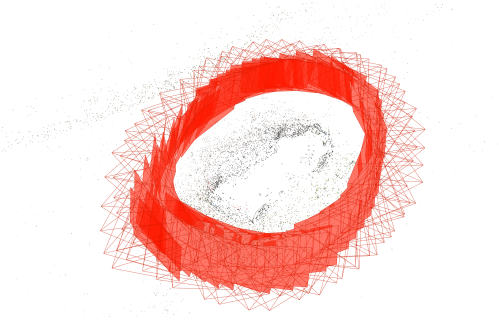
Obrázek 5.4 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car4-phone`.

- **car5-phone**: zde COLMAP uspěl pouze částečně. Sice správně odhadnul pozice většiny snímků, nicméně zaměnil některé snímky zádi vozu, které umístil zcela na opačnou stranu, jako by zabíraly příď (Obrázek 5.5). Triangulované body se nachází výhradně na automobilu, opět zejména v oblasti kol, SPZ a kolem oken. Z toho lze usoudit, že na standardním automobilu je COLMAP schopen extrahovat SIFT deskriptroy, které spolehlivě popíšou zejména boky automobilu s tím, že je COLMAP schopen rozeznat pravý a levý bok. Potíž však nastává na přídi a zádi, kde sice nalezne mhoho příznaků kolem SPZ, pravděpodobně se však u zadní SPZ domnívá, že se jedná o přední SPZ, na zbytku zádi není schopen extrahovat dostatečné množství unikátních deskriptorů a tedy dojde k záměně.



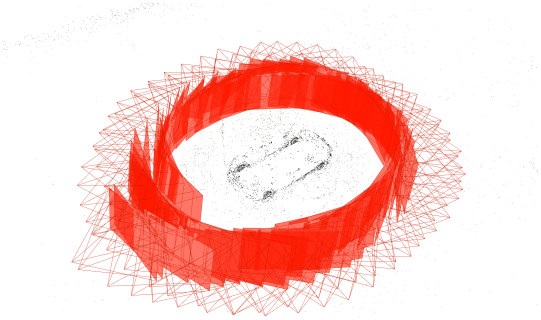
Obrázek 5.5 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car5-phone`.

- **car6-phone**: zde COLMAP správně odhadnul pozice kamer, které tvoří dva úplné kruhy (Obrázek 5.6). Na automobilu byl schopen nalézt příznaky zejména v oblasti kol, přední i zadní SPZ, pod kapotou na logu výrobce a dále i na zádi auta. To doplňuje předchozí pozorování a odhaluje, že nutnou podmínkou pro odlišení přídi a zádi auta je nalezení příznaků i mimo SPZ. Zároveň však COLMAP využil mnoho příznaků mimo automobil, zejména na zemi a částečně i na stromech v okolí.



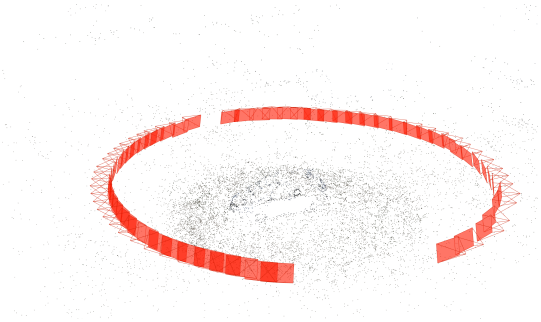
Obrázek 5.6 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car6-phone`.

- **car7-phone**: zde COLMAP správně odhadnul pozice kamer, které tvoří dva úplné kruhy (Obrázek 5.7). Většina příznaků byla nalezena v oblasti kol, na kapotě a rámu zadního skla, menší množství příznaků bylo nalezeno na zemi.



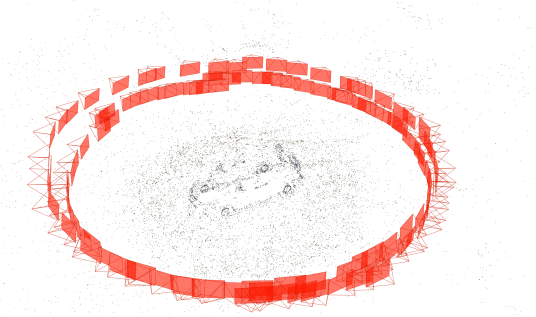
Obrázek 5.7 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car7-phone`.

- **car8-drone-active-track**: zde COLMAP správně odhadnul pozice kamer, které tvoří jeden neúplný kruh (Obrázek 5.8), chybí část pohledů z levého boku. Pouze menšina příznaků byla nalezena na automobilu, většina je na výrazně členité, a tudíž na deskriptory bohaté, zemi.



Obrázek 5.8 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car8-drone-active-track`.

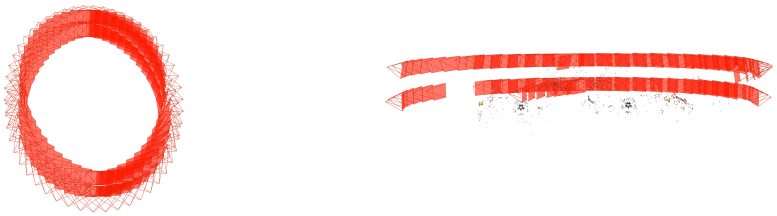
- **car8-drone-manual**: zde COLMAP správně odhadnul pozice kamer, které tvoří dva úplné kruhy, přičemž během nižšího obletu bylo pořízeno výrazně více snímků než během obletu vyššího. Značná část příznaků se nachází na zemi.



Obrázek 5.9 Vizualizace odhadnutých pozic kamer a extrahovaného sparse point cloudu pro dataset `colmap-car8-drone-manual`.

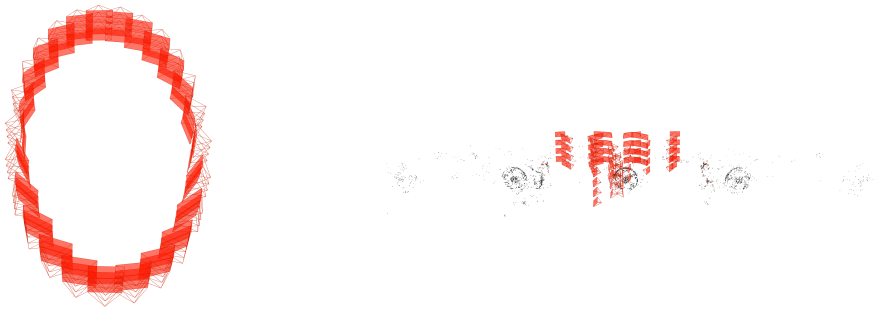
Při generování syntetických datasetů byly v Blenderu spolu se snímky vygenerovány i přesné pozice kamer v COLMAP formátu. Syntetické datasety byly generovány s neutrálním pozadím, jsou tedy zvlášť chudé na unikátní regiony dobře popsatelné deskriptorem. Přesto však byl COLMAP na těchto datech spuštěn. Na následujících obrázcích jsou pro každý dataset nejprve v levém sloupci vizualizovány pozice kamer z Blenderu. Tyto vizualizace neobsahují řídký point cloud, jelikož Blender neprovádí triangulaci a nemá jej tedy z čeho generovat. Následně jsou v pravém sloupci vizualizovány pozice kamer odhadnuté COLMAPem spolu s řídkým point cloudem:

- **synthetic-porsche-911**: nejprve jsou vidět přesné pozice kamer seskupených ve dvou kruzích. COLMAP však na této scéně nedokázal rozlišit pravý bok od levého, snímky zabírají pouze příď, záď a jeden bok. Příznaky byly nalezeny na jménu značky vykreslenému na zádi a pravděpodobně na jednom kole.



Obrázek 5.10 Vizualizace přesných pozic kamer pro dataset `syhthetic-porsche-911` (vlevo) a odhadnutých pozic kamer spolu s extrahovaným sparse point cloudem (vpravo).

- **synthetic-volvo-s90**: nejprve jsou vidět pozice kamer seskutených ve čtyřech kruzích. Z druhého obrázku však vyplývá, že COLMAP měl opět problémy s odlišením některých stran automobilu. Je vidět, že nalezl deskriptory na zádi auta, okolo zadních světel a zadních kol, přední kola nedokázal rozlišit a body na nich nalezené umístil do jednoho shluku.



Obrázek 5.11 Vizualizace přesných pozic kamer pro dataset `syhthetic-volvo-s90` (vlevo) a odhadnutých pozic kamer spolu s extrahovaným sparse point cloudem (vpravo).

Z popsaných výsledků vyplývá, že COLMAP na syntetických datech není schopen odhadnout pozice kamer dostatečně přesně na to, aby mohly být využity pro trénování NeRF. Proto byly pro experimenty se syntetickými datasety použity pouze COLMAP pozice exportované z Blenderu. Zároveň také vyplývá, že sice není nutné mít několik obletů ve více výškových rovinách, je však nutné, aby se vždy dva sousední snímky dostatečně překrývaly, jinak COLMAP selže.

5.2 Nerfstudio

Nerfstudio [87] je nástroj, který umožňuje vytvářet, trénovat a testovat modely typu NeRF. Poskytuje implementace několika metod (např. Instant NGP [10], NeRF [1], Mip-NeRF [6]), které lze trénovat na několika připravených datasetech, zejména těch, na kterých modely trénovali jejich autoři. Modulární architektura nástroje však umožňuje i použití vlastních datasetů nebo implemetnaci vlastních modelů.

