TransNeRF

Improving Neural Radiance fields using transfer learning for efficient scene reconstruction

Navneet Paul

M.Sc Thesis

University of Twente

UNIVERSITY OF TWENTE.

TransNeRF

Improving Neural Radiance fields using transfer learning for efficient scene reconstruction

by

Navneet Paul

s2273756

M.Sc. Electrical Engineering

Faculty of Electrical Engineering,
Mathematics & Computer Science

Thesis committee

Prof. dr. ir. George Vosselman	Committee chair, UT
Dr. ir. Michael Ying Yang	Academic Supervisor, UT
Dr. ir. C.H. Slump (Kees)	External Examiner
Project Duration:	February, 2021 - October, 2021

UNIVERSITY
OF TWENTE.

Preface

It is a pleasure to submit my master thesis research towards the partial fulfillment of my Masters' degree on Electrical Engineering at the University of Twente. A summary of the accomplished tasks and milestones achieved during the course of my research have been documented in this document. On a personal level, this graduation thesis presented me with multiple challenging tasks, which essentially led to a steep learning curve. My knowledge on topics such as deep learning, 3D reconstruction and computer vision have been sharpened further and I am genuinely satisfied with the overall development in my personality.

I would genuinely like to thank my academic supervisor, Dr. Michael Yang, who has been an immense inspiration and thoroughly supportive throughout my tenure and my academic supervisor. My research thesis would not have been at this certain stage of completion unless for his constant feedback and esteemed guidance. I would like to thank my parents for their prayers and support making me capable enough to take on these challenges . Indeed there were several setbacks faced emotionally and financially along the way due to the pandemic, but I am glad everything fell into its place, which fortunately happened. So it almost seems like someone from above is happy with me for granting me this opportunity to work with and learn from wonderful people and opportunities here in the Netherlands.

Navneet Paul
Enschede, November 2021

Abstract

Neural Radiance Fields (NeRF) achieve impressive view synthesis results for a variety of capture settings including forward-facing capture of bounded and unbounded scenes. We present a transfer learning-based method for neural radiance fields to efficiently synthesize novel views of complex scenes using only a sequence of sample images from the UAVid dataset. We build on transferring the feature color weights of a multilayer perceptron from low resolution images to high resolution scenes by modelling the density and color of the scene as a function of 3D coordinates, latent appearance encodings and viewing directions. Qualitative quantitative metric evaluations are conducted between our proposed approach and other pre-existing NeRF and NeRF++ models, delivering a significant improvement over the later in synthesizing novel scenes at high quality and high fps from very less input number of captured images.

Contents

List of Figures

List of Tables

1 Introduction

1.1. Motivation & Research Statement

Photorealistic representation and rendering of real-world scenes are a highly challenging research topics, with many important applications that range from movie production to virtual and augmented reality. Real-world scenes are notoriously hard to model using classical mesh-based representations, as they often contain structures, semi-transparent objects and topologies that constantly evolves over time due to the often complex scene motion of multiple objects and people [26]. This ultimately demands the rendering of the similar scenes from different freely placed viewpoints in a possibly unbounded scene. This approach of novel view synthesis task is a long-standing problem in computer vision and CGI [8, 13, 25].

Recent deep learning based approaches have led to significant update towards generation of photorealistic novel view synthesis. Neural radiance fields (or NeRFs) is one of the recent state of the art learning methods that are utilize for synthesizing novel views of complex scenes by optimizing an underlying continuous volumetric scene function using a sparse set of input views. In short NeRF is an implicit MLP-based model that maps 5D vectors (3D coordinates plus 2D viewing directions) to output volume density and view-dependent emitted radiance at that spatial location, using fully-connected (non convolutional) deep network, computed by fitting the model to a set of training views. The resulting 5D function can then be used to generate novel views with conventional volume rendering techniques [39]. For instance Fig. 1.1 represents the gist of NeRF approach that optimizes a continuous 5D neural radiance field representation (volume density and view-dependent color at any continuous location) of a scene from a set of hundred input images.

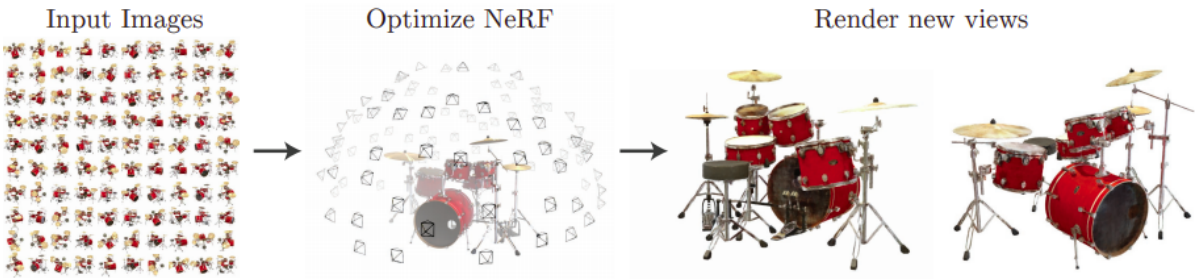

Figure 1.1: Novel View renders from optimized NeRF representation [39]

Since its advent over the past two years, there has been a considerable amount of research being carried out pertaining to novel view synthesis of static and dynamic scenes using NeRF. Pumarola, Albert et al & Park, Keunhong et al in 2020, proposed methods that extended NeRF to dynamic scene domain, allowing to reconstruct and render novel images of real objects under rigid and non-rigid motions from a single camera moving 360° around the scene [46, 73]. Same could be related to the case of the research released by Tianye Li et al in March 2021, on a novel neural 3D view synthesis, which is

able to represent multi-view video recordings of a dynamic real world scene in a compact & expressive representation that enables high-quality view synthesis and motion interpolation [26].These previous works propose approaches to decompose a dynamic scene into a canonical neural radiance field and a set of deformation fields that map observation-space points to the canonical space, thereby enabling them to learn the dynamic scene from images. This has ultimately led to new found applications of NeRFs extended to CGI particularly in 3D character animation and motion. NeRF has been used in reconstruction of animation human models from a multi-view video scenes[50].
In the past, generative adversarial learning approaches such as vid2vid and pix2pixHD approaches have enabled high-resolution image synthesis [82, 3]. But they largely lack an understanding of the 3D world and the image or video generation process because they do not provide precise control over camera viewpoint or object pose. Recent NeRF based approaches employ generative model for radiance fields which have recently proven successful for novel view synthesis of a single scene [63]. This has predominantly led to the applicability of NeRF on satellite imagery. Shadow-NeRF uses a self-supervising learning framework for novel 3D view synthesis from very high spatial resolution optical images taken from known viewing angles [14].
Now that we have established the potential of the concept and application aspects of novel view synthesis using NeRF, we would like to introduce this research, where we develop a framework for 3D scene reconstruction of UAV imagery. Some sample UAV captured imagery are shown below in Figure 1.2.

Figure 1.2: Sample aerial images from public datasets such as UAVid & Aeroscapes, captured using UAVs[33, 43]

1.2. Research Objectives and Expected Outcomes

The primary objective of this research is to develop a new learning framework to improve on the exisiting baseline NeRF models for effecient 3D view synthesis of UAV imagery, with an optimal balance between computational expenses, execution speed and the overall prediction accuracy. Specifically, we perform the below-mentioned tasks in this research:

1 Conceptualize a tranfer learning framework for NeRF model for novel 3D view synthesis.

2 Implement and train the above conceptualization on a targeted UAVid dataset available online [33].

3 Perform a qualitative evaluation of the generated 3D scenes using several standard metrics such as MSE, PSNR, SSIM, etc.

4 Provide a detailed comparison between the existing NeRF models and our proposed method.

1.2.1. RA 1 Conceptualization

Neural radiance fields for novel view syntesis and 3D reconstruction has been addressed using various approaches. Conceptualization hence, can be further broken down into the following:

- Analyze implement the exisiting baseline NeRF based architetures.
- Analyze the shortcomings in the existing architecture and look for possible remedies
- Conduct a background study on the transfer learning.
- Analyze how transfer learning could benefit or improve the existing NeRF architecture for efficient view synthesis.
- Implement transNeRF on the UAVid dataset [33] and analyze the results based on quantitative evaluation between pre-existing models.
- Analyze the perfomance of the proposed model based on ablation studies involving the effect of the input parameters on it's quantitative efficiency.

1.2.2. RA 2 Implementation Aspects

Nowadays a lot of machine learning (or deep learning) libraries are available today such as TensorFlow [35], PyTorch [48]etc. After taking into consideration the ease of programming and architecture building, we decided to move ahead with PyTorch [48], because it has a very consistent programming structure, multi-GPU setups and supportive documentation. Once the required deep learning libraries are decided, we move on to the implementation of basic pre-existing NeRF models on the UAVid dataset[33], thereby determining the shortcomings on the existing models and improvement on the prosposed architecture.

1.2.3. RA 3 Evaluation & Comparisons

Once the proposed NeRF architecture has been trained, it needs to be evaluated based on standard quantitative metrics. This stage involves the comparative performance evaluation between our proposed approach and different state-of-the art baseline approaches, explained along the current literature.

1.3. Research Contributions

After the evaluating differentiating the performance of our proposed approach from the other baseline models using qualitative and quantitative metrics. We propose a model that is efficient in synthesizing high quality novel scenes at a higher frame rate from a minimum sequence of sampled UAV imagery. At the end, we also outline the model's drawbacks limitations in rendering dynamic scenes and transient lighting conditions in the UAV imagery, while also suggesting possible future work that could be pursued for improvement.

1.4. Report Structure

The following document contains 6 chapters. We begin with a detailed related work analysis, and complete each of the above mentioned objectives in a sequential order. The above mentioned strategy complies more or less with the overall structure of the report and the research questions have been answered in respective sections. In the chapter 4 we propose our achitecture/model in detail followed by the experiments and related results based on quantitative and qualitative analysis in chapter 5. Chapter 6 details on the ablation studies conducted on our proposed model to evaluate its efficiency and performance. Towards the end, we present the shortcoming of our proposed idea and suggest certain possible improvements that could be made as a short conclusion for this research.

2

Related Research

2.1. Structure from Motion

With the recent development & availability of lightweight and compact digital cameras along with the continuous development of newer and more powerful algorithms in computer vision has revolutionized the classical aerial photogrammetry approaches leading to a new digital photogrammetry based on a Structure-from-Motion (SfM) approach that provides excellent results in terms of spatial resolution at a low computation cost.
Structure from Motion (SfM) is a relatively new photogrammetric approach that is gaining widespread use for generation of high-resolution mapping products (i.e., point clouds and orthoimages) from imagery acquired with inexpensive, consumer-grade cameras with sufficient endlap and sidelap. The general steps for SfM photogrammetry are shown in Figure 2.1. The processing starts with automatic extraction of key features from the acquired imagery [75]. The extracted features are described in multidimensional descriptors like SIFT [32] which match the extracted features based on the multidimensional maximum likelihood of descriptors [69] and outlier rejection criteria [10]. The procedure is followed by bundle adjustment [77] to simultaneously solve for the intrinsic and extrinsic orientation parameters of the camera to generate a sparse point cloud[75]. The intrinsic orientation (IO) parameters describe the optical characteristics of the camera, such as its focal length, principal point, skew coefficient, and radial and tangential lens distortion coefficients. The extrinsic orientation (EO) parameters are the 3D position and orientation of the camera when the images were acquired.

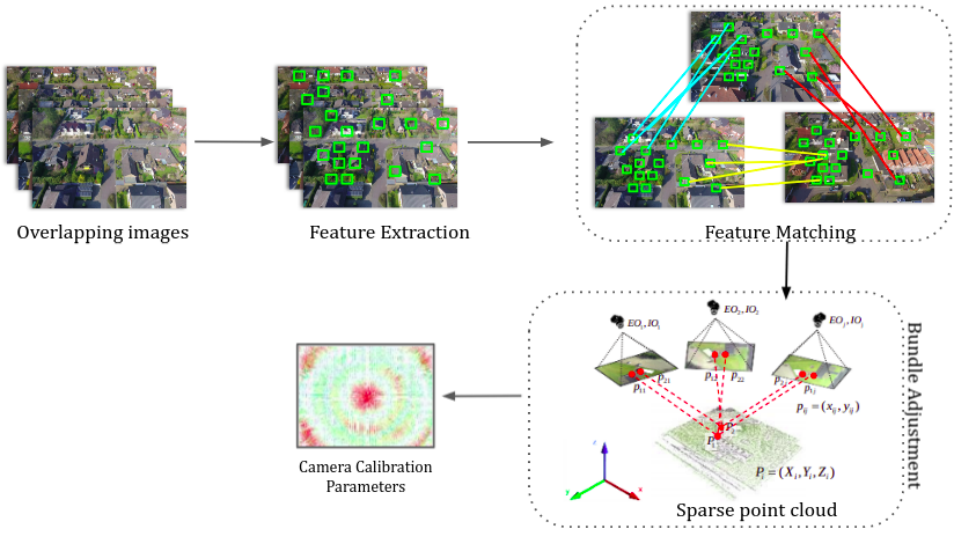

Figure 2.1: SfM Photogrammetry workflow for a set of overlapping image sequence captured using a UAV.[21]

These viewing camera positions are coupled with the ground control points (GCP) or the UAV's global

navigation satellite system (GNSS) for a secondary bundle adjustment, to render the reconstruction as per real world coordinate system The sparse point cloud is densified for rendering the depth maps using an algorithm called multi-view stereopsis (MVS) [15]. Multiview stereopsis generates a depth map for pixels of the image based on photo-consistency of the pixels in the oriented block from the secondary bundle adjustment process, which could be later used for rendering mesh surfaces, digital terrain models (DTMs), and orthorectified imagery [21].

2.1.1. COLMAP

A variety of SfM strategies have been proposed including incremental [69, 2], hierarchical [18], and global approaches [74, 85]. Arguably, incremental SfM is the most popular strategy for reconstruction of sequence of photo collections. A recent work by Schonberger, L et al [62] propose COLMAP an incremental SfM strategy that introduces a geometric verification strategy that augments the scene graph with information subsequently improving the robustness of the initialization and triangulation method that produces significantly more complete scene structure than the state of the art at reduced computational cost (see Figure 2.2).

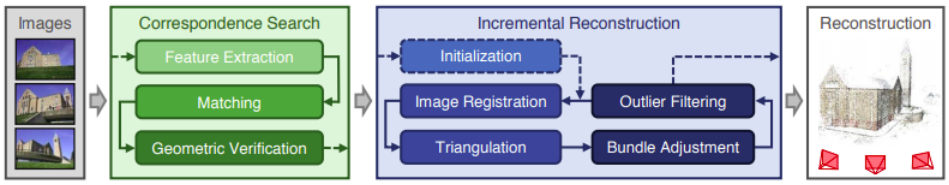

Figure 2.2: COLMAP - an increamental SfM technique for efficient reconstruction [62]

They introduce an iterative bundle adjustment, re-triangulation and outlier filtering strategy that significantly improves completeness and accuracy by mitigating drift effects. The final and most important contribution from the literature is a more efficient bundle adjustment parameterization for dense photo collections through redundant view mining. This results in a system that clearly outperforms the previous state of the art reconstruction techniques in terms of robustness and completeness while preserving its robustness & efficiency.

2.2. Multi Layer Perceptron

One of the recent works in computer vision and more particularly in the domain of 3D scene reconstruction involves encoding scenes or objects in the weights of an MLP that directly maps from a 3D spatial location to an implicit representation of the shape By using a MLP, the researchers leveraged machine learning to have the machine detect the correct function to represent the scene [11]. As input, the model takes in a series of 3D coordinates together to sample the scene throughout, in the format of a fixed-width vector. By taking in these coordinates and producing an RGB and density value along a ray, the model can composite the ray to generate a final pixel value for an image prediction. It then compares images it generates to those that already exist, or “ground truth”. It then uses the difference between its prediction and the ground truth, on a per pixel basis to compute a loss value, or a rough estimate on how well this iteration of the model performed. Through back-propagation and stochastic gradient descent, the model can directly use this loss function to adjust its parameters and create a new iteration of the model that results in a lower loss. MLPs have found their applications recently in mapping from low-dimensional coordinates to colors has also been used for representing other graphics functions such as images [71], textured materials [55, 56], and indirect illumination values [59].

2.3. Neural 3D scene synthesis

Over the recent years, several approaches have been investigated with regard to implicit representation of continuous 3D shapes as level sets by optimizing deep neural networks that map coordinates (xyz) to signed distance functions or occupancy fields [17, 22, 37]. However, these models are limited by their requirement of access to ground truth 3D geometry, typically obtained from synthetic 3D shape datasets such as Pix3D & ShapeNet [72, 7]

Subsequent works have relaxed the requirement of ground truth 3D shapes by formulating differentiable rendering functions that allow neural implicit shape representations to be optimized using only 2D images. Niemeyer et al [42] represent surfaces as 3D occupancy fields and use a numerical method to find the surface intersection for each ray, then calculate an exact derivative using implicit differentiation. Each ray intersection location is provided as the input to a neural 3D texture field that predicts a diffuse color for that point. Sitzmann et al [67] proposed Scene Representation Networks (SRNs), a continuous, 3D structure-aware scene representation that encodes both geometry and appearance. SRNs represent scenes as continuous functions that map world coordinates to a feature representation of local scene properties. A major advantage of this approach of using SRNs is that it can be trained end-end from only 2D images and their camera poses, without access to depth (or shape). Though these studies have been successful in potentially represent complicated and high resolution geometry, they been limited to simple shapes with low geometric complexity, resulting in over-smoothed or sometimes distorted renderings.

2.4. View synthesis & volumetric rendering

Previous studies have shown that, photorealistic novel views could be renderd, given a dense sampling of views utilising sample interpolation techniques [12, 25]. For novel view synthesis with sparse view samplings, there has been a substantial progress in domain of computer vision and graphics research community by predicting traditional geometric appearances and representations from captured images. One of the recent popular class of approaches involves mesh-based representations of scenes with either diffuse [79] or view-dependent appearance [13, 86]. Differentiable rasterizers [29, 31] or path-tracers [27, 44] can directly optimize mesh representations to reproduce a set of input images using gradient descent.
However, gradient-based mesh optimization based on image reprojection is often difficult, likely because of local minima or poor conditioning of the loss landscape. Furthermore, this strategy requires a template mesh with fixed topology to be provided as an initialization before optimization [27], which is typically unavailable for unconstrained real-world scenes.
Volumetric renderings are a more recent approaches that are able to realistically represent complex shapes and materials, are well-suited for gradient-based optimization, and tend to produce less visually distracting artifacts than mesh-based methods. Although early literatures were inclined more towards captured images to directly color voxel grids [64, 76], recent studies have proven to used a sequence of multiple scenes (or from a set of sampled input images) to train deep networks to predict sampled volumetric representation and then use either alpha-compositing [70, 52, 51, 95]to render novel views at test time. Previously, a combination of convolutional neural networks (CNNs) and sampled voxel grids have been utilised for volumetric rendering for specific scenes. In these approaches CNN mostly compensates for discretising from a low resolution voxel grids to a fairly decent resolution output rendering as shown in one of the literatures by Lombardi et al on neural volume rendering depicted in Fig.2.3 [68, 30].

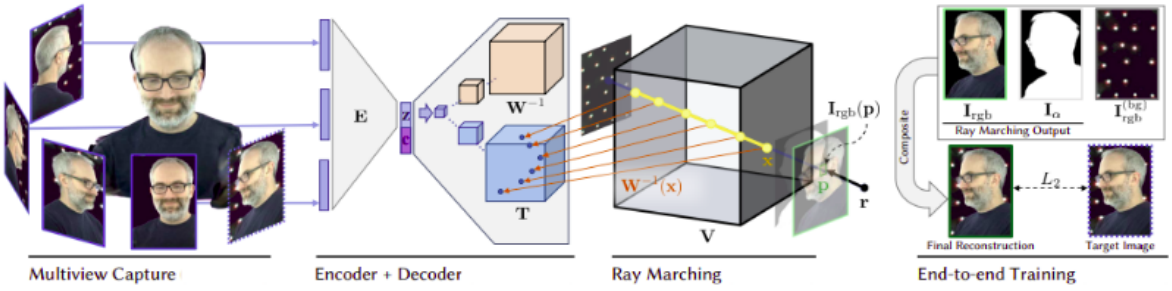

Figure 2.3: In the "Neural Volumes: Learning Dynamic Renderable Volumes from Images" approach, a latent code is decoded into a 3D volume and a new image is then obtained by volume rendering from multiview captured images. [30]

While these volumetric techniques have achieved impressive results for novel view synthesis, their ability to scale to higher resolution imagery is fundamentally limited by time and space complexity due to their discrete sampling. Moreover, rendering higher resolution images requires a finer sampling of

3D space [39]. Neural radiance fields tends to solve this challenge by encoding a continuous volume within the parameters of a deep fully-connected neural network (FCN), thereby producing renderings with significantly higher quality as compared to prior conventional volumetric rendering approaches discussed earlier.

2.5. Neural Radiance Fields & related research

Over the past two years the approach of Neural volumetric rendering or Neural Radiance Fields (NeRF), has garnered significant attention in the computer vision and CGI community [39]. At the backbone of NeRF is an implicit MLP-based model that maps 5D vectors (i.e, 3D coordinates plus 2D viewing directions), to opacity and color values. These 5D representation vectors are later computed by fitting the model to a set of training views. The resulting 5D function can then be used to generate novel views with conventional volume rendering techniques. The Fig. 2.4 depicts the basic overview of the NeRF representation process. In NeRF, the continuous 5D input (x,y,z,θ,ϕ) scene representation is approximated with an MLP network F(Θ) and optimize its weights Θ to map from each input 5D coordinate to its corresponding directional emitted color (i.e, RGB) & volume density (σ). These colors and densities form rays that intersect the object. Volume rendering then composites these values into an actual image. After this image is produced, its rendering loss is computed on a per pixel basis. In the current NeRF literature[39] researchers have adopted two main strategies to reduce this rendering loss and generate high quality renderings, namely positional encoding and hierarchical volume sampling. Positional encoding was mainly adopted to map continuous input coordinates into a higher dimensional space to enable MLP layer to more easily approximate a higher frequency function (i.e, 5D inputs comprising of position and directions)[78]. Heirarchial volumetric rendering, inspired by previous work in volume rendering, was adopted to increase rendering efficiency by allocating samples proportionally to their expected effect on the final rendering. The author(s) suggest utilizing two simultaneously optimize networks: coarse and fine for respective sample locations on the final rendering. The hierarchial volume rendering approach allocates more samples to regions we expect to contain visible content, thereby improving on the quality of final rendered output scene.

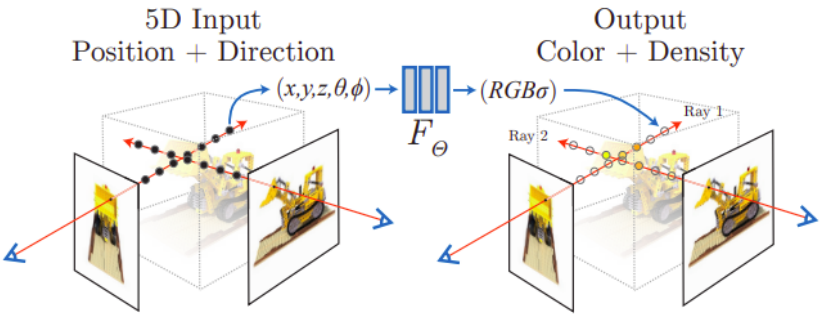

Figure 2.4: In the NeRF approach, synthesized images by sampling 5D coordinates (location and viewing direction) along camera rays are fed into an MLP to produce a color and volume density. Later using volume rendering techniques these composite values are converted into the final rendering. [39]

Arguably, the impact of the NeRF paper lies in its simplicity i.e, just an MLP taking in a 5D coordinate and outputting density and color. However the basic (or vanilla) NeRF model has been open to many recent opportunities to improve upon such as its ability to represent only static scenes, not able to generalize over different scenes (or even objects), while coming at a fairly slow training and rendering speed. More recent works have been instrumental on improving over the baseline NeRF in terms of its performance, speed of training and rendering [28, 58, 93], pose estimation [91], scene composition [41]and relighting [34, 4], dynamic scene rendering [47, 88, 26], etc.

2.6. Recent trends in improving Neural Radiance Fields

As discussed breifly in the previous sections, the capacity of NeRF to model view-dependent appearance, in the absence of regularization, leads to an inherent ambiguity between 3D shape and radiance

that can output degenerate solutions. For any random irregular shape, there is a possibility of radiance fields that perfectly works fine with the training images, but generalizes poorly to render novel test views [93].