V rámci práce byla testována implementace Mip-NeRF na několika reálných datasetech. Výsledky však nebyly uspokojivé i přes použití scén s kvalitně odhadnutými pozicemi kamer a velmi dlouhou dobou trénování (1 000 000 trénovacích kroků), viz Obrázek 5.12, kde je vidět, že model odhadnul pouze část pozadí a přidal do scény mnoho nevysvětlitelných artefaktů. Vzhledem k dosaženým výsledkům, mnohým nedostatkům modelu Mip-NeRF (viz sekce 2.2.3) a nedostupnosti implementace pokročilejších metod v rámci tohoto nástroje, např. Mip-NeRF 360, bylo od jeho dalšího použití upuštěno.

Obrázek 5.12 Výsledky Mip-NeRF v Nerfstudio pro vybrané datasety po 999 500trénovacích krocích. Vlevo originální testovací snímek, uprostřed a vpravo výsledky modelu.

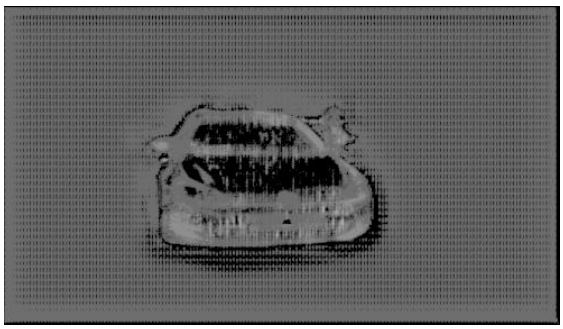
5.3 Odstranění odlesků

Dle benchmarku na portálu Papers With Code [21] dosahuje nejlepších výsledků na téměř všech testovaných datasetech model DSRNet (viz sekce [14]). Model nabízí dobře použitelnou implementaci [88], která byla byla otestována na reálných datasetech.

5.3.1 Předtrénovaný model

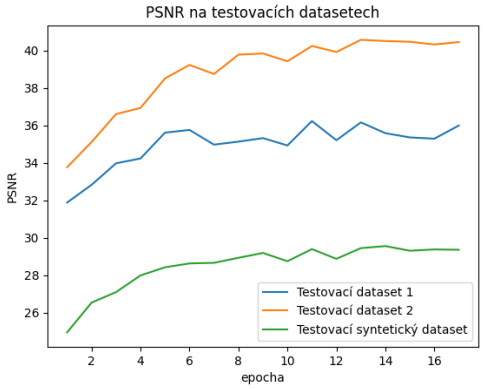
Nejprve byl otestován předtrénovaný model, který byl trénován na datasetu PASCAL VOC [89], Real20 [12] a Nature [90]. Výsledky však nebyly přesvědčivé, jak je patrné z Obrázku 5.13, kde průhledová vrstva je prakticky totožná se vstupním obrázkem, k odstranění odlesků nedošlo, nedošlo ani k jejich separaci do dekompozičního rezidua. Podrobnější analýza odrazové vrstvy (Obrázek 5.14) ukázala, že model dokáže odlesky detekovat správně, nicméně nikoli s dostatečnou intenzitou, a proto je výsledná průhledová vrstva na pohled k nerozeznání od vstupního obrázku.

Obrázek 5.13 Výsledky předtrénovaného modelu. Zleva vstupní obrázek, predikovaná průhledová vrstva, predikované residuum.



Obrázek 5.14 Predikovaná odrazová vrstva převedená do šedotónu a přeškálovaná z hodnot $[0; 10]$ na hodnoty $[0; 255]$.

Pravděpodobnou příčinou nemožnosti jednoduché adaptace předtrénovaného modelu na odstraňování odlesků z metalízy vozu je rozdíl v povaze obou úloh. Trénovací datasety tvořeny buď pohledy přes sklo nebo kompozicí dvou obrazů. Datasety použité v této práci však odlesky způsobené pohledem přes sklo neobsahují a z odlesků okolí na metalíze není možné vytvořit ucelenou scénu, která byla použita ke kompozici s jinou scénou při tvorbě syntetického datasetu. Textura okolí se na metalízu odráží jen místy a navíc je deformovaná tvarem karoserie i úhlem, ze kterého byl snímek reflexivní části pořízen. Proto byl vytvořen vlastní dataset, jak bylo popsáno v sekci 4.3.



Obrázek 5.15 PSNR na testovacích datech

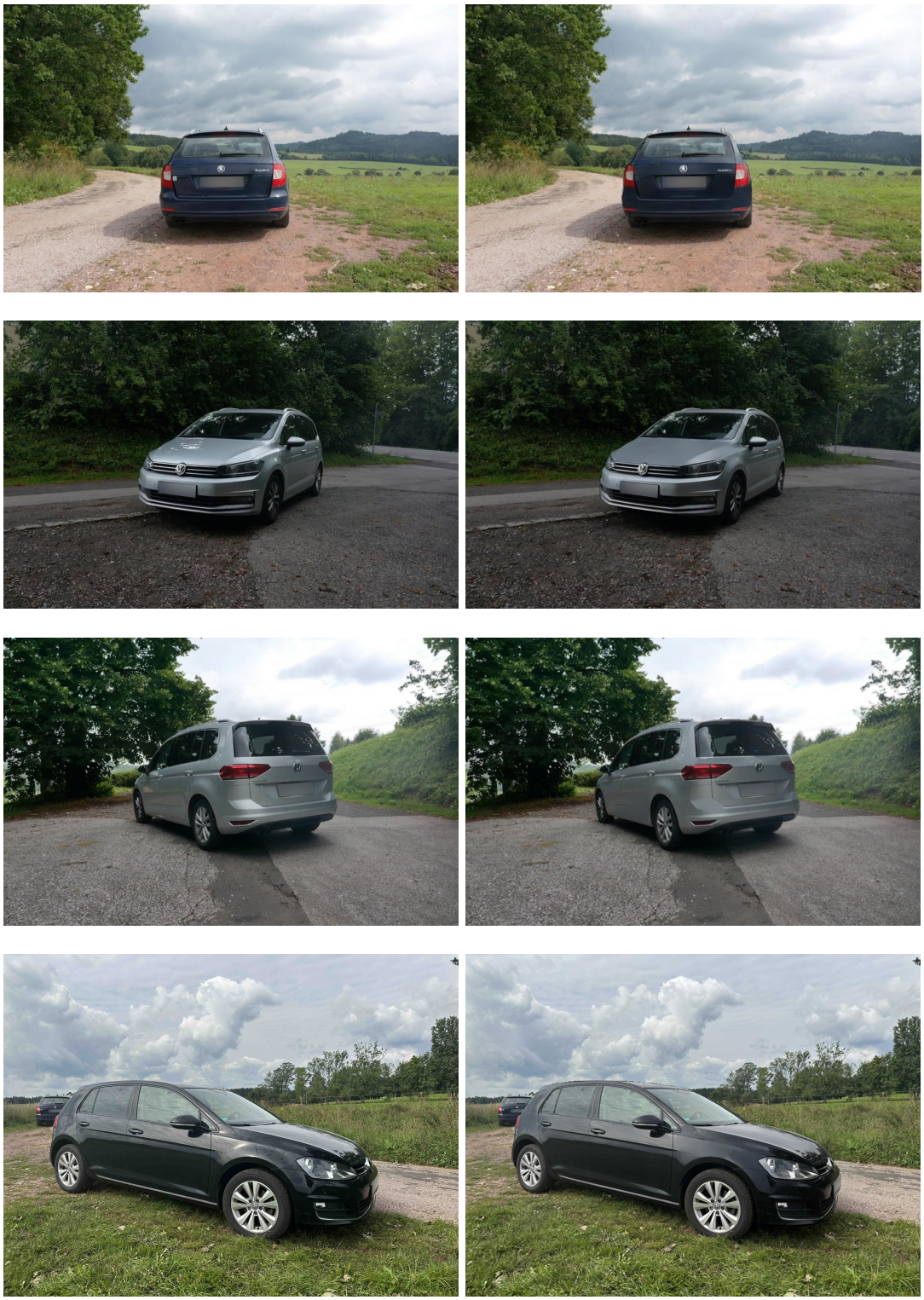
5.3.2 Vlastní trénování

Na vlastním datasetu byl DSRNet znovu natrénován. Trénování bylo velmi náročné na VRAM, proto minibatche obsahovaly pouze 1 snímek. Ten byl zmenšen tak, aby delší strana měla maximálně 864 pixelů. Maximální počet epoch byl nastaven na 60, trénování však bylo monitorováno a jakmile již nedocházelo k výraznému zlepšování PSNR na testovacích datech (viz Obrázek 5.15), bylo trénování po 17 epochách zastaveno. Model byl trénován na grafické kartě NVIDIA A100 80 GB po dobu přibližně 34 hodin. Náročnost na VRAM lze podtrhnout faktem, že model při použití minibatche o jednom nepříliš velkém snímku zabral při trénování 75 GB VRAM. Při inferenci však model zabírá výrazně méně a je nasaditelný na standardní GPU.

Výsledky pro některé snímky jsou zobrazeny na Obrázku 5.16. Je patrné, že model si zvlášť dobře poradil s přesaturovanými regiony způsobenými přímými odlesky slunce. Na poslední dvojici obrázků je také patrné, že model odstranil i část složitějších odlesků okolní vegetace, zejména z oblasti nad pravým předním kolem.

5.4 Segmentace

Původní implementace systému pro generování nových pohledů [57] využívala k segmentaci vozidel knihovnu [91], konkrétně zde dostupné modely U2-Net [59] a SAM [60], kterému jako prompt předával bounding box automobilu detekovaného modelem YOLO [31]. Tento postup byl reimplementován tak, že nejprve proběhla detekce a segmentace s využitím YOLO a SAM. Pokud YOLO selhalo, byl použit U2-Net, který kromě snímku samotného žádný další vstup nepotřeboval a důležitý objekt ve scéně určil sám. Vrácena byla vždy maska největšího automobilu. Při testování na dostupných datasetech však bylo zjištěno, že tento postup je nedostačující ze dvou důvodů: ačkoliv je YOLOv8 široce používané a třída automobil se vyskytuje i v důležitých trénovacích datasetech, na kterých bylo YOLO trénováno, model selhal zejména při pohledech z netradičního úhlu nebo pokud na autě bylo zvlášť velké množství odlesků se složitější texturou. Oba segmentační modely pak k segmentovanému vozu místy přidávaly i některé prominentní objekty na pozadí, například zeleň či stavby. Takto vysegmentované snímky pak nebylo možné použít pro trénování NeRF, jelikož tyto nekonzistentní dodatečné objekty ve scéně model matou a výsledky pak obsahují velké množství artefaktů, popř. různých nevysvětlitelných obláčků. Proto byl tento segmentační postup zcela přepracován, a použit instance segmentation model GLEE [35, 92], podrobněji viz sekce 2.5.2. Tento model již těmito problémy netrpěl a poskytoval



Obrázek 5.16 Výsledky DSRNet na reálných datech. Nejprve je vlevo zobrazen originální obrázek, vpravo je zobrazen snímek po odstranění odlesků z karoserie.

Obrázek 5.17 Porovnání výsledků původní a nové segmentační techniky.

výrazně spolehlivější výsledky bez nežádoucích artefaktů. Výsledky původní a nové segmentační techniky jsou porovnány na Obrázku 5.17.

5.5 Mip-NeRF 360

Nejprve bylo testováno, zda je pro trénování vhodnější použití kompletních snímků vozu s pozadím, nebo snímků obsahujících pouze vysegmentovaný automobil. Původní implementace [57] použila segmentované snímky, čímž u modelu NeRF docílila kvalitnějších a méně zašuměných výsledků. Jak však vyplývá ze sekce 2, pokročilejší architektury NeRF mají výrazně vyšší kapacitu a jsou adaptovány i pro komplexní scény. Výsledky na segmentovaných datech (Obrázek 5.18) však nebyly uspokojivé:

- na Obrázku 5.18a je zejména v oblasti kol a předního nárazníku patrné, že v původním snímku auto stálo na trávníku, jelikož spolu s pneumatikami byla vysegmentována i stébla trávy. Na předním nárazníku je také patrný výrazný odlesk okolní zelené scény.
- na Obrázku 5.18b je nesprávně vysegmentovaný automobil, kterému chybí část střechy a pravého boku. To je způsobeno zejména nekonzistentní segmentací sítí U2-Net [59] v různých snímcích. Části vozu, které v některých snímcích segmentovány byly, a v jiné nikoli, pak způsobily, že je model téměř vůbec nevzal v potaz a auto pak nevypadá realisticky. Použití kvalitnějších segmentačních technik (SAM + YOLO [60, 31], případně GLEE [35]) sice tyto nekonzistence výrazně omezilo, zcela je však neeliminovalo.
- na Obrázku 5.18c jsou na pozadí vidět bílé artefakty, které by se v segmentované scéně neměly vyskytovat. Tyto artefakty vznikly z důvodu nekonzistence průhledů okny a skutečné scény. Syntetická data byla generována ve scéně s neutrálním bílým pozadím. Toto pozadí bylo segmetnací odstraněno a zůstal pouze samotný automobil. Segmentací však nebylo dotčeno to, co bylo vidět průhledem skrz skla automobilu, kde stále zůstalo původní bílé pozadí. To model správně interpretoval a umístil tuto bílou hmotu do správné vzdálenosti za automobil. Jakmile se však pohled otočil, a část scény, která byla původně viditelná pouze skrz okno, začala být viditelná i přímo, nastala nekonzistence, jelikož v přímém pohledu byla původní bílá hmota odstraněna při segmentaci.
- na Obrázku 5.18d je hloubková mapa scény, která byla natrénována ze segmentovaných dat. Absence pozadí způsobí, že je reprezentována jen výseč scény ohraničená trajektorií fotoaparátu. Pohledy nacházející se na hraně této výseče, případně za ní, pak už nejsou schopny pracovat s informací uvnitř, proto jsou tyto pohledy zastřené, případně se v nich automobil vůbec nenachází. To je v rozporu s požadavkem generovat pohledy v různých vzdálenostech, když je spolehlivě možné generovat pouze bližší pohledy.

Kvůli těmto problémům a vzhledem k dostatečné kapacitě použitých modelů, byly modely

(a) Okolní scéna zasahující do segmentovaného snímku.

(b) Nesprávně vysegmentovaný automobil.

(c) Nekonzistence průhledů okny a pozadí.

(d) Hloubková mapa scény natrénované ze segmentovaných dat.

Obrázek 5.18 Výsledky modelu Mip-NeRF 360 na segmentovaných datech.

trénovány na snímcích s pozadím. Odhadnul-li COLMAP správně parametry kamery pro příslušný záběr (ideálně pro všechny snímky v datasetu), byl model schopen generovat kvalitní pohledy z různých vzdáleností, jak je vidět například na Obrázku 5.19a. Na pohledech pořízených z větší vzdálenosti (Obrázek 5.19b) je patrná místy nižší kvalita části scény, která se nově ocitla v záběru, a kterou se model učil pouze z některých bočních pohledů. Tato část scény však nikdy neobsahuje automobil. Z toho však také vyplývá, že mají-li být generovány pohledy z větší vzdálenosti než původní záběry, je nutné, aby původní záběry obsahovaly vždy celý automobil ideálně i s malým přesahem do pozadí. Na výsledcích pro dataset `car3-drone-active-track` (Obrázek 5.19c a 5.19d) je vidět chování modelu pro části scény, kde nebyly správně odhadnuty pozice kamer. Snímky ukazují, že v případě zcela nesprávně odhadnutých parametrů kamery není možné dosáhnout kvalitních výsledků a trénování modelů v tomto případě je zcela zbytečné. Nabízí se zavést verifikaci odhadnutých parametrů tak, aby se předešlo zbytečným finančním i časovým nákladům trénování na datasetech s nesmyslnými pozicemi.

Kompletní výsledky modelu Mip-NeRF 360 na reálných datasetech jsou zachyceny v Příloze na Obrázku A.1. Každému datasetu odpovídá jeden řádek. Na každém řádku jsou ukázány vždy dvě dvojice pohledů ze stejného úhlu, ale z jiné vzdálenosti. Na snímcích byly manuálně rozmazány SPZ.

Na syntetických datasetech, kde jsou pozice zcela přesné, si model vedl velmi dobře. Pouze na některých výsledcích pro dataset `synthetic-volvo-s90` došlo k deformaci nad levým zadním kolem, tyto problémy však byly odstraněny posunutím záběru dále. Výsledky modelu na syntetických datech jsou zachyceny v Příloze na Obrázku A.2

5.6 Zip-NeRF

Na výsledcích modelu Zip-NeRF (Obrázek 5.20c) je vidět, že na snímcích se špatně odhadnutými pozicemi kamer je vidět vícenásobný posun vozidla, který je dán snahou utility CamP o optimalizaci pozic kamer, kdy zejména translace v osách x a z pro dataset `car4-phone` nekonvergovala. Problémy měl model také se syntetickými daty (Obrázek A.4 v Příloze), kdy z některých pohledů vůz částečně či zcela zmizel. I ve scénách s dobře odhadnutými pozicemi kamer (Obrázek 5.20a a 5.20b) je ve scénách více artefaktů a výsledky nejsou tak čisté jako u modelu Mip-NeRF 360. Kompletní výsledky modelu Zip-NeRF na reálných datasetech jsou zachyceny v Příloze na Obrázku A.3.

(a) Nový pohled na vozidlo z datasetu `car2-drone-manual`.

(b) Nový pohled na vozidlo z datasetu `car2-drone-manual` zachycený z větší vzdálenosti.

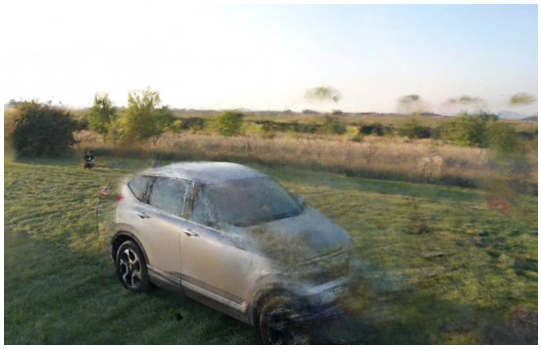
(c) Nový pohled na vozidlo z datasetu `car4-phone`.

(d) Nový pohled na vozidlo z datasetu `car4-phone` zachycený z větší vzdálenosti.

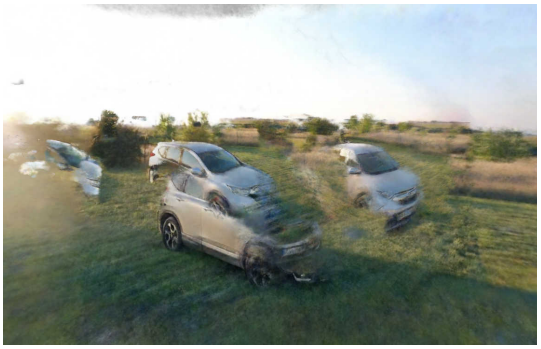
Obrázek 5.19 Výsledky modelu Mip-NeRF 360 na vybraných datasetech.