2.6.1. Neural scene rendering from unconstrained photo collections

Although neural radiance fields have greatly impacted the 3D reconstruction and rendering, it requires images that are captured in static conditions, minor illumination disturbances and no transient/dynamic objects in scenes [refer Figure 2.5]. In contrast to the conventional NeRF model [39], NeRF in Wild proposed by Martin-Brualla, Ricardo, et al [34], enables failry decent & reliable rendering from unconstrained photo collections. The proposed model separates static and transient objects during the rendering process using a Generative Latent Optimization (GLO) [60] scheme as means to adapt the NeRF [39] to variable lighting and photometric post-processing. Since the transient objects are not certainly located in their current poses in different scenes, their model computes the uncertainty of rays. Then, based on the computed uncertainty, their model focuses less on rays with high uncertainty for rendering transient scenes. A major drawback in the NeRF-in-wild framework, which the authors suggest is the degradation in rendering quality pertaining to areas in the scene that are least observable in the training dataset [34].

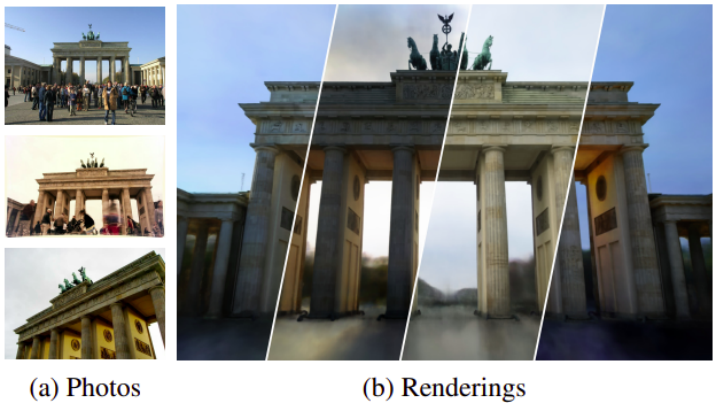

Figure 2.5: (a) Random unstructured photos from internet, (b) Scene renderings using NeRF-W [34]

2.6.2. D-NeRF: NeRF for dynamic scene rendering

Despite achieving an unprecedented level of photorealistic rendered images, NeRF [39] is only applicable to static scenes, where the same spatial location can be queried from different images. The proposed model by Pumarola, Albert, et al [53] aims to resolve that issue where it only requires a single view for each timestamp, indicating it is highly applicable to real-world rendering. They achieve this level of dynamic scene rendering through addition of an additional ”deformation network”. The main function of this deformation network is to predict the positional difference in a specific timestamp with respect to a certain location. With estimated location by the deformation network, the conventional MLP function in the vanilla NeRF [39] predicts colors and volume density. The proposed idea is simple and intuitive since it enables rendering video sequences with a single view for a specific timestamp.

2.6.3. NeRF++

Based on the recent research by Zhang, Kai, et al. [93] in 2020 & Rebain, Daniel, et al [58] on improving neural radiance fields, outlined the major downside in the exisiting NeRF [39] model’s challenge in rendering outdoor scenes due to the ambiguity of setting background depth. In simple terms, NeRF is incapable of rendering unbounded scenes. NeRF++[93] addresses this issue of disambiguity by means of foreground and background seperation, where the authors set the unit sphere to separate the foreground and background of scenes. For the points in the foreground, that is to say, points inside the unit sphere, they follow the same method from the original baseline NeRF research [39]. However, for the points in the background, that is to say, points outside the unit sphere, they reparameterize the coordinate with its distance. As a consequence, the foreground network receives 5-dimensional inputs;

however, the background network receives 6-dimensional inputs. They label this approach as inverted sphere parameterization.

2.6.4. Neural radiance fields for shadow-aware multi-view satellite photogrammetry of aerial images.

Neural radiance fields have found their application extended in the domain of aerial scene reconstruction, more specifically reconstruction of multi-view satellite imagery. Recently Derksen, Dawa, et al [14], proposed a generic scene reconstruction approach ”Shadow-NeRF”, that leverages neural radiance fields to harness non-correlation effects (such as that of shadows) in satellite images and use them as a source of information rather than a source of perturbation for novel 3D views synthesis. For each scene, Shadow-NeRF is trained using very high spatial resolution optical images as input views taken from known viewing angles. The learning process is a self-supervised by an image reconstruction loss & requires no labels or shape priors. To accommodate for transient lighting conditions both from a directional light source (like the sun) and a diffuse light source (like the sky), Shadow-NeRF extend the NeRF approach in two ways, firstly by modelling the direct illumination from the sun using a local light source visibility field and then learning the indirect illumination from a diffuse light source (sky) as a non-local color field function of the position of the sun. Combinating these two factors during the volume rendering stage reduces the altitude and color errors in shaded areas as compared to just utilizing the baseline NeRF model[39]. Figure 2.6 shows the S-NeRF architecture as per the literature [14].

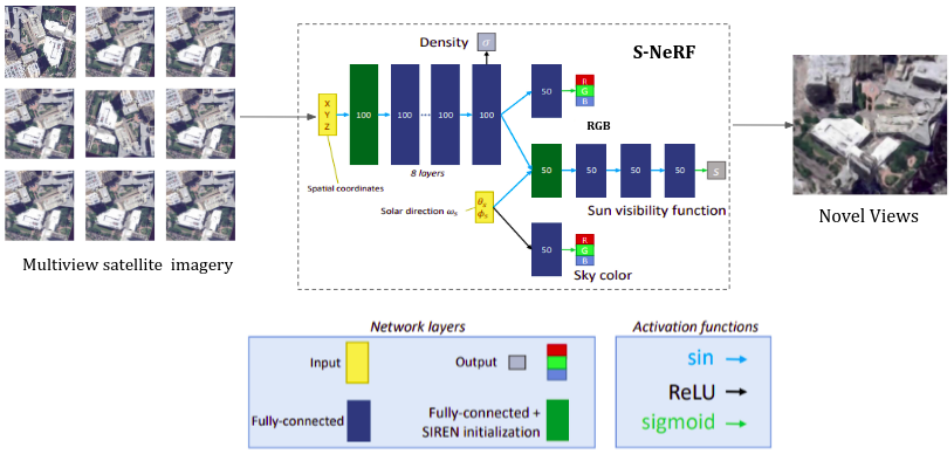

Figure 2.6: Shadow-NeRF samples multiview satellite imagery for synthesizing novel 3D scenes [14].

3

Transfer Learning

3.1. Preview

Over the past many years, traditional machine learning approaches have achieved great success and have been successfully applied in many practical applications. But they are still confined to certain limitations for certain real-world scenarios. The ideal scenario of a machine learning approach is that there are abundant labeled training instances, which have the same distribution as the test data. However, collecting sufficient training data is often expensive, time-consuming, or even unrealistic in many scenarios. Semi-supervised learning [96] or reinforcement learning [84] approaches can partly solve this problem by relaxing the need of mass labeled data. Typically, a semi-supervised approaches [97] only requires a limited number of labeled data, and it utilizes a large amount of unlabeled data to improve the learning accuracy, where as reinforcement learning does not rely on any particular labelled dataset and functions on a state-action-reward space [84, 23]. But in many cases, unlabeled instances are also difficult to collect, which usually makes the resultant traditional models unsatisfactory[16].
Transfer learning focuses on "*transferring the knowledge across domains*, has been a promising machine learning method that's been used for solving the above challenges faced by the conventional approaches over the recent years. The main science behind transfer learning comes from the human psycology, where according to the "*generalization theory of transfer*" proposed by psychologist C.H. Judd, suggests that learning to transfer is the result of the generalizing experiences. It is possible to realize the transfer from one situation to another as long as a person generalizes his experience. According to this theory, the prerequisite of transfer is that there needs to be a connection between two learning activities. Machine learning researchers take their insights from the classical psychology to formulate transfer learning categories/approaches in the recent machine learning applications.

3.2. Categorizing transfer learning

Transfer learning could be classified into different categories. For example, transfer learning problems can be divided into three categories, i.e., transductive, inductive, and unsupervised transfer learning [45]. These three categories can be interpreted from a supervised or label-setting aspect. Transductive transfer learning refers to the situations where the label information only comes from the source domain. Inductive transfer learning involves a scenario if the label information of the target domain instances is available. If the label information is unknown for both the source and the target domains the situation is known as unsupervised transfer learning.
According to the survey conducted by Pan, S. J., Yang, Q [45], transfer learning approaches can be classified into four groups: instance-based, feature-based, parameter-based, and relational-based approaches. Instance-based transfer learning approaches are mainly based on the instance weighting strategy. Feature-based approaches transform the original features to create a new feature representation; they can be further divided into two subcategories, i.e., asymmetric and symmetric feature based transfer learning. Asymmetric approaches transform the source features to match the target ones. Symmetric approaches attempt to find a common latent feature space and then transform both the source and the target features into a new feature representation. The parameter-based transfer

learning approaches transfer the knowledge at the parameter level. Relational-based transfer learning approaches mainly focus on the problems in relational domains. Such approaches transfer the logical relationship or rules learned in the source domain to the target domain. Fig. 3.1 shows the categorization of the transfer learning process.

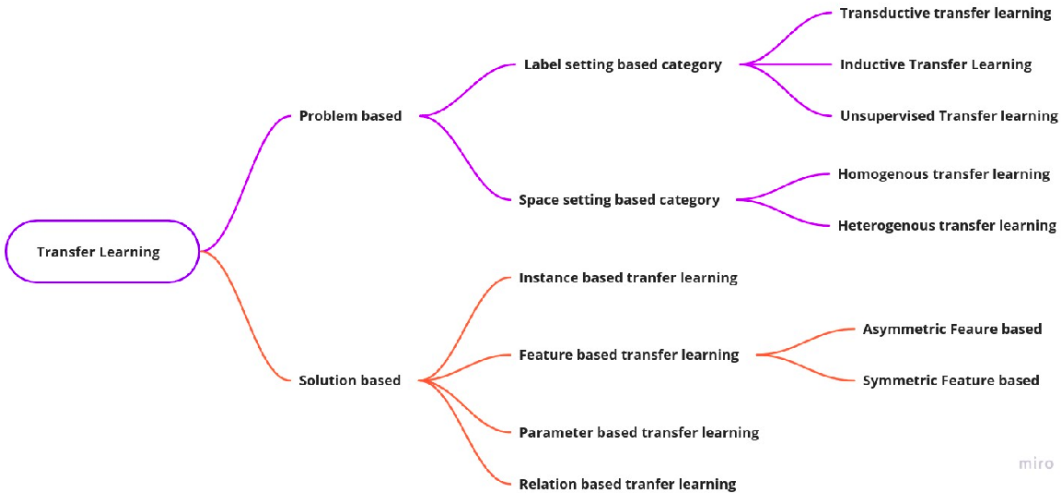

Figure 3.1: Classifying transfer learning approaches [45]

For our current research we are focusing more towards a feature based transfer learning approach which is mainly used in applications involving computer vision and visual categorizations [80, 66].

3.3. Transfer Learning in computer vision applications

Transfer learning is a popular method in computer vision because it allows us to build accurate models in a time saving way[57]. With transfer learning, instead of starting the learning process from scratch, you start from patterns that have been learned when solving a different problem. This way you leverage previous learning and avoid starting from scratch. In computer vision, transfer learning is usually expressed through the use of pre-trained models. Accordingly, due to the computational cost of training such models, it is common practice to import and use models from published literature like VGG, Inception, MobileNet, etc, made possible thanks to convolutional neural networks (CNNs). As an example for a use case for transfer learning in object detection using CNNs shown in Fig. a, the lower layers are typically responsible for detecting patterns like lines and edges, the middle ones learn filters that detect parts of objects, while the last layers learn to recognize full objects, in different shapes and positions. The knowledge gained may be reused in the similar problem domain using transfer learning [92, 6]. The concept of transfer learning is to use most of the layers from a pre-trained model and retrain only a final few layers for different tasks or adapting to a new domain of tasks.

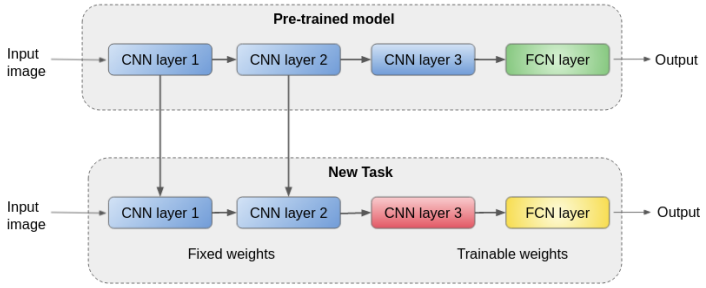

Figure 3.2: Schematic view of transfer learning approach using convolution neural networks [6]

These type of network based transfer learning approaches (involving CNNs) have been mainly focused

on two aspects/strategies:

- **Feature extraction** - training a new classifier on top of the pre-trained base model. In this method we leave the weights learned by convolution layers unchanged and train only the last, fully connected layer. It is a fast, simple, but still quite effective way to use ready-made architecture.