(a) Nový pohled na vozidlo z datasetu `car2-drone-manual`.

(b) Nový pohled na vozidlo z datasetu `car2-drone-manual` zachycený z větší vzdálenosti.



(c) Nový pohled na vozidlo z datasetu `car3-drone-active-track`.



(d) Nový pohled na vozidlo z datasetu `car3-drone-active-track`. zachycený z větší vzdálenosti.

Obrázek 5.20 Výsledky modelu Zip-NeRF na vybraných datasetech.

5.7 Gaussian Splatting

Model Gaussian Splatting byl spouštěn pouze na datasetech pořízených mobilním telefonem nebo dronem. Na syntetických datasetech spuštěn nebyl, jelikož zde pozice kamer byly exportovány přímo z Blenderu namísto použití COLMAP, a sparse point cloud produkovaný COLMAP tak není k dispozici. Snímky jsou vždy ukázány ve dvojicích – nejprve je ukázána scéna vykreslená v jedné z trénovacích pozic, následuje scéna vykreslená v pozici mírně posunuté od předchozí pozice. Tím je testováno, zda model dokáže správně generalizovat a vykreslit scénu z úhlů nepokrytých v trénovacím datasetu.

Všechny scény se správně odhadnutými pozicemi kamer byl model schopen správně vymodelovat, oproti výsledkům modelů Mip-NeRF 360 a Zip-NeRF je ve snímcích méně šumu a objekty působí hladším dojmem. Některé části pozadí datasetu `car5-phone` jsou rozmazané (Obrázek 5.21a a 5.21b), textury zde pravděpodobně nebylo možné dostatečně odlišit (příslušná část trávníku splynula se zbytkem trávníku), proto byly reprezentovány generičtěji pomocí rozmazané textury podobné barvy.

5.8 Postprocessing

Model GLEE úspěšně segmentoval drtivou většinu snímků vycházejících z datasetů pořízených mobilním telefonem, nebo dronem. Větší problémy měl se syntetickým datasetem, selhával zejména

(a) Pohled na vozidlo z datasetu `car5-phone` odpovídající pozici z trénovacího datasetu. **(b)** Nový pohled na vozidlo z datasetu `car5-phone`.

Obrázek 5.21 Vybrané výsledky modelu Gaussian Splatting.

v pohledech z netradičních úhlů nebo takových, které zabíraly pouze část automobilu. Vzhled segementovaného vozu zasazeného do neutrálního pozadí je demonstrován na Obrázku 5.22.

Obrázek 5.22 Segmentovaný vůz zasazený do neutrálního pozadí.

Kapitola 6

Diskuze

Zásadní limitací COLMAP se ukázala velmi omezená schopnost extrakce příznaků na hladké metalíze, díky čemuž pak systém selhal na scénách s menším množstvím snímků nebo nedostatečně heterogenním pozadím. Bude tedy potřeba analyzovat alternativní algoritmy pro extrakci příznaků tak, aby mohly být spolehlivě párovány i příznaky přímo na metalíze.

Při odstraňování odlesků musely být snímky zpracované modelem DSRNet zmenšeny tak, aby delší strana měla maximálně 864 pixelů. To bylo způsobeno neúměrnou paměťovou náročností modelu při trénování, kdy i při použití takto zmenšených snímků proces zabíral 75 GB VRAM. Model byl opakovaně trénován na různých grafických kartách s různou maximální velikostí snímků a bylo zjištěno, že se zvyšujícím se rozlišením snímků model funguje lépe. Navíc při inferenci model zabírá jen zlomek paměti nutné pro trénování. Aby bylo možné použít snímky v originální velikosti, bylo by nutno model natrénovat pouze na CPU na stroji s RAM v řádu nižších stovek GB. Inference by pak mohla probíhat standardně na GPU. Takové trénování by však trvalo velmi dlouho, pravděpodobně desítky dní. Dále by bylo možné experimetnovat s odstraňováním odlesků i ze skel, což by zjednodušilo tvorbu trénovacího datasetu pro odstraňování odlesků a zrychlilo celý proces.

Techniky pro homogenizaci osvětlení použité v původní implementaci nedosáhly uspokojivých výsledků tak, aby byly ztmaveny pouze přesvětlené části scény. Buď byly tyto části ztmaveny nedostatečně, nebo původně nepřesvětlené části scény zase zůstaly příliš tmavé. Tato technologie také nijak neřešila odlesky. Řešení navržené v této práci se naproti tomu zaměřuje zejména na odlesky a zachovává realistický vzhled odstatních částí vozu, které není třeba upravovat.

Vývoj architektur NeRF neustále postupuje, proto je vhodné tento výzkum sledovat a testovat nově vyvinuté modely na scény s automobily. Zvlášť vhodné je zaměření na architektury, které při zachování kapacity Mip-NeRF 360 a Zip-NeRF optimalizují výpočetní náročnost tak, aby byl systém v praxi snadno použitelný. Dále je možné spojit krok odstraňování odlesků a trénování NeRF a využít modely, které se spolu se scénou samotnou učí i odlesky a mohou je pak ze scény odstraňovat, např. NeRFReN [93], který ale vychází z klasického NeRF a není tak vhodný pro trénování na neohraničených 360° scénách.

Všechny modely pro generování nových pohledů (Mip-NeRF 360, Zip-NeRF a Gaussian Splatting) byly schopny generovat realističtější a méně zašuměné výsledky než Instant NGP použitá v původní implementaci a to i při trénování na neohraničených scénách s pozadím, zatímco ze snímků pro Instant NGP bylo odstraněno pozadí.

Na jednom z datasetů zcela a na některých dalších datastech částečně selhal odhad parametrů

kamer. Tato skutečnost však nebyla kromě vizuální inspekce v grafickém rozhraní COLMAP nijak detekována a i na těchto nevhodných datech byly modely pro generování nových pohledů natrénovány, generovaly však nepoužitelná data. Proto by bylo vhodné zavést verifikaci odhadnutých parametrů kamery s ohledem na apriorní znalost trajektorie, po které by se kamera v dronu nebo v mobilním telefonu měla přibližně pohybovat. Tím bude možné ušetřit mnoho času i finančních prostředků a uživatel bude upozorněn na potřebu pořízení kvalitnějšího datasetu.

Z výsledků postprocessingu je patrné, že je potřeba rozšířit zpracování masky vozu tak, aby byla hladká a nečinila okraje vozidla rozpixelovanými. Dále je potřeba vyvinout algoritmus, který spolehlivě umístí vůz zachycený z libovolného směru do složitější scény obsahující např. stěnu i podlahu tak, aby auto vypadalo, jako by bylo umístěné přesně na zemi v této scéně a tato podoba byla konzistentní vůči pohledům z různých směrů.

Závěr

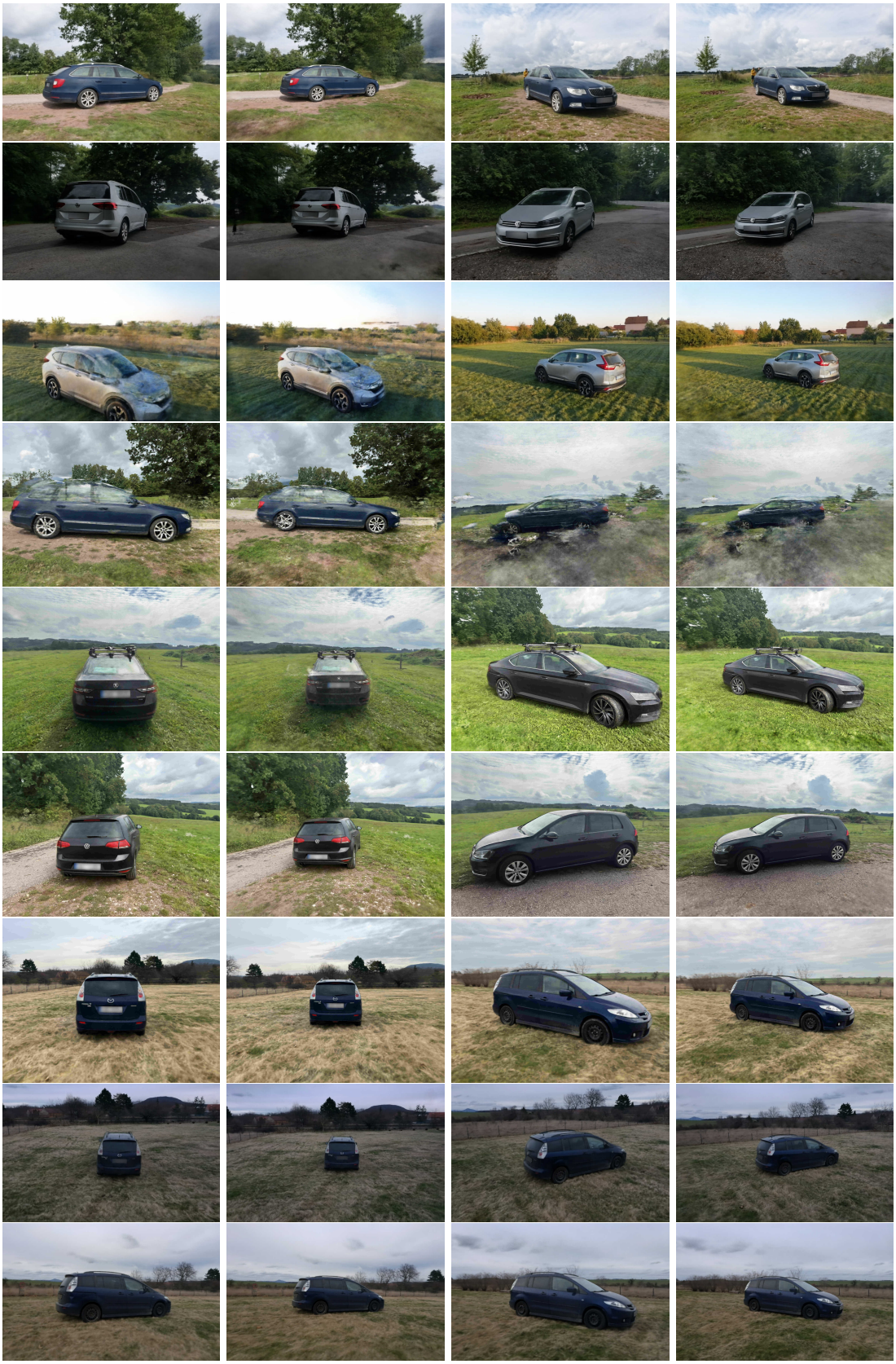
Práce se zabývala problematikou generování nových pohledů na automobily focené v exteriéru a odstraňování odlesků. Byly vytvořeny rozsáhlé datasety pro trénování modelu pro odstranění odlesků z metalízy vozu. Zároveň byly analyzovány, adaptovány a natrénovány dvě pokročilé architektury NeRF a jedna rychlostně optimalizovaná metoda, která reprezentuje objem ve scéně jako 3D gaussiány. Tyto modely byly spojeny do jednoho systému, ve kterém byly nejprve odhadnuty pozice kamer, následně byly ze snímků odstraněny odlesky a nakonec byly natrénovány modely pro generování nových pohledů.

Model pro odstranění odlesků natrénovaný na vytvořeném datasetu byl schopen úspěšně odstranit odlesky slunce, případně samotného okolí vozu. Všechny tři modely pro generování nových pohledů byly schopny generovat realistické nové pohledy na vůz z libovolného úhlu a vzdálenosti za předpokladu kvalitního odhadu parametrů kamer. Nevýhodou těchto modelů je nedostatečná schopnost korekce nesprávně odhadnutých parametrů kamer a absence geometrické verifikace těchto parametrů, která by mohla zabránit zbytečnému trénování na nekvalitních datasetech.

Stanovené cíle práce byly splněny, byly použity pokročilé modifikace technologie NeRF, model pro odstraňování odlesků a modely pro segmentaci. Řešení je součástí projektu aplikovaného výzkumu a předpokládá se jeho využití v praxi. Proto bylo implementováno modulárně a tak, aby bylo snadno nasaditelné v produkčním prostředí.

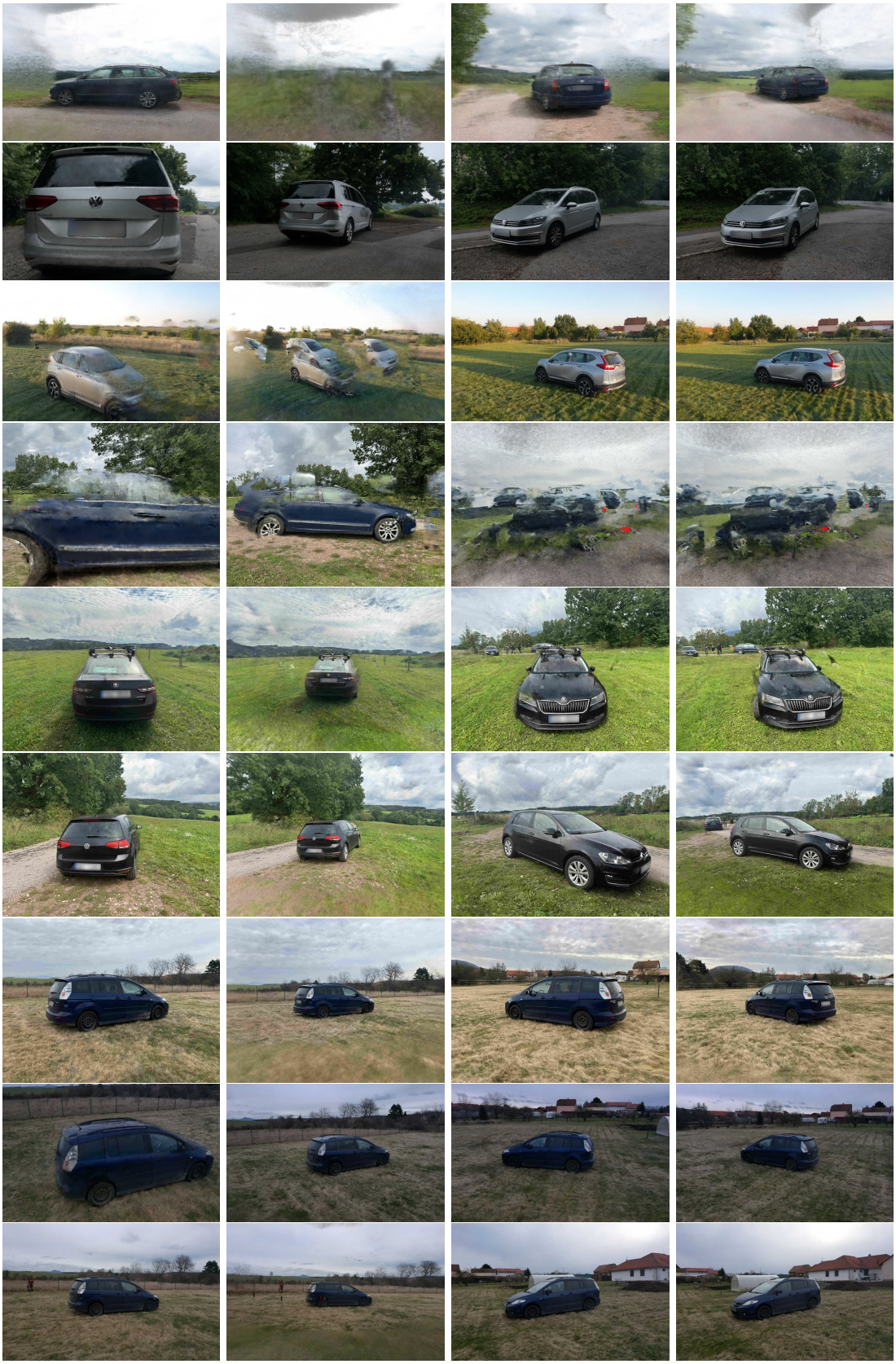
Příloha A

Příloha



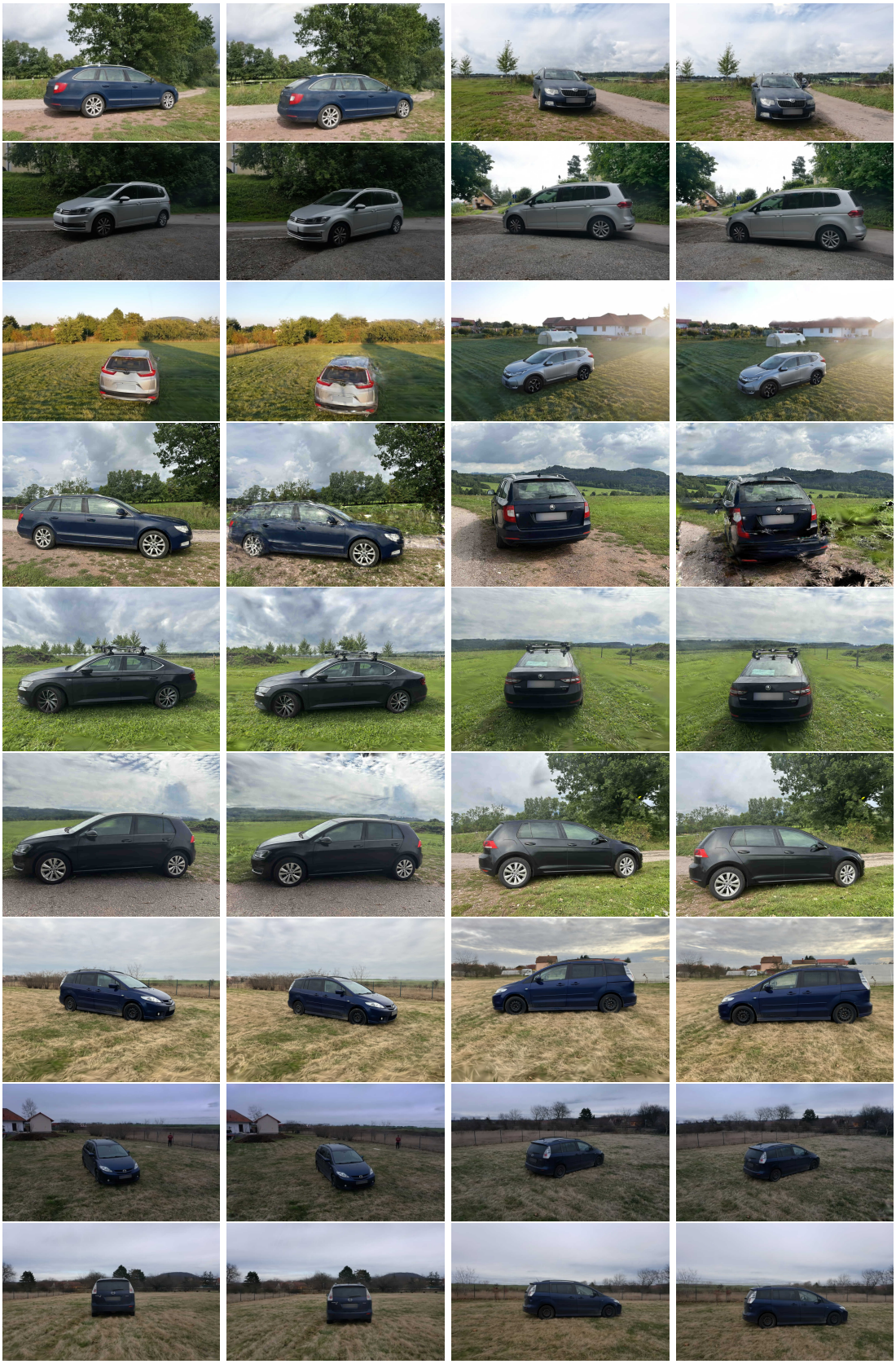
Obrázek A.1 Výsledky modelu Mip-NeRF 360 na reálných datasetech.