- **Fine-tuning** - where we retrain not only the fully connected layer (FCN), by also adjusting one or more convolution layers. In this strategy we unlock minimal layers of a base model and train both the newly-added classifier and the last few layers of the base model. The weights from the original training are treated as the initial learning point. Unlocked convolution layers are not trained from the beginning, but only tuned to a new task. This method is used to improve model's performance. A major drawback is that it can sometimes lead to overfitting and sometimes time-consuming.

The effectiveness of convolutional neural networks has been proven in many computer vision problems due to their powerful feature representation. Complete algorithms, used in many of our researches, require only simple data preprocessing and augmentation. It is then followed by re-training final layers of existing model, according to transfer learning methodology [6].

3.3.1. GANs & transfer learning

Deep learning applications on computer vision involve the use of large-volume and representative data to obtain state-of-the-art results due to the massive number of parameters to optimise in deep models. Generative Adversarial Networks (GANs) being one of the prominent deep learning approaches, have been extremely successful in various application domains such as computer vision, medicine, and natural language processing since their introduction in 2014 [19]. Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset. GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated), shown in the Figure.3.3 below. The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples [65].

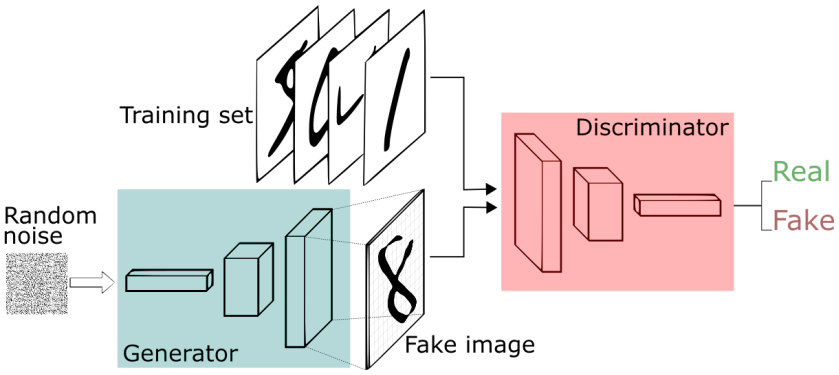

Figure 3.3: The generative adversarial learning framework proposed by Goodfellow, Ian, et al [19]

3.3.2. Transfer learning with GANs

Most of the recent applications of GANs have been centered over the synthesizing highly realistic scenes using image-to image translation [65, 81, 1, 82] and 3D reconstruction of objects [90, 87]. Wang, Ting-Chun, et al in their previous works have introduced feature based transfer learning frameworks using conditional GANs [40] for synthesizing high resolution photo-realistic images from semantic and instance label maps. They propose coarse-to-fine generator architectures and multi-scale discriminator models which are suitable for conditional image generation from lower resolutions to higher resolutions [81]. The coarse to fine generator architecture comprises of two generator networks- one for lower

resolution (G1) scenes and another one for handling high resolution scenes (G2). During the course of training the input scenes are downsampled to lower resolutions and fed them into the low resolution network (G1). Features from the last feature layer of G1 are transfered to the intermediate feature layer of G2. These summed features are then fed into another series of residual blocks to output the higher resolution images [see Fig.3.3] [82].

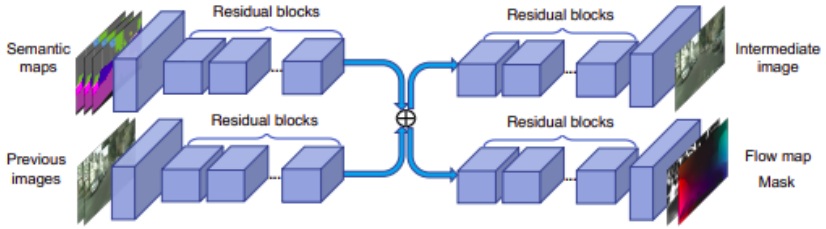

Figure 3.4: The proposed generative adversarial learning approach - vid2vid for synthesizing high resolution simulated scenes, by Wang, Ting-Chun, et al [82]

4

TransNeRF

We begin this chapter by introducing transfer learning for improving neural scene rendering and explaining every component of our proposed ”TransNeRF” architecture. This section details on the overview of our model, the pretraining stage and the final fine tuning aspect for scene rendering. Fig. h represents the overview of our model. Initially in pretraining stage, we train a coordinate-based MLP function which is similar to NeRF++ model [93] on unstructured low resolution scenes from the UAVid dataset and obtain the pretrained model parameter optimized for generalization, which could be useful while training model for higher resolution scenes. We chose to use two NeRF model as our base for pretraining since it offers an effective solution that separately models foreground and background, addressing the challenge of modeling unbounded 3D scenes using an inverted sphere scene parameterization. Also when compared to the basic NeRF model manages to resolve the shape-radiance ambiguity, as well as a remedy for the parameterization of unbounded scenes in the case of 360° UAV imagery.

4.1. Proposed architecture

During the pre-training process the captured images from the UAVid dataset are downsampled at a lower resolution from 4K to batches of 512x256, 1024x512 and 2048x1024 resolutions. The proposed model (shown in Fig. 4.1), primarily comprises of the following units Generative Latent Optimizer(GLO) and the NeRF model.

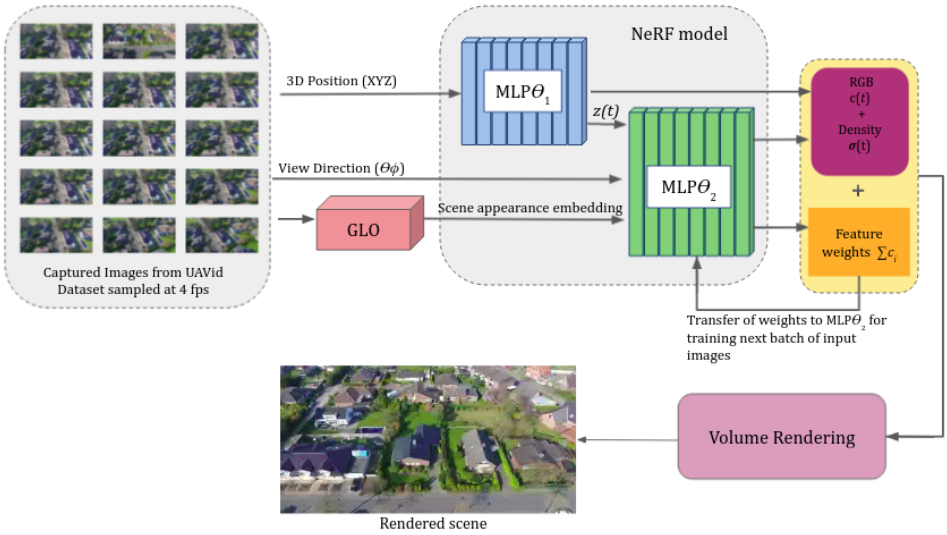

Figure 4.1: The pretraining process block of the proposed TransNeRF model with the NeRF++ model comprising of the two MLP units (MLPθ_1 & MLPθ_2). Here the sequence of captured images are downsampled at a lower resolution (say 512X256) while training. GLO stands for Generative Latent Optimization.

4.2. Generative Latent Optimization (GLO) for photometric variation

In aerial photography, time of day and atmospheric conditions directly impact the illumination (and consequently, the emitted radiance) of objects in the scene. This issue is exacerbated by photographic imaging pipelines, as variation in auto-exposure settings, tone-mapping, etc, across photographs may result in additional photometric inconsistencies [89]. To account for these photometric inconsistencies & adapt our TransNeRF model to outdoor lighting variations, we use a Generative Latent Optimization strategy (proposed by Bojanowski, Piotr, et al[5]).

GLO is a generative model which enjoys many of the desirable properties of GANs [19] including modeling data distributions, generating realistic samples, interpretable latent space, but more importantly, it doesn't suffer from unstable adversarial training dynamics. GLO trains a deep convolutional generators (proposed by Salimans, Tim, et al [61]) using simple reconstruction losses to account for these photometric discripancies & encoding scene appearance embeddings. Using these scene appearance embeddings as input to only the branch of the network that emits color grants our model the freedom to vary the emitted radiance of the scene in a particular image while still guaranteeing that the 3D geometry (predicted earlier by MLPθ_1) is static and shared across all images.

Alternatively, GLO could be modelled as an auto-encoder where the latent representation is not produced by a parametric encoder, but learned freely in a non-parametric manner. In contrast to GANs, GLO tracks the correspondence between each learned latent vector and the image that it represents. Hence, the goal of GLO is to find a meaningful organization of the latent appearance vectors, such that they can be mapped to their target images. Figure 4.2 shows the schematic overview of the GLO framework suggested by Bojanowski, Piotr, et al[5]).

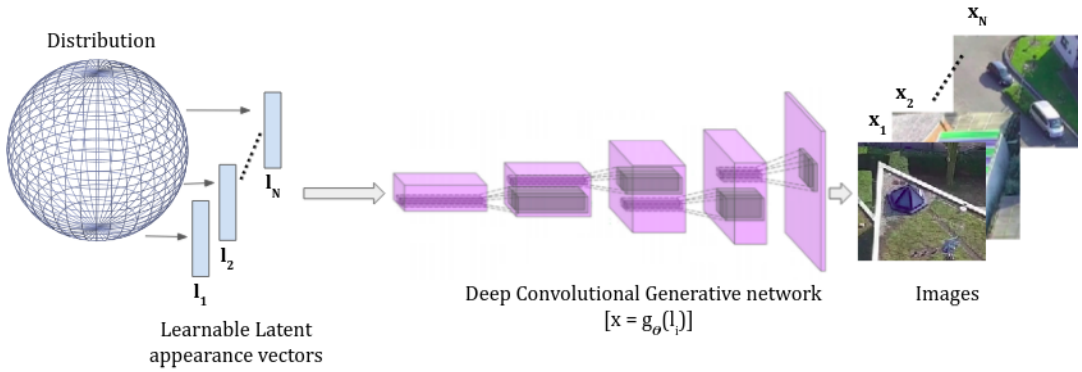

Figure 4.2: Generative Latent Optimization model utilizes a deep convolutional generator to map latent appearance features to the target images[5]

.

GLO learns the parameters $\theta \in \Theta$ of a generator $g_\theta : \mathcal{L} \rightarrow X$ with optimal latent appearance vectors $l_i \in \mathcal{L}$ for each image $x_i \in \mathcal{X}$, by solving the following equation:

$$\min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \left[\min_{l_i \in \mathcal{L}} L\left(g_\theta\left(l_i\right), x_i\right) \right] \tag{4.1}$$

where L is the associated loss function measuring the reconstruction error from $g(l)_i$ to x_i. While using GLO we jointly optimize the inputs latent appearance vectors along with the model parameters θ.

4.3. Neural Radiance Fields model

NeRF represents a scene using a learned, continuous volumetric radiance field Fθ defined over a bounded 3D volume. Fθ is modeled using a multilayer perceptron (MLP) that takes as input a function of 3D coordinate positions (*x, y, z*) and normal viewing directions as **d** = (*dx, dy, dz*), and produces as output a density σ and color **c** = (*r, g, b*). The camera ray (**r**), emitted from the center of projection of a camera **o** through a given pixel on the image plane could be expressed as a function of the viewing

direction: **r=o+*t*d**. Based on this, NeRF is available to apporximate the expected color function ($\hat{C}$(r)) on that pixel through:

$$\hat{\mathbf{C}}(r) = \mathcal{R}(r, c, \sigma) = \sum_{k=1}^{K} T\left(t_k\right) \alpha\left(\sigma\left(t_k\right) \delta_k\right) c\left(t_k\right) \tag{4.2}$$

$$\text{where } T\left(t_k\right) = \exp\left(-\sum_{k'=1}^{k-1} \sigma\left(t_{k'}\right) \delta_{k'}\right) \tag{4.3}$$

$\mathcal{R}(r, c, \sigma)$ represents the volume rendering of color c with density σ. $\mathbf{c}\left(t_k\right)$ & $\sigma\left(t_k\right)$ denotes the color and density at point r(t). $\delta_k = t_{k+1} - t_k$ is the distance between two quadrature points or adjacent samples and exponential function is used $\alpha(x) = 1 - \exp(-x)$. NeRF architecture utilizes a stratified sampling approach to select the adjacent sample points between the near and far planes of the camera. While the conventional approach of neural volume rendering requires only a single MLP layer with ReLU activation for representing both color (RGB) volume density functions [39]. In our apporach, NeRF model represents the volumetric density σ(t) and color c(t) using two simultaneous ReLU based MLPs with parameters θ=[θ_1,θ_2] as shown in equations below:

$$c_i(t) = \text{MLP}_{\theta_2}\left(z(t), \gamma_d(d), \ell_i^{(a)}\right) \tag{4.4}$$

$$[\sigma(t), z(t)] = \text{MLP}_{\theta_1}\left(\gamma_x(r(t))\right) \tag{4.5}$$

γ_d, γ_x denote the encoding functions of viewing directions and position respectively. To fit parameters θ, the NeRF models minimize the sum of squared reconstruction errors with respect to an RGB image collection, where each image I_i is paired with its corresponding intrinsic and extrinsic camera parameters which can be estimated using structure from motion [62].

4.3.1. 3D scene generating network

The positional encoding of the input location (γ_x) is passed through 8 fully-connected ReLU layers, each with 256 channels. A skip connection is included that concatenates this input to the fifth layer's activation. An additional layer outputs the volume density σ (which is rectified using a ReLU to ensure that the output volume density is non negative) and a 256-dimensional feature vector. A final layer with a sigmoid activation outputs the emitted RGB radiance at position x along with $z(t)$. Figure 4.2 details on the network architecture of our first submodel: MLP_{θ_1}.

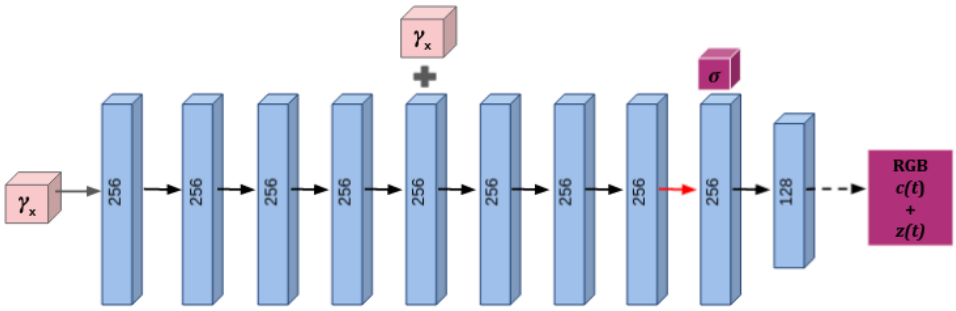

Figure 4.3: A visualization of our fully-connected MLP_{θ_1} network architecture. Input vectors are shown in pink, intermediate hidden layers are shown in blue, output vectors are shown in red, and the number inside each block signifies the vector's dimension. All layers are standard fully-connected layers, black arrows indicate layers with ReLU activations, red arrow indicate layers with no activation, dashed black arrows indicate layers with sigmoid activation, and "+" denotes additional vector concatenation.

4.3.2. Color emmiting network

The secondary NeRF unit (MLP_{θ_1}) takes viewing direction encoding (γ_d), $z(t)$ and latent scene appearance embedding ($\ell_i^{(a)}$), from the generative latent optimization strategy [5]) as inputs with 8 fully-connected layers) to output the parameters that store the encoding features pertaining to sampled color function (from equation 4.1) which could be expressed sum of all sampled colors c_i. The directional

encoding of the camera views (γ_d) is passed through 8 fully-connected ReLU layers, each with 256 channels. Similar to the MLP$_{\theta_1}$ model a skip connection is introduced here that concatenates this input to the fifth layer's activation. Towards the final layer of the network a sigmoid activation outputs the feature weights of encoded sampled color functions c_i. The architecture of the MLP$_{\theta_2}$ is depicted in Fig. 4.3. In our model, we try to leverage a transfer of these learnt weights (or feature color encodings) as inputs along with encoded viewing directions latent appearance embedding to MLP$_{\theta_2}$ the while training the next batch of captured image sequences of high resolution. These appearance embeddings and feature weights (from the previous batch of low resolution images) are used as input to the branch of the model that emits color grants MLP$_{\theta_2}$ network, the freedom to vary the emitted radiance of the scene in a particular image while still guaranteeing that the 3D geometry (predicted earlier by MLP$_{\theta_1}$) is static and shared across the sequence of images. The color emitting function from equation 4.4, with considerations to our new inputs could be transformed as shown in equation 4.6.

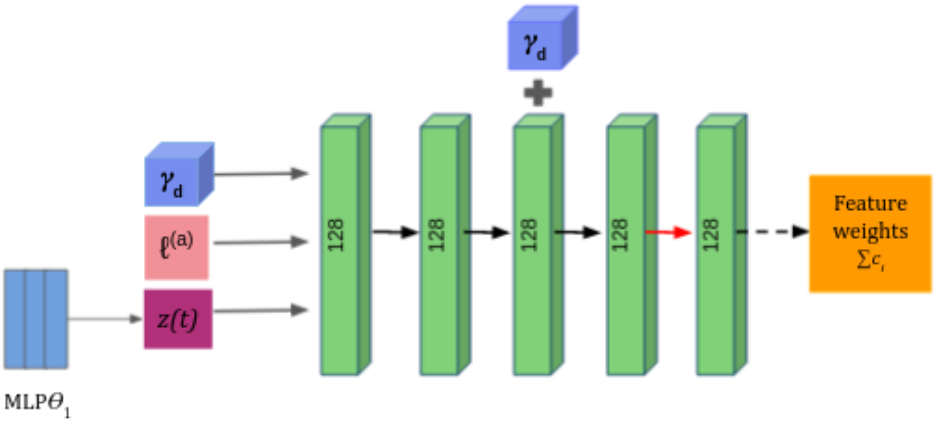

Figure 4.4: A visualization of our fully-connected MLP$_{\theta_2}$ network architecture. Input direction vectors (γ_d) shown in blue, intermediate layers are shown in green, the latent appearance embedding ($\ell_i^{(a)}$) from the GLO is depicted in pink box, output feature weights are shown in orange box and the number inside each block signifies the vector's dimension. All layers are standard fully-connected layers, black arrows indicate layers with ReLU activations, dashed black arrows indicate layers with sigmoid activation, and "+" denotes additional vector concatenation.

$$c_{i+1}(t) = \text{MLP}_{\theta_2}\left(z(t), \gamma_d(d), \ell_{i+1}^{(a)}, c_i\right) \tag{4.6}$$

4.4. Optimizing TransNeRF

In the previous sections we discussed about the the core components necessary for modeling a scene as a neural radiance field and rendering novel views from this representation. However, we observe that these components are not sufficient for achieving state-of-the-art quality. First we adopt a positional encoding of the input coordinates that assists the MLP in representing high-frequency functions, and the second is a hierarchical sampling procedure that allows us to efficiently sample this high-frequency representation.

4.4.1. Positional Encoding

Over the years researchers have found that having the network F_θ directly operate on position (xyz) & direction ($\theta\phi$) input coordinates results in renderings that perform poorly at representing high-frequency variation in color and geometry while using function approximators such as neural networks [20]. Previous research in this domain suggests that deep neural networks are biased towards adapting to low frequency functions which contain minimal amount of inputs [54]. They additionally show that optimizing the inputs to a higher dimensional space using high frequency functions before passing them to the network enables better fitting of data that contains high frequency variation. For our current study we leverage a similar approach as explained in the previous NeRF literature findings in the context of neural scene representations, where F_θ could be reformulated as a composite function of F_{mlp} and $\gamma(p)$. $\gamma(p)$ represents the mapping function that encodes features from lower dimensional space (R) to a

higher one (R^{2L}). We consider the encoding function suggested in the previous NeRF architecture[39], shown below. F_{mlp} is a regular MLP network.

$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right) \tag{4.7}$$

This function $\gamma(p)$ is applied separately to each of the three coordinate values in **x** (after they are normalized to lie in [−1, 1]) and the three components of viewing directions in the cartesian viewing direction **d**. We use these positional encoding functions to map continuous input coordinates into a higher dimensional space to enable our F_{mlp} to more easily approximate a higher frequency function. Removing the positional encoding drastically decreases the model's ability to represent high frequency geometry and texture, resulting in an oversmoothed appearance, as shown in Figure 4.5.

Figure 4.5: We show the effectiveness of employing positional encoding to improve our model's ability to represent high frequency geometry and texture, resulting in a fine realistic appearance (seen in right-side images).

4.4.2. Heirarchial Volume rendering

Although we use a discrete set of samples to estimate the integral, stratified sampling enables us to represent a continuous scene representation because it results in the MLP being evaluated at continuous positions over the course of optimization. We use these samples to estimate $C(r)$ with the quadrature function (equation 4.7) discussed in previous volume rendering literatures [36].

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) \mathbf{c}_i, \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \tag{4.8}$$

Where $\delta_i = t_{i+1} - t_i$ denotes the distance between adjacent samples or scenes. Previous rendering strategies of densely evaluating the neural radiance field network at N query points along each camera ray are inefficient as free space and occluded regions are still sampled repeatedly that do not contribute to the rendered image. Neural Radiance fields take inspiration from the previous works on neural volume rendering [24], by adopting a hierarchical scene representation that increases rendering efficiency by allocating samples proportionally to their expected effect on the final rendering. During the volume rendering phase, our model simultaneously optimizes two networks: *coarse* and *fine*.

We first sample a set of N_c locations with the RGB [$c(t)$] and density [$\sigma(t)$] outputs from the proposed NeRF model, using stratified sampling, and evaluate the "coarse" network at these locations. Given the output of this "coarse" network, we then produce a more informed weighted sampling of points along each ray where samples are biased towards the relevant parts of the volume as expressed in equation 4.8. Using these weighted colors we compute the final rendered color of the ray for the coarse samples $\hat{C}_c(r)$.

$$\hat{C}_c(r) = \sum_{i=1}^{N_c} w_i c_i, \quad w_i = T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) \tag{4.9}$$

The weights are normalized using $\hat{w}_i = w_i / \sum_{j=1}^{N_c} w_j$ to produce a Probability Density Function (PDF). We sample a second set of N_f locations from this distribution using inverse transform sampling, evaluate our "fine" network at the union of the first and second set of samples, and compute the final rendered color of the ray using the equation 4.7. An important note: we consider all the samples while computing the final rendered ray color, i.e, N_c+N_f, at this stage. This is done to ensure that more samples are allocated to regions we expect to contain visible content. In our approach we use the sampled values as a non-uniform discretization of the whole integration domain rather than treating each sample as an independent probabilistic estimate of the entire integral similar to the one used in the original NeRF approach [39].

4.5. Final volume rendering & Loss function

We optimize a separate neural continuous volume representation network for each scene. The only requirement is for a dataset of captured RGB images of the scene, the corresponding camera poses and intrinsic parameters, and scene bounds we use ground truth camera poses, intrinsics, and bounds for synthetic data, and use the COLMAP structure-from-motion package [62] to estimate these parameters for the captured images. At each optimization iteration, we randomly sample a batch of camera rays from the set of all pixels in the dataset, and then follow the hierarchical sampling for sampling N_c coarse samples from coarse network and N_c+N_f fine samples from fine network (described in section 4.4.2). The final step is the volume rendering using computed rays from each ray of the two samples using the function in the equation 4.9.

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right) \tag{4.10}$$

Where $C(r)$ is the expected color of the camera ray $\mathbf{r}(t)$=$\mathbf{o} + td$. **d** is the viewing direction values. $T(t)$ is the transmittance probability along the ray from t_n to t, i.e, the probability that the ray travels from tn to t without hitting any other particle. **c** is the RGB vector information and σ is the computed density vector.

4.5.1. Loss function

We estimate our loss function based on the the total squared error between the rendered and true pixel colors for both the coarse and fine samples. The loss function is shown below in equation 4.10:

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \left[\left\| \hat{C}_c(r) - C(r) \right\|_2^2 + \left\| \hat{C}_f(r) - C(r) \right\|_2^2 \right] \tag{4.11}$$

where $\mathcal{R}$ is the set of rays in each batch, and $C(r)$, $\hat{C}_c(r)$ and $\hat{C}_f(r)$ are the ground truth, coarse volume predicted, and fine volume predicted RGB colors for ray r respectively. Note that even though the final rendering comes from $\hat{C}_f(r)$, we also minimize the loss of $\hat{C}_c(r)$ so that the weight distribution from the coarse network can be used to allocate samples in the fine network.

5

Experimental Setup and Results

In this chapter, we demonstrate the effectiveness and the scene rendering efficiency of our proposed TransNeRF architecture by conducting a comparative study between the pre-existing NeRF models like the basic NeRF model proposed by Mildenhall, Ben, et al [39] and NeRF++ model proposed by Zhang, Kai, et al [93] on publically available dataset benchmarks (UAVid)[33].