Obrázek A.2 Výsledky modelu Mip-NeRF 360 na syntetických datasetech.



Obrázek A.3 Výsledky modelu Zip-NeRF na reálných datasetech.

Obrázek A.4 Výsledky modelu Zip-NeRF na syntetických datasetech.



Obrázek A.5 Výsledky modelu Gaussian Splatting na reálných datasetech.

Bibliografie

1. MILDENHALL, Ben; SRINIVASAN, Pratul P.; TANCIK, Matthew; BARRON, Jonathan T.; RAMAMOORTHI, Ravi; NG, Ren. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. 2020. Dostupné z DOI: 10.48550/arXiv.2003.08934.
2. MEYDENBAUER, Albrecht. Die Photometrographie. *Wochenblatt des Architektenvereins zu Berlin* [https://opus4.kobv.de/opus4-btu/files/749/db186714.pdf]. 1867, č. 14, s. 125–126.
3. LOWE, David G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*. 2004, roč. 60, č. 2, s. 91–110. ISSN 0920-5691. Dostupné z DOI: 10.1023/B:VISI.0000029664.99615.94.
4. BAY, Herbert; TUYTELAARS, Tinne; VAN GOOL, Luc. SURF: Speeded Up Robust Features. In: LEONARDIS, Aleš; BISCHOF, Horst; PINZ, Axel (ed.). *Computer Vision – ECCV 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, s. 404–417. ISBN 978-3-540-33833-8. Dostupné z DOI: 10.1007/11744023_32.
5. ZHANG, Kai; RIEGLER, Gernot; SNAVELY, Noah; KOLTUN, Vladlen. *NeRF++: Analyzing and Improving Neural Radiance Fields*. 2020. Dostupné z DOI: 10.48550/arXiv.2010.07492.
6. BARRON, Jonathan T.; MILDENHALL, Ben; TANCIK, Matthew; HEDMAN, Peter; MARTIN-BRUALLA, Ricardo; SRINIVASAN, Pratul P. *Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields*. 2021. Dostupné z DOI: 10.48550/arXiv.2103.13415.
7. WILLIAMS, Lance. Pyramidal parametrics. *SIGGRAPH Comput. Graph.* 1983, roč. 17, č. 3, s. 1–11. ISSN 0097-8930. Dostupné z DOI: 10.1145/964967.801126.
8. BARRON, Jonathan T.; MILDENHALL, Ben; VERBIN, Dor; SRINIVASAN, Pratul P.; HEDMAN, Peter. *Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields*. 2022. Dostupné z DOI: 10.48550/arXiv.2111.12077.
9. BARRON, Jonathan T.; MILDENHALL, Ben; VERBIN, Dor; SRINIVASAN, Pratul P.; HEDMAN, Peter. *Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields*. 2023. Dostupné z DOI: 10.48550/arXiv.2304.06706.
10. MÜLLER, Thomas; EVANS, Alex; SCHIED, Christoph; KELLER, Alexander. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* 2022, roč. 41, č. 4, 102:1–102:15. Dostupné z DOI: 10.1145/3528223.3530127.
11. KERBL, Bernhard; KOPANAS, Georgios; LEIMKÜHLER, Thomas; DRETTAKIS, George. *3D Gaussian Splatting for Real-Time Radiance Field Rendering*. 2023. Dostupné z DOI: 10.48550/arXiv.2308.04079.
12. ZHANG, Xuaner; NG, Ren; CHEN, Qifeng. *Single Image Reflection Separation with Perceptual Losses*. 2018. Dostupné z DOI: 10.48550/arXiv.1806.05376.

13. WAN, Renjie; SHI, Boxin; LI, Haoliang; HONG, Yuchen; DUAN, Ling-Yu; KOT, Alex C. Benchmarking Single-Image Reflection Removal Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2023, roč. 45, č. 2, s. 1424–1441. Dostupné z DOI: 10.1109/TPAMI.2022.3168560.
14. HU, Qiming; GUO, Xiaojie. Single Image Reflection Separation via Component Synergy. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, s. 13092–13101. Dostupné z DOI: 10.1109/ICCV51070.2023.01208.
15. FAN, Qingnan; YANG, Jiaolong; HUA, Gang; CHEN, Baoquan; WIPF, David. *A Generic Deep Architecture for Single Image Reflection Removal and Image Smoothing*. 2018. Dostupné z DOI: 10.48550/arXiv.1708.03474.
16. SIMONYAN, Karen; ZISSERMAN, Andrew. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. Dostupné z DOI: 10.48550/arXiv.1409.1556.
17. WEI, Kaixuan; YANG, Jiaolong; FU, Ying; WIPF, David; HUANG, Hua. *Single Image Reflection Removal Exploiting Misaligned Training Data and Network Enhancements*. 2019. Dostupné z DOI: 10.48550/arXiv.1904.00637.
18. LI, Yu; LIU, Ming; YI, Yaling; LI, Qince; REN, Dongwei; ZUO, Wangmeng. *Two-Stage Single Image Reflection Removal with Reflection-Aware Guidance*. 2021. Dostupné z DOI: 10.48550/arXiv.2012.00945.
19. YANG, Jie; GONG, Dong; LIU, Lingqiao; SHI, Qinfeng. Seeing Deeply and Bidirectionally: A Deep Learning Approach for Single Image Reflection Removal. In: FERRARI, Vittorio; HEBERT, Martial; SMINCHISESCU, Cristian; WEISS, Yair (ed.). *Computer Vision – ECCV 2018*. Springer International Publishing, 2018, s. 675–691. ISBN 978-3-030-01219-9. Dostupné z DOI: 10.1007/978-3-030-01219-9_40.
20. WEN, Qiang; TAN, Yinjie; QIN, Jing; LIU, Wenxi; HAN, Guoqiang; HE, Shengfeng. Single Image Reflection Removal Beyond Linearity. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, s. 3766–3774. Dostupné z DOI: 10.1109/CVPR.2019.00389.
21. *Reflection Removal — Papers With Code*. [B.r.]. Dostupné také z: https://paperswithcode.com/task/reflection-removal.
22. LIN, Tsung-Yi; MAIRE, Michael; BELONGIE, Serge; BOURDEV, Lubomir; GIRSHICK, Ross; HAYS, James; PERONA, Pietro; RAMANAN, Deva; ZITNICK, C. Lawrence; DOLLÁR, Piotr. *Microsoft COCO: Common Objects in Context*. 2015. Dostupné z DOI: 10.48550/arXiv.1405.0312.
23. DENG, Jia; DONG, Wei; SOCHER, Richard; LI, Li-Jia; LI, Kai; FEI-FEI, Li. ImageNet: A large-scale hierarchical image database. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, s. 248–255. Dostupné z DOI: 10.1109/CVPR.2009.5206848.
24. HE, Kaiming; GKIOXARI, Georgia; DOLLÁR, Piotr; GIRSHICK, Ross. *Mask R-CNN*. 2018. Dostupné z DOI: 10.48550/arXiv.1703.06870.
25. REN, Shaoqing; HE, Kaiming; GIRSHICK, Ross; SUN, Jian. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. Dostupné z DOI: 10.48550/arXiv.1506.01497.
26. RIBANI, Ricardo; MARENGONI, Mauricio. A Survey of Transfer Learning for Convolutional Neural Networks. In: *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*. 2019, s. 47–57. Dostupné z DOI: 10.1109/SIBGRAPI-T.2019.00010.
27. LATKA, Matěj. *Snímání a následná detekce a klasifikace vad skleněných tyčí* [http://hdl.handle.net/10467/95069]. Praha, 2021.
28. REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross; FARHADI, Ali. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. Dostupné z DOI: 10.48550/arXiv.1506.02640.