5.1. Dataset

Robustness of an algorithm can be demonstrated by benchmarking it on multiple, possibly unrelated datasets. With this ideology in mind, we benchmark our system on the UAVid dataset[33]. UAVid is a publicly available dataset for urban scene understanding that we use in this research, but unlike other promimently available public datasets like Cityscapes [9], etc, this dataset is collected from unmanned aerial vehicles (UAVs). Figure 5.1 shows different captured sample images of the UAVid dataset. This dataset introduces large scale variations, making it a very competitive and a tough benchmark. The dataset contains 300 densely labeled images into a total of 8 classes and 100 images are available for testing [33]. The resolution of the images is 4K i.e.3840×2160, which coupled with the small size of the dataset, makes it a very challenging benchmark. The reason why we decided to select UAVid datasets were due to the fact that they provide for a real-time analysis and benchmark for urban-scene understanding from different viewpoints, thereby effectively establishing the robustness of our architecture.

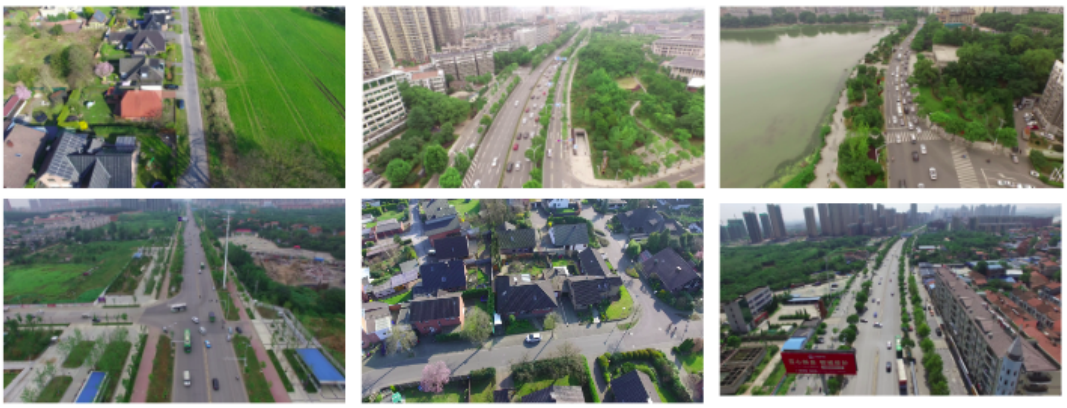

Figure 5.1: Sample training images from the UAVid dataset [33]

5.1.1. Training settings

As in original NeRF literature[39], for each scene we train a model initialized to random weights. Our experimental dataset primarily consists of video/images from the UAVid dataset and also some random

UAV captured videos from https://www.youtube.com/. Images are sampled at 5 frames per second & scaled down pre-training .The input images are scaled down from 4K resolution to resolutions of 512x256, 1024x512 and 2048x1024 respectively. Batch-sizes are set at 3 for UAVid and since we train and evaluate on a single GPU, we do not employ cross-GPU synchronized batch normalization. Furthermore, training iterations for each batch of image resolutions are set at 500k for UAVid dataset using Adam optimizer with initial learning rate of 0.001, decaying ten-fold every 150,000 iterations. All our experiments are conducted on a single NVIDIA RTX 2080Ti, with PyTorch 1.4 [49] and CUDA 10.2. Regarding our positional encoding functions γ we use 15 frequencies for encoding position and 4 when encoding viewing direction. During training, 512 points per ray are sampled from each of the coarse and fine models for a total of 1024 points per ray. We double that number during evaluation to 2048. The latent embedding vector for appearance has an embedding dimension of size $n(a)$ = 48.

5.1.2. Training procedure

We initially sample 150 input scenes from the UAVid dataset [33] videos at 4 frames per second. For the initial training phase we scale down the image resolution (originally at 4k) to a significantly lower ones like 512x256, 1024x512 2048x1024 resolutions. We train our model (proposed TransNeRF) on the first batch of low resolution images (i.e, 512x256) with the acquired camera properties (from COLMAP, [62]), position encoding input γ_x as inputs for the first 3D scene generating network ($MLP_{\theta 1}$) (Figure 4.3, page no:16). This at the end of the training $MLP_{\theta 1}$, outputs the RGB vector c(t), the density vector ($\sigma(t)$) and feature vector $z(t)$ which serves as an input to the color emmiting network ($MLP_{\theta 2}$) (Figure 4.4, page no:17). The second layer of the model, ($MLP_{\theta 2}$) is trained simultaneously with the latent appearance embedding input from the GLO, the viewing directional encoding (γ_d) that stores the viewing directions ($\theta\phi$) and feature vector $z(t)$ to output the feature color weights c_t. These feature color weights (which store RGB embeddings) are crucial as they are "resued" or "transfered" to the $MLP_{\theta 2}$, while training for the next batch of high resolution image samples. The outputs from the $MLP_{\theta 1}$ & $MLP_{\theta 2}$ namely the RGB and density features are later processed using volume rendering techniques for scene reconstruction. The subsequent training and validation losses for each batch of input image resolution are shown in the Figures. 5.2 & 5.3 below.

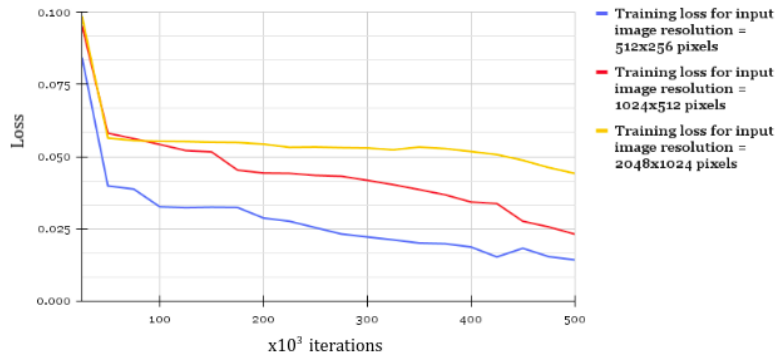

Figure 5.2: The training loss for each batch of the input image resolution (from 512x256 pixels to 2048x1024 pixels) over 500k iterations

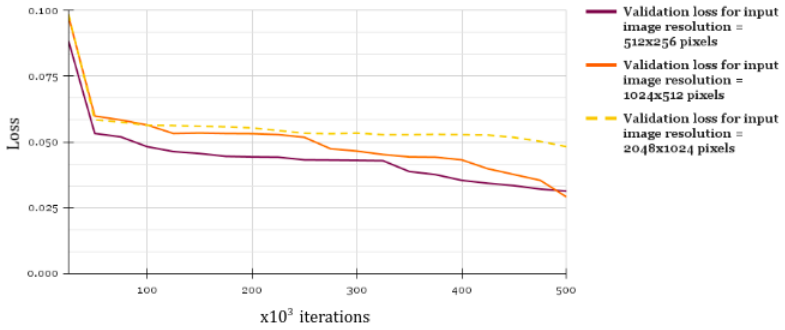

Figure 5.3: The validation loss for each batch of the input image resolution (from 512x256 pixels to 2048x1024 pixels) over 500k iterations

5.2. Experimental Results

We qualitatively (section 5.3.1) & quantitatively (section 5.3.2) show that our proposed approach outperforms prior work on a standard benchmark dataset and provide extensive ablation studies to validate our design choices. We show results on complex real-world scenes captured with roughly forward-facing images from the UAVid dataset. Our current experiments are carried out on 7 scenes captured via forward facing images from the UAVid at 4k resolution. We have tried to select mostly static scenes with very minimal motions of people and vehicles moving, as our model does not account for the the positional difference at a particular timestamp with respect to a particular location.

5.3. Comparisons

To evaluate our model we compare against current top-performing baseline techniques for neural view synthesis such as NeRF & NeRF++, detailed in the coming sections. All methods use the same set of input views i.e, 150 images (sampled at 4 fps) to train a separate network for each scene. For training the NeRF and NeRF++ models, The camera poses are estimated by COLMAP SfM [62]. For NeRF, we normalize the scene such that all cameras are inside the sphere of radius $\frac{1}{8}$. This normalization ensures that the unit sphere covers the majority of the background scene content (although some background geometry still lies outside the bounding unit sphere). To numerically compute the volumetric rendering integral for each camera ray, we uniformly sample points from the ray origin to its intersection with the unit sphere. Since NeRF++ ray-casts both inner and outer volumes and hence uses twice the number of samples per camera ray compared to a single volume, we also double the number of samples used by NeRF for the sake of fairness. Specifically, NeRF's coarse-level MLP uses 128 uniform samples, while the fine-level MLP uses 256 additional importance samples. Under this hyper-parameter setting, NeRF has roughly the same computation cost and GPU memory footprint during training and testing as NeRF++. We randomly sample 2048 camera rays at each training iteration, and train NeRF and NeRF++ for 500k iterations with a learning rate of $5e^{-4}$. A comparative analysis based on the training & validation loss for the different approaches is shown in Fig. 5.4 & 5.5 respectively.

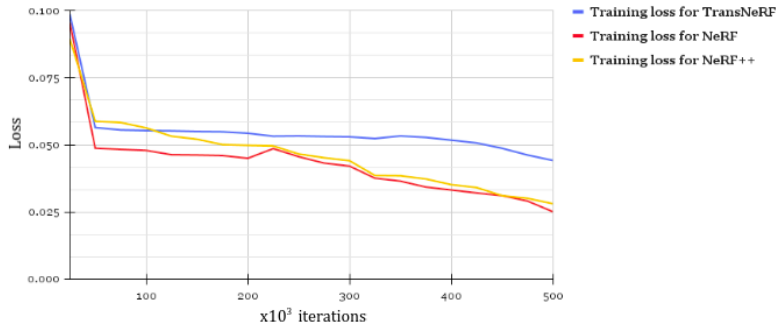

Figure 5.4: The training loss for NeRF, NeRF++ TransNeRF (our approach) over 500k iterations. The input image are sampled at 2048x1024 resolution while training NeRF, NeRF++ TransNeRF.

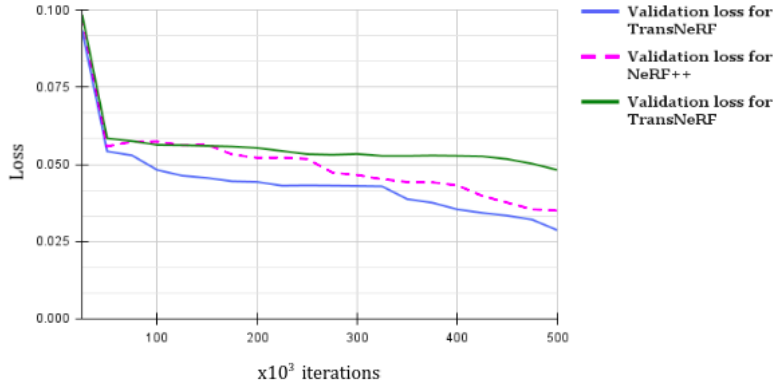

Figure 5.5: The validation loss for NeRF, NeRF++ TransNeRF (our approach) over 500k iterations. The input image are sampled at 2048x1024 resolutions.

5.4. Qualitative Evaluation

5.4.1. Rendering stable & consistent scenes

Comparisons on test-set views for scenes from UAVid dataset generated with a physically-based renderer are shown in Fig 5.6. It is clearly visible that our method is able to reconstruct fine details in both geometry and appearance while maintaining the structural stability of the scene when compared to the previous baseline models of NeRF and NeRF++. Comparisons on test-set views of real world scenes. NeRF and NeRF++ models are specifically designed for use case involving forward-facing captures of real scenes. Our method is able to represent fine geometry more consistently across rendered views than the conventional NeRF & NeRF++ models, as shown in rendering of different car models in Town A, Town C, Town D & Town E in Figure 5.6 below.

Figure 5.6: Qualitative evaluation of different scenes from the UAVid dataset using the NeRF, NeRF++ and TransNeRF architectures based on rendering stability and generation of consistent scenes.

5.4.2. Maintaining color balance across the rendered scene

TransNeRF is successful in rendering scenes that maintains the RGB & volume integrity of the captured images. This could be observed from the reconstructed scene in Fig. 5.7, where our approach is able to render reconstruction of the scene with the right color balance while maintaining the structural integrity of the objects within the scene, as observed in Town B (scene 2) & Construction site (scene 4).

Figure 5.7: Qualitative evaluation of different scenes from the UAVid dataset using the NeRF, NeRF++ and TransNeRF architectures based on maintaining color balance across the rendered scenes.

5.4.3. Efficient reconstruction of objects in foreground & background

Our method also efficiently differentiate between & reconstruct the complex foreground and background objects in the scenes, particularly the scene 7 - Forest, where our model is able to render a complicated object like the building communication grid in the background & forground scene respectively, that NeRF & NeRF++ model struggles to deliver a fine render [see Fig. 5.8]. Basic NeRF & NeRF++ model captures only the low-frequency geometry and color variation in each scene but is unable to reproduce any fine detail.

Figure 5.8: Qualitative evaluation of different scenes from the UAVid dataset using the NeRF, NeRF++ and TransNeRF architectures based on reconstruction of objects in foreground and backgrounds.

5.5. Quantitative Evaluation

We report Peak Signal to Noice Ratio (PSNR) factor, Structural Similarity Index Metric, and Learned Perceptual Image Patch Similarity (LPIPS) metric proposed by Zhang, Richard, et al [94] as our main quantitative metrics for measuring the quality of synthesized test scenes.

5.5.1. Peak Signal to Noice Ratio

Scene enhancement or improving the visual quality of a digital image can be subjective varying from one approach to another. For this reason, it is necessary to establish quantitative metrics to compare the effects of image enhancement algorithms on image quality. Peak signal-to-noise ratio (PSNR) is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. Using the same set of tests images, different 3D reconstruction approaches (like NeRF, NeRF++, TransNeRF) could be compared systematically to

identify whether a particular method produces better results. PSNR metric could be useful metric in identifying approaches can enhance/reconstruct a degraded known image to more closely resemble the original. Because many signals have a very wide dynamic range, (ratio between the largest and smallest possible values of a changeable quantity) the PSNR is usually expressed in terms of the logarithmic decibel scale (see equations 5.1 & 5.2 below).

$$PSNR = 20 \log_{10} \left(\frac{MAX_f}{\sqrt{MSE}} \right) \tag{5.1}$$

Where MSE is the mean square error expressed as:

$$MSE = \frac{1}{mn} \sum_{0}^{m-1} \sum_{0}^{n-1} \|f(i,j) - g(i,j)\|^2 \tag{5.2}$$

In equations 5.1 and 5.2, f represents the matrix data of our original image, g represents the matrix data of our degraded image in question, m represents the numbers of rows of pixels of the images and i represents the index of that row n represents the number of columns of pixels of the image and j represents the index of that column & $MAXf$ is the maximum signal value that exists in our original "known to be good" image.

5.5.2. Structural Similarity Index Metric

The Structural Similarity Index (SSIM) is a perceptual metric that quantifies image quality degradation caused by processing such as data compression or by losses in data transmission. Based on the work by Wang, Zhou, et al [83], SSIM calculates the Structural Similarity Index between 2 given images which is a value between the range [0,1]. A value of 1 indicates that the two given images are very similar or the same while a value of 0 indicates the two given images are very different. SSIM determines these values by means of extracting three key features: luminance, contrast and structure from the generated/rendered scene. The SSIM score between two images (x, y), where x being original one and y being the reconstructed/generated one is estimated as a function of:

$$\text{SSIM}(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \tag{5.3}$$

where the $l(x, y)]$ is the luminance comparison function, $c(x, y)$ is the contrast comparison function and $s(x, y)$ is the structural comparison funciton. α,β and γ denote the relative importance of each metric function.

5.5.3. Learned Perceptual Image Patch Similarity

Despite the most widely used perceptual metrics today, such as PSNR and SSIM, are simple, shallow functions, and fail to account for many nuances of human perception. LPIPS is a relatively recent effective metric that evaluates the distance between image patches. Higher the value of LPIPS, represents more dissimilarity between the rendered scene and original set of input images. Lower LPIPS score means there is more similarity between the generated scene and input image sequence.

5.5.4. Quantitative metric results

From the Table 5.1, it is clearly evident that our model outperforms the pre-existing NeRF models by quite some margin. We see a substantial increase in the PSNR & SSIM metric scores and decrease in the LPIPS scores for our TransNeRF model compared to the baseline NeRF approaches highlighted in Table 5.1. We also report the final rendered frame rate of different models & mean $\pm$ standard deviation across 7 scenarios with different random initializations. Best results are highlighted in the Table 5.2.

Scene 1 - Town A			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	18.54	0.534	0.598
NeRF++	19.432	596	0.574
TransNeRF	20.346	0.623	0.498
Scene 2 - Town B			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	19.83	0.498	0.634
NeRF++	19.783	0.498	0.583
TransNeRF	19.98	0.502	0.494
Scene 3 - Town C			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	17.65	0.501	0.698
NeRF++	18.564	0.557	0.543
TransNeRF	19.998	0.628	0.538
Scene 4 - Contruction site			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	18.86	0.523	0.686
NeRF++	19.564	0.589	0.643
TransNeRF	20.487	0.625	0.587
Scene 5 - Town E			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	19.86	0.478	0.634
NeRF++	20.568	0.543	0.584
TransNeRF	21.367	0.572	0.416
Scene 6 - Town D			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	19.344	0.523	0.745
NeRF++	20.178	0.554	0.673
TransNeRF	21.764	0.643	0.534
Scene 7 - Forest			
Model	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	19.86	0.395	0.632
NeRF++	20.345	0.496	0.534
TransNeRF	20.796	0.563	0.521

Table 5.1: Quantitative evaluation of the NeRF, NeRF++ & TransNeRF models on different scenes from the UAVid dataset. We use metrics such as PSNR, SSIM and LPIPS during our evaluation.

Model	PSNR ↑	SSIM ↑	LPIPS ↓	FPS
NeRF	18.452±0.005	0.433±0.004	0.644±0.115	6.8
NeRF++	19.98±0.009	0.486±0.011	0.584±0.113	6.7
TransNeRF	**20.546±0.002**	**0.524±0.012**	**0.532±0.043**	7.1

Table 5.2: Mean ± standard deviation of the metrics across 7 scenes with different random initializations. TransNeRF outperforms the baseline NeRF models, handling color perturbations efficiently as seen in difference in the PSNR values between NeRF & TransNeRF model.

6

Discussion

From the results section it could be inferred that our approach thoroughly outperform both baseline NeRF techniques that also optimize a separate network per scene (NeRF & NeRF++) in all scenarios. Furthermore, we produce qualitatively and quantitatively superior renderings compared to the aforementioned models across all evauation metrics, while using only the input images as our entire training set. The basic NeRF model proposed in [39], blends between different scene representations for rendering different views in complicated scenes from the UAVid dataset images, resulting in perceptually distracting inconsistent scenes. NeRF++ model, when compared to the basic NeRF model is able to differentiate between foreground and background objects in the scene render complicated geometry and texture, but its representational power for view synthesis is restricted by selection of only a single depth and color per camera ray. Our model utilises a latent optimization strategy that enables it to compute the latent apprearance features from the images. The biggest practical trade-offs between these methods are time versus space. However, conventional neural rendering apporaches like Local Light Field Fusion (LLFF) [38] and Scene Representation Networks (SRNs)[67] produces a large 3D voxel grid for every input image, resulting in enormous storage requirements (over 15GB for one “Realistic Synthetic” scene). Our method like the conventional NeRF model requires only 7 MB for the network weights (a relative compression of 3000× compared to other neural rendering approaches like), which is even less memory than the input images alone for a single scene from any of our datasets. Since our input images are trained over batches of several resolutions (from lower to higher), we face the challenge of longer training hours and computing memory compared to the baseline models.

Model	Memory	Training Period (hours)
NeRF	1158.23 MB	18.42
NeRF++	1176.34 MB	19.25
TransNeRF	1974.56 MB	26.49

Table 6.1: Computational expenses and run-time measurements for all the models have been done on a single RTX 2080Ti, on an input resolution of 1024×2048

6.1. Ablation Studies

We validate our approach’s design choices and parameters with ablation studies shown below. We present our mean results over a set of 7 different scenes from the UAVid dataset [33].

6.1.1. Varying input parameters

We study the effect of each input parameter on our model’s efficiency to reconstruct highly realistic scenes from sample images. Row 1 of the table 6.2, shows a minimalistic version of our model without

positional encoding (PE), view dependence (VD), latent appearance (LA) embedding and hierarchical sampling (HV). In rows 2-5 we remove a parameter while training the model. From the quantitave metric results it is clearly visible that the latent appearance embedding (row 5) & positional encoding (row 2) provide the largest quantitative benefit followed by viewing directions (row 3) & hierarchical volume sampling (row 4) respectively. iterations.

Parameter	Input	No. of Images	L	(N_c,N_f)	PSNR↑	SSIM↑	LPIPS↓
1) No PE, LA, VD & HV	xyz	150	-	(256,-)	17.367	0.527	0.643
2) No Positional Encoding (PE)	$xyz\theta\phi$	150	-	(64,512)	18.673	0.542	0.6418
3) No Viewing Directions (VD)	xyz	150	10	(64,128)	18.665	0.546	0.6414
4) No Hierarchial Volume (HV)	$xyz\theta\phi$	150	10	(256,-)	19.702	0.552	0.6404
5) No Latent Appearance embedding (LA)	$xyz\theta\phi$	150	10	(64,128)	18.543	0.489	0.684
6) Complete model	$xyz\theta\phi$	150	10	(64,128)	20.576	0.634	0.453

Table 6.2: An ablation study of our model based on the effect of input parameters on TransNeRF model. Metrics are averaged over the 7 scenes from UAVid dataset.

6.1.2. Varying resolution of input images

We study the effect of the input resolution images on final rendering using our model based on quantitaive metrics. Table 6.3 shows the significant increase in the rendering quality of our approach, as inferred from the increase in the quantitative metrics, PSNR & SSIM values as the resolution of input images increases.

Parameter	Input	No. of Images	L	(N_c,N_f)	PSNR↑	SSIM↑	LPIPS↓
1) Lowest Resolution (256x512)	$xyz\theta\phi$	150	10	(64,128)	16.934	0.623	0.523
2) Second lowest resolution (512x1024)	$xyz\theta\phi$	150	10	(64,128)	19.35	0.603	0.454
3) Complete model (resolution=1024x2048)	$xyz\theta\phi$	150	10	(64,128)	20.576	0.634	0.453

Table 6.3: Evaluating quantitative metrics (PSNR, SSIM & LPIPS) over increase in resolution of input images using our approach. Metrics are averaged over the 7 scenes from UAVid dataset.

6.1.3. Effect of number of input images for training the model

In this section we study the effect of the number of input images on the final scene quality of our proposed approach. From the table 6.4 based on the increase in PSNR & SSIM values, we can see that our model is able to successfully render novel realistic scenes from a very few sample input images (say 45-100).

Parameter	Input	No. of Images	L	(N_c,N_f)	PSNR↑	SSIM↑	LPIPS↓
1) Very few input images	$xyz\theta\phi$	45	10	(64,128)	18.543	0.623	0.59
2) Few input images	$xyz\theta\phi$	100	10	(64,128)	19.96	0.628	0.489
3) Complete model (resolution=1024x2048)	$xyz\theta\phi$	150	10	(64,128)	20.576	0.634	0.453

Table 6.4: Quantitative comparison of our model's performance over varying number of input images. Metrics are averaged over the 7 scenes from UAVid dataset.

6.1.4. Variations in input frequencies

In this section we study the performance of our approach with respect to the input frequencies of the positional and viewing direction encoder fucntion. L is the frequency factor used to calculate the positional encoding $\gamma(x)$ for x and the viewing directions $\gamma(d)$ [refer to equation 4.7, page:18]. While limiting to L=5 frequencies reduces performance of our model, increasing the number of frequencies from L=10 to L=15 does not improve performance significantly either. The benefit of increasing L is limited once $2L$ exceeds the maximum frequency present in the sampled input images (roughly resolution of 1024x2048 in our dataset)

Parameter	Input	No. of Images	L	(N_c,N_f)	PSNR↑	SSIM↑	LPIPS↓
1) Lower Frequencies (L)	$xyz\theta\phi$	150	5	(64,128)	19.45	0.534	0.624
2) Mid-order Frquencies (L)	$xyz\theta\phi$	150	10	(64,128)	20.576	0.634	0.453
3) High-order Frequencies	$xyz\theta\phi$	150	15	(64,128)	20.346	0.625	0.476

Table 6.5: An ablation study of our model. Metrics are averaged over the 7 scenes from UAVid dataset.