29. BOCHKOVSKIY, Alexey; WANG, Chien-Yao; LIAO, Hong-Yuan Mark. *YOLOv4: Optimal Speed and Accuracy of Object Detection*. 2020. Dostupné z DOI: 10.48550/arXiv.2004.10934.
30. JOCHER, Glenn. *YOLOv5 by Ultralytics*. 2020. Ver. 7.0. Dostupné z DOI: 10.5281/zenodo.3908559.
31. JOCHER, Glenn; CHAURASIA, Ayush; QIU, Jing. *Ultralytics YOLO*. 2023. Ver. 8.0.0. Dostupné také z: https://github.com/ultralytics/ultralytics.
32. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N.; KAISER, Lukasz; POLOSUKHIN, Illia. *Attention Is All You Need*. 2023. Dostupné z DOI: 10.48550/arXiv.1706.03762.
33. CARION, Nicolas; MASSA, Francisco; SYNNAEVE, Gabriel; USUNIER, Nicolas; KIRILLOV, Alexander; ZAGORUYKO, Sergey. *End-to-End Object Detection with Transformers*. 2020. Dostupné z DOI: 10.48550/arXiv.2005.12872.
34. DOSOVITSKIY, Alexey; BEYER, Lucas; KOLESNIKOV, Alexander; WEISSENBORN, Dirk; ZHAI, Xiaohua; UNTERTHINER, Thomas; DEHGHANI, Mostafa; MINDERER, Matthias; HEIGOLD, Georg; GELLY, Sylvain; USZKOREIT, Jakob; HOULSBY, Neil. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. Dostupné z DOI: 10.48550/arXiv.2010.11929.
35. WU, Junfeng; JIANG, Yi; LIU, Qihao; YUAN, Zehuan; BAI, Xiang; BAI, Song. *General Object Foundation Model for Images and Videos at Scale*. 2023. Dostupné z DOI: 10.48550/arXiv.2312.09158.
36. SCHÖNBERGER, Johannes L.; FRAHM, Jan-Michael. Structure-from-Motion Revisited. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, s. 4104–4113. Dostupné z DOI: 10.1109/CVPR.2016.445.
37. SCHÖNBERGER, Johannes L.; ZHENG, Enliang; FRAHM, Jan-Michael; POLLEFEYS, Marc. Pixelwise View Selection for Unstructured Multi-View Stereo. In: LEIBE, Bastian; MATAS, Jiri; SEBE, Nicu; WELLING, Max (ed.). *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, s. 501–518. ISBN 978-3-319-46487-9. Dostupné z DOI: 10.1007/978-3-319-46487-9_31.
38. FISCHLER, Martin A.; BOLLES, Robert C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*. 1981, roč. 24, č. 6, s. 381–395. ISSN 0001-0782. Dostupné z DOI: 10.1145/358669.358692.
39. TRIGGS, Bill; MCLAUCHLAN, Philip F.; HARTLEY, Richard I.; FITZGIBBON, Andrew W. Bundle Adjustment — A Modern Synthesis. In: TRIGGS, Bill; ZISSERMAN, Andrew; SZELISKI, Richard (ed.). *Vision Algorithms: Theory and Practice*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, s. 298–372. ISBN 978-3-540-44480-0. Dostupné z DOI: 10.1007/3-540-44480-7_21.
40. KAJIYA, James T.; HERZEN, Brian Von. Ray tracing volume densities. *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. 1984. Dostupné také z: https://api.semanticscholar.org/CorpusID:6722621.
41. DENG, Boyang; BARRON, Jonathan T.; SRINIVASAN, Pratul P. *JaxNeRF: an efficient JAX implementation of NeRF*. 2020. Ver. 0.0. Dostupné také z: https://github.com/google-research/google-research/tree/master/jaxnerf.
42. BRADBURY, James; FROSTIG, Roy; HAWKINS, Peter; JOHNSON, Matthew James; LEARY, Chris; MACLAURIN, Dougal; NECULA, George; PASZKE, Adam; VANDERPLAS, Jake; WANDERMAN-MILNE, Skye; ZHANG, Qiao. *JAX: composable transformations of Python+NumPy programs*. 2018. Ver. 0.3.13. Dostupné také z: http://github.com/google/jax.
43. GREENE, Ned; HECKBERT, Paul S. Creating Raster Omnimax Images from Multiple Perspective Views Using the Elliptical Weighted Average Filter. *IEEE Computer Graphics and Applications*. 1986, roč. 6, č. 6, s. 21–27. Dostupné z DOI: 10.1109/MCG.1986.276738.

44. PARK, Keunhong; HENZLER, Philipp; MILDENHALL, Ben; BARRON, Jonathan T.; MARTIN-BRUALLA, Ricardo. *CamP: Camera Preconditioning for Neural Radiance Fields*. 2023. Dostupné z DOI: 10.48550/arXiv.2308.10902.
45. YU, Alex; FRIDOVICH-KEIL, Sara; TANCIK, Matthew; CHEN, Qinhong; RECHT, Benjamin; KANAZAWA, Angjoo. *Plenoxels: Radiance Fields without Neural Networks*. 2021. Dostupné z DOI: 10.48550/arXiv.2112.05131.
46. WANG, Z.; SIMONCELLI, E.P.; BOVIK, A.C. Multiscale structural similarity for image quality assessment. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. 2003, sv. 2, 1398–1402 Vol.2. Dostupné z DOI: 10.1109/ACSSC.2003.1292216.
47. MERRILL, Duane G.; GRIMSHAW, Andrew S. Revisiting sorting for GPGPU stream architectures. In: *Proceedings of the 19th International Conference on Parallel Architectures and Compilation Techniques*. Vienna, Austria: Association for Computing Machinery, 2010, s. 545–546. PACT '10. ISBN 9781450301787. Dostupné z DOI: 10.1145/1854273.1854344.
48. RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. Dostupné z DOI: 10.48550/arXiv.1505.04597.
49. XU, Qingqing; ZHU, Zhiyu; GE, Huilin; ZHANG, Zheqing; ZANG, Xu. Effective Face Detector Based on YOLOv5 and Superresolution Reconstruction. *Computational and Mathematical Methods in Medicine*. 2021, roč. 2021, s. 1–9. Dostupné z DOI: 10.1155/2021/7748350.
50. REIS, Dillon; KUPEC, Jordan; HONG, Jacqueline; DAOUDI, Ahmad. *Real-Time Flying Object Detection with YOLOv8*. 2023. Dostupné z DOI: 10.48550/arXiv.2305.09972.
51. WANG, Chien-Yao; LIAO, Hong-Yuan Mark; YEH, I-Hau; WU, Yueh-Hua; CHEN, Ping-Yang; HSIEH, Jun-Wei. *CSPNet: A New Backbone that can Enhance Learning Capability of CNN*. 2019. Dostupné z DOI: 10.48550/arXiv.1911.11929.
52. LIU, Shu; QI, Lu; QIN, Haifang; SHI, Jianping; JIA, Jiaya. *Path Aggregation Network for Instance Segmentation*. 2018. Dostupné z DOI: 10.48550/arXiv.1803.01534.
53. LIN, Tsung-Yi; DOLLÁR, Piotr; GIRSHICK, Ross; HE, Kaiming; HARIHARAN, Bharath; BELONGIE, Serge. *Feature Pyramid Networks for Object Detection*. 2017. Dostupné z DOI: 10.48550/arXiv.1612.03144.
54. SHAO, Shuai; LI, Zeming; ZHANG, Tianyuan; PENG, Chao; YU, Gang; ZHANG, Xiangyu; LI, Jing; SUN, Jian. Objects365: A Large-Scale, High-Quality Dataset for Object Detection. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, s. 8429–8438. Dostupné z DOI: 10.1109/ICCV.2019.00852.
55. KRISHNA, Ranjay; ZHU, Yuke; GROTH, Oliver; JOHNSON, Justin; HATA, Kenji; KRAVITZ, Joshua; CHEN, Stephanie; KALANTIDIS, Yannis; LI, Li-Jia; SHAMMA, David A.; BERNSTEIN, Michael S.; LI, Fei-Fei. *Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations*. 2016. Dostupné z DOI: 10.48550/arXiv.1602.07332.
56. LI, Feng; ZHANG, Hao; XU, Huaizhe; LIU, Shilong; ZHANG, Lei; NI, Lionel M.; SHUM, Heung-Yeung. *Mask DINO: Towards A Unified Transformer-based Framework for Object Detection and Segmentation*. 2022. Dostupné z DOI: 10.48550/arXiv.2206.02777.
57. VÍTEK, Martin. *Návrh a implementace algoritmů pro sběr a analýzu fotodokumentace vozidel s využitím kamery a neuronových sítí* [http://hdl.handle.net/10467/108940]. Praha, 2023.
58. MITTAL, Anish; MOORTHY, Anush Krishna; BOVIK, Alan Conrad. No-Reference Image Quality Assessment in the Spatial Domain. *IEEE Transactions on Image Processing*. 2012, roč. 21, č. 12, s. 4695–4708. Dostupné z DOI: 10.1109/TIP.2012.2214050.
59. QIN, Xuebin; ZHANG, Zichen; HUANG, Chenyang; DEHGHAN, Masood; ZAIANE, Osmar R.; JAGERSAND, Martin. U2-Net: Going deeper with nested U-structure for salient object

detection. *Pattern Recognition*. 2020, roč. 106. ISSN 0031-3203. Dostupné z DOI: 10.1016/j.patcog.2020.107404.

60. KIRILLOV, Alexander; MINTUN, Eric; RAVI, Nikhila; MAO, Hanzi; ROLLAND, Chloe; GUSTAFSON, Laura; XIAO, Tete; WHITEHEAD, Spencer; BERG, Alexander C.; LO, Wan-Yen; DOLLÁR, Piotr; GIRSHICK, Ross. *Segment Anything*. 2023. Dostupné z DOI: 10.48550/arXiv.2304.02643.
61. GRIWODZ, Carsten; GASPARINI, Simone; CALVET, Lilian; GURDJOS, Pierre; CASTAN, Fabien; MAUJEAN, Benoit; LILLO, Gregoire De; LANTHONY, Yann. AliceVision Meshroom: An open-source 3D reconstruction pipeline. In: *Proceedings of the 12th ACM Multimedia Systems Conference - MMSys '21*. ACM Press, 2021. Dostupné z DOI: 10.1145/3458305.3478443.
62. BLENDER ONLINE COMMUNITY. *Blender – a 3D modelling and rendering package*. Stichting Blender Foundation, Amsterdam: Blender Foundation, 2018. Dostupné také z: http://www.blender.org.
63. DJI. *Support for Mavic 2* [online]. [cit. 2024-04-11]. Dostupné z: https://www.dji.com/cz/support/product/mavic-2.
64. DRONEMADE. *All about DJI's Active Track intelligent drone flight mode* [online]. [cit. 2024-04-11]. Dostupné z: https://www.drone-made.com/post/dji-active-track-mode.
65. LEXYC16. *Porsche 911 (930) Turbo 1975* [online]. [cit. 2024-04-11]. Dostupné z: https://sketchfab.com/3d-models/porsche-911-930-turbo-1975-de1ffd344c41481892511f7fd332c136.
66. LAZERCAR. *Volvo S90 Recharge (Free)* [online]. [cit. 2024-04-11]. Dostupné z: https://sketchfab.com/3d-models/volvo-s90-recharge-free-9462b07c10244fd4a28d86846dc9e3a9.
67. KREUZBERGER, Dominik; KÜHL, Niklas; HIRSCHL, Sebastian. *Machine Learning Operations (MLOps): Overview, Definition, and Architecture*. 2022. Dostupné z DOI: 10.48550/arXiv.2205.02302.
68. PASZKE, Adam; GROSS, Sam; MASSA, Francisco; LERER, Adam; BRADBURY, James; CHANAN, Gregory; KILLEEN, Trevor; LIN, Zeming; GIMELSHEIN, Natalia; ANTIGA, Luca; DESMAISON, Alban; KÖPF, Andreas; YANG, Edward; DEVITO, Zach; RAISON, Martin; TEJANI, Alykhan; CHILAMKURTHY, Sasank; STEINER, Benoit; FANG, Lu; BAI, Junjie; CHINTALA, Soumith. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. Dostupné z DOI: 10.48550/arXiv.1912.01703.
69. NVIDIA CORPORATION & AFFILIATES. *CUDA Toolkit Documentation* [online]. [cit. 2024-04-14]. Dostupné z: https://docs.nvidia.com/cuda/index.html.
70. NVIDIA CORPORATION. *NVIDIA cuDNN* [online]. [cit. 2024-04-14]. Dostupné z: https://developer.nvidia.com/cudnn.
71. NVIDIA CORPORATION. *NVIDIA Driver Downloads* [online]. [cit. 2024-04-14]. Dostupné z: https://www.nvidia.com/download/index.aspx.
72. MERKEL, Dirk. Docker: lightweight Linux containers for consistent development and deployment. *Linux J.* 2014, roč. 2014, č. 239. ISSN 1075-3583.
73. HXBOXY-XH; FERRERA, Maxime. *Under what circumstances will multiple folders be generated under sparse and dense directory (for example, 0, 1, 2, ..., etc.)* [online]. [cit. 2024-04-14]. Dostupné z: https://github.com/colmap/colmap/issues/1507.
74. GIMP DEVELOPMENT TEAM; GIMP WEB TEAM. *GIMP – GNU Image Manipulation Program* [online]. [cit. 2024-04-16]. Dostupné z: https://www.gimp.org.
75. GIMP DEVELOPMENT TEAM; GIMP WEB TEAM. *3.12. Clone* [online]. [cit. 2024-04-16]. Dostupné z: https://docs.gimp.org/en/gimp-tool-clone.html.
76. BROOKS, Justin. *COCO Annotator* [https://github.com/jsbroks/coco-annotator/]. 2019.

77. HUANG, T.; YANG, G.; TANG, G. A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 1979, roč. 27, č. 1, s. 13–18. Dostupné z DOI: 10.1109/TASSP.1979.1163188.
78. LI, Jessica. *Stanford Cars Dataset* [online]. [cit. 2024-04-16]. Dostupné z: https://www.kaggle.com/datasets/jessicali9530/stanford-cars-dataset.
79. KRAUSE, Jonathan; STARK, Michael; DENG, Jia; FEI-FEI, Li. 3D Object Representations for Fine-Grained Categorization. In: *2013 IEEE International Conference on Computer Vision Workshops*. 2013, s. 554–561. Dostupné z DOI: 10.1109/ICCVW.2013.77.
80. ROUGETET, Arnaud. *Landscape Pictures* [online]. [cit. 2024-04-16]. Dostupné z: https://www.kaggle.com/datasets/arnaud58/landscape-pictures.
81. YOUNG, Peter; LAI, Alice; HODOSH, Micah; HOCKENMAIER, Julia. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*. 2014, roč. 2, s. 67–78. Dostupné z DOI: 10.1162/tacl_a_00166.
82. GOOGLE RESEARCH. *MultiNeRF: A Code Release for Mip-NeRF 360, Ref-NeRF, and RawNeRF*. 2022. Dostupné také z: https://github.com/google-research/multinerf.
83. BARRON, Jonathan. *CamP Zip-NeRF: A Code Release for CamP and Zip-NeRF*. 2024. Dostupné také z: https://github.com/jonbarron/camp_zipnerf.
84. GRAPHDECO - INRIA. *3D Gaussian Splatting for Real-Time Radiance Field Rendering*. 2023. Dostupné také z: https://github.com/graphdeco-inria/gaussian-splatting.
85. BENZOIX. *Free Photo — Abstract luxury plain blur grey and black gradient, used as background studio wall for display your products.* [online]. [cit. 2024-04-28]. Dostupné z: https://www.freepik.com/free-photo/abstract-luxury-plain-blur-grey-black-gradient-used-as-background-studio-wall-display-your-products_17601164.htm.
86. SCHONBERGER, Johannes L. *Graphical User Interface – COLMAP* [online]. [cit. 2024-04-15]. Dostupné z: https://colmap.github.io/gui.html.
87. TANCIK, Matthew; WEBER, Ethan; NG, Evonne; LI, Ruilong; YI, Brent; WANG, Terrance; KRISTOFFERSEN, Alexander; AUSTIN, Jake; SALAHI, Kamyar; AHUJA, Abhik; MCALLISTER, David; KERR, Justin; KANAZAWA, Angjoo. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In: *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings*. ACM, 2023. SIGGRAPH '23. Dostupné z DOI: 10.1145/3588432.3591516.
88. HU, Qiming. *DSRNet: Single Image Reflection Separation via Component Synergy (ICCV 2023)*. 2023. Dostupné také z: https://github.com/mingcv/DSRNet.
89. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*. 2010, roč. 88, s. 303–338. ISSN 1573-1405. Dostupné z DOI: 10.1007/s11263-009-0275-4.
90. LI, Chao; YANG, Yixiao; HE, Kun; LIN, Stephen; HOPCROFT, John E. *Single Image Reflection Removal through Cascaded Refinement*. 2020. Dostupné z DOI: 10.48550/arXiv.1911.06634.
91. GATIS, Daniel. *Rembg is a tool to remove images background* [online]. [cit. 2024-04-17]. Dostupné z: https://github.com/danielgatis/rembg.
92. FOUNDATIONVISION. *GLEE: General Object Foundation Model for Images and Videos at Scale*. 2024. Dostupné také z: https://github.com/FoundationVision/GLEE.
93. GUO, Yuan-Chen; KANG, Di; BAO, Linchao; HE, Yu; ZHANG, Song-Hai. *NeRFReN: Neural Radiance Fields with Reflections*. 2022. Dostupné z DOI: 10.48550/arXiv.2111.15234.

Obsah příloh

- `readme.txt` — stručný popis obsahu média
- `car-window-segmentation` — adresář modelu pro segmentaci skel aut
  - `car_window_segmentation.py` — implementace trénování modelu
- `dron` — hlavní adresář s implementací práce
  - `colmap` — adresář modulu `colmap`
  - `data-raw` — adresář pro datasety
  - `data-processed` — adresář se zpracovanými datasety a výsledky
  - `gaussian-splatting` — adresář modulu `gaussian-splatting`
  - `mipnerf360` — adresář modulu `mipnerf360`
  - `postprocessing` — adresář modulu `postprocessing`
  - `reflections` — adresář modulu `reflections`
  - `scripts` — adresář se skripty pro spuštění modulů
    - `example.sh` — příklady spouštění skriptu `run-all.sh`
    - `run-all.sh` — skript pro spuštění všech modulů na jednom datasetu
    - `run-module.sh` — skript pro spuštění jednoho modulu na jednom datasetu
  - `src` — adresář se zdrojovými kódy a checkpointy použitých modelů
    - `DSRNet` — adresář modelu DSRNet
    - `GLEE` — adresář modelu GLEE
    - `gaussian-splatting` — adresář modelu Gaussian Splatting
    - `multinerf` — adresář modelu Mip-NeRF 360
    - `zipnerf` — adresář modelu Zip-NeRF a utility CamP
  - `LICENSE` — soubor s českým a anglickým zněním Prohlášení
  - `README.md` — stručný popis obsahu složky, instrukce k instalaci a spuštění
- `DSRNet` — adresář s implementací DSRNet
  - `car_windows_segmentation.py` — implementace segmentace a ztmavování skel aut
  - `data` — adresář s implementací práce s datasety
    - `sirs_dataset.py` — implementace tvorby datasetů pro trénování DSRNet
  - `test_sirs_custom.py` — implementace testování DSRNet
  - `train_sirs_custom.py` — implementace trénování DSRNet
- `reflection-dataset-creation` — adresář s implementací tvorby datasetu odlesků
  - `algorithm_manual.ipynb` — implementace tvorby datasetu odlesků
- `thesis` — adresář s textem práce ve formátu LaTeX
  - `thesis.pdf` — text práce ve formátu PDF