6.2. Limitations of our approach

While TransNeRF is able to produce photorealistic and temporally consistent renderings from a sequence of sampled captured images, rendering quality degrades in areas of the scene that are rarely observed in the training images, or only observed at very oblique angles or foreground. Similar to baseline models like NeRF and NeRF++, our approach is also sensitive to camera calibration errors, which can lead to distorted reconstructions on the parts of the scene that have been captured by incorrectly calibrated cameras. We too are limited by the training time and memory costs incurred during deployment of our model as compared to the previous NeRF based methods [39, 93]. Another major limitation of our model is its failure to account for dynamic scenes during reconstruction. Since our approach leverages images captured from the front-viewing camera and does not account for neither the deformation/dynamic motion of the objects in the captured scenes nor the time function as an input while training the model is it unable to render dynamic scenes. This is a similar drawback to the other baseline NeRF models as well.

6.2.1. Inefficiency during transient lighting conditions

A primary concern or drawback of our model like the conventional NeRF models is that, we address here is the assumption that the world is geometrically, materially, and photometrically static i.e, that the density and radiance of the world is constant. It could be attributed to the fact that images from the UAVid dataset [33] are mostly restricted while exhibiting considerable variation in terms of exposure, color correction, and tone-mapping. Hence our model is unable to account for the transient lighting changes during the training phase which could result in poor quality of the final sene rendering.

7

Conclusion

Our work directly addresses deficiencies of prior work that uses MLPs to represent objects and scenes as continuous functions. We demonstrate that representing scenes as 5D neural radiance fields (an MLP that outputs volume density and view-dependent emitted radiance as a function of 3D location and 2D viewing direction) produces better renderings than the previously-dominant approach of training deep convolutional networks to output discretized voxel representations.

We have presented TransNeRF, a novel approach that leverages transfer learning approach for efficient 3D scene reconstruction of complex environments from sequence of UAV captured images that builds upon the baseline NeRF model. We learn a per-image latent embedding capturing photometric appearance variations often present in UAV imagery data. Experimental evaluation on real-world data, more specifically the UAVid dataset, demonstrates significant qualitative and quantitative improvement over previous state of the art neural rendering approaches. Our approach of transfering learned color features from lower resolution input scenes, while training for high resolution static scenes has quantitatively and qualitatively outperformed other baseline NeRF based models like NeRF++. Our future work would be focused on deploying the framework on other standard datasets, especially in the domain of synthetic objects from the DeepVoxel dataset released by Sitzmann, Vincent, et al [68] and evaluating its performance quantitatively and qualitatively.

Although we have proposed a hierarchical sampling strategy to make rendering more sample-efficient (for both training and testing), there is still much more progress to be made in investigating techniques to efficiently optimize and render neural radiance fields. One possible domain of research could focus on improving neural scene rendering over the dynamic scenes and objects for efficient 3D scene reconstruction. We will also be looking forward to extending our model to improve on transient lighting changes while training in the future.

References

[1] Harold Achicanoy, Deisy Chaves, and Maria Trujillo. “StyleGANs and Transfer Learning for Generating Synthetic Images in Industrial Applications”. In: *Symmetry* 13.8 (2021), p. 1497.

[2] Sameer Agarwal et al. “Building rome in a day”. In: *Communications of the ACM* 54.10 (2011), pp. 105–112.

[3] Felix Altenberger. “High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs”. In: ().

[4] Sai Bi et al. *Neural Reflectance Fields for Appearance Acquisition*. 2020. arXiv: 2008.03824 [cs.CV].

[5] Piotr Bojanowski et al. “Optimizing the latent space of generative networks”. In: *arXiv preprint arXiv:1707.05776* (2017).

[6] Andrzej Brodzicki et al. “Transfer learning methods as a new approach in computer vision tasks with small datasets”. In: *Foundations of Computing and Decision Sciences* 45 (2020).

[7] Angel X Chang et al. “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012* (2015).

[8] Shenchang Eric Chen and Lance Williams. “View interpolation for image synthesis”. In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. 1993, pp. 279–288.

[9] Marius Cordts et al. “The cityscapes dataset for semantic urban scene understanding”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.

[10] David J Crandall et al. “SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.12 (2012), pp. 2841–2853.

[11] Brian Curless and Marc Levoy. “A volumetric method for building complex models from range images”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 303–312.

[12] Abe Davis, Marc Levoy, and Fredo Durand. “Unstructured light fields”. In: *Computer Graphics Forum*. Vol. 31. 2pt1. Wiley Online Library. 2012, pp. 305–314.

[13] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. “Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 11–20.

[14] Dawa Derksen and Dario Izzo. “Shadow Neural Radiance Fields for Multi-view Satellite Photogrammetry”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1152–1161.

[15] Yasutaka Furukawa and Jean Ponce. “Accurate, dense, and robust multiview stereopsis”. In: *IEEE transactions on pattern analysis and machine intelligence* 32.8 (2009), pp. 1362–1376.

[16] KDDX Fuzhen and Zhuang Zhiyuan Qi. “A Comprehensive Survey on Transfer Learning”. In: *Proceedings of the IEEE*. 2021.

[17] Kyle Genova et al. “Local deep implicit functions for 3d shape”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4857–4866.

[18] Riccardo Gherardi, Michela Farenzena, and Andrea Fusiello. “Improving the efficiency of hierarchical structure-and-motion”. In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 1594–1600.

[19] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[20] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.

[21] Farid Javadnejad. “Small unmanned aircraft systems (UAS) for engineering inspections and geospatial mapping”. In: (2017).

[22] Chiyu Jiang et al. “Local implicit grid representations for 3d scenes”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6001–6010.

[23] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. “Reinforcement learning: A survey”. In: *Journal of artificial intelligence research* 4 (1996), pp. 237–285.

[24] Marc Levoy. “Efficient ray tracing of volume data”. In: *ACM Transactions on Graphics (TOG)* 9.3 (1990), pp. 245–261.

[25] Marc Levoy and Pat Hanrahan. “Light field rendering”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 31–42.

[26] Tianye Li et al. *Neural 3D Video Synthesis*. 2021. arXiv: 2103.02597 [cs.CV].

[27] Tzu-Mao Li et al. “Differentiable monte carlo ray tracing through edge sampling”. In: *ACM Transactions on Graphics (TOG)* 37.6 (2018), pp. 1–11.

[28] Lingjie Liu et al. “Neural sparse voxel fields”. In: *arXiv preprint arXiv:2007.11571* (2020).

[29] Shichen Liu et al. “Soft rasterizer: A differentiable renderer for image-based 3d reasoning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7708–7717.

[30] Stephen Lombardi et al. “Neural volumes: Learning dynamic renderable volumes from images”. In: *arXiv preprint arXiv:1906.07751* (2019).

[31] Matthew M Loper and Michael J Black. “OpenDR: An approximate differentiable renderer”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 154–169.

[32] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[33] Ye Lyu et al. *The UAVid Dataset for Video Semantic Segmentation*. 2018. eprint: arXiv:1810.10438.

[34] Ricardo Martin-Brualla et al. “NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections”. In: *CVPR*. 2021.

[35] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[36] Nelson Max. “Optical models for direct volume rendering”. In: *IEEE Transactions on Visualization and Computer Graphics* 1.2 (1995), pp. 99–108.

[37] Lars Mescheder et al. “Occupancy networks: Learning 3d reconstruction in function space”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4460–4470.

[38] Ben Mildenhall et al. “Local light field fusion: Practical view synthesis with prescriptive sampling guidelines”. In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–14.

[39] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *European conference on computer vision*. Springer. 2020, pp. 405–421.

[40] Mehdi Mirza and Simon Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).

[41] Michael Niemeyer and Andreas Geiger. “Giraffe: Representing scenes as compositional generative neural feature fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 11453–11464.

[42] Michael Niemeyer et al. “Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3504–3515.

[43] Ishan Nigam, Chen Huang, and Deva Ramanan. “Ensemble knowledge transfer for semantic segmentation”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2018, pp. 1499–1508.

[44] Merlin Nimier-David et al. “Mitsuba 2: A retargetable forward and inverse renderer”. In: *ACM Transactions on Graphics (TOG)* 38.6 (2019), pp. 1–17.

[45] Sinno Jialin Pan and Qiang Yang. “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.

[46] Keunhong Park et al. “Deformable neural radiance fields”. In: *arXiv preprint arXiv:2011.12948* (2020).

[47] Keunhong Park et al. “Nerfies: Deformable Neural Radiance Fields”. In: *ICCV* (2021).

[48] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[49] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32 (2019), pp. 8026–8037.

[50] Sida Peng et al. “Animatable Neural Radiance Fields for Human Body Modeling”. In: *arXiv preprint arXiv:2105.02872* (2021).

[51] Eric Penner and Li Zhang. “Soft 3D reconstruction for view synthesis”. In: *ACM Transactions on Graphics (TOG)* 36.6 (2017), pp. 1–11.

[52] Thomas Porter and Tom Duff. “Compositing digital images”. In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. 1984, pp. 253–259.

[53] Albert Pumarola et al. “D-nerf: Neural radiance fields for dynamic scenes”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10318–10327.

[54] Nasim Rahaman et al. “On the spectral bias of neural networks”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 5301–5310.

[55] Gilles Rainer et al. “Neural btf compression and interpolation”. In: *Computer Graphics Forum*. Vol. 38. 2. Wiley Online Library. 2019, pp. 235–244.

[56] Gilles Rainer et al. “Unified neural encoding of BTFs”. In: *Computer Graphics Forum*. Vol. 39. 2. Wiley Online Library. 2020, pp. 167–178.

[57] Waseem Rawat and Zenghui Wang. “Deep convolutional neural networks for image classification: A comprehensive review”. In: *Neural computation* 29.9 (2017), pp. 2352–2449.

[58] Daniel Rebain et al. “Derf: Decomposed radiance fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14153–14161.

[59] Peiran Ren et al. “Global illumination with radiance regression functions”. In: *ACM Transactions on Graphics (TOG)* 32.4 (2013), pp. 1–12.

[60] Oleh Rybkin et al. “Model-Based Reinforcement Learning via Latent-Space Collocation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9190–9201.

[61] Tim Salimans et al. “Improved techniques for training gans”. In: *Advances in neural information processing systems* 29 (2016), pp. 2234–2242.

[62] Johannes L Schonberger and Jan-Michael Frahm. “Structure-from-motion revisited”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4104–4113.

[63] Katja Schwarz et al. “Graf: Generative radiance fields for 3d-aware image synthesis”. In: *arXiv preprint arXiv:2007.02442* (2020).

[64] Steven M Seitz and Charles R Dyer. “Photorealistic scene reconstruction by voxel coloring”. In: *International Journal of Computer Vision* 35.2 (1999), pp. 151–173.

[65] Pourya Shamsolmoali et al. “Image synthesis with adversarial networks: A comprehensive survey and case studies”. In: *Information Fusion* (2021).

[66] Ling Shao, Fan Zhu, and Xuelong Li. “Transfer learning for visual categorization: A survey”. In: *IEEE transactions on neural networks and learning systems* 26.5 (2014), pp. 1019–1034.

[67] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. “Scene representation networks: Continuous 3d-structure-aware neural scene representations”. In: *arXiv preprint arXiv:1906.01618* (2019).

[68] Vincent Sitzmann et al. “Deepvoxels: Learning persistent 3d feature embeddings”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2437–2446.

[69] Noah Snavely, Steven M Seitz, and Richard Szeliski. “Modeling the world from internet photo collections”. In: *International journal of computer vision* 80.2 (2008), pp. 189–210.

[70] Pratul P Srinivasan et al. “Pushing the boundaries of view extrapolation with multiplane images”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 175–184.

[71] Kenneth O Stanley. “Compositional pattern producing networks: A novel abstraction of development”. In: *Genetic programming and evolvable machines* 8.2 (2007), pp. 131–162.

[72] Xingyuan Sun et al. “Pix3d: Dataset and methods for single-image 3d shape modeling”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2974–2983.

[73] I. Surname, I. Surname, and I. Surname. *The Title of the Book*. 8th ed. City, State or Country: Publisher, 2000.

[74] Chris Sweeney et al. “Optimizing the viewing graph for structure-from-motion”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 801–809.

[75] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[76] Richard Szeliski and Polina Golland. “Stereo matching with transparency and matting”. In: *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE. 1998, pp. 517–524.

[77] Bill Triggs et al. “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372.

[78] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[79] Michael Waechter, Nils Moehrle, and Michael Goesele. “Let there be color! Large-scale texturing of 3D reconstructions”. In: *European conference on computer vision*. Springer. 2014, pp. 836–850.

[80] Mei Wang and Weihong Deng. “Deep visual domain adaptation: A survey”. In: *Neurocomputing* 312 (2018), pp. 135–153.

[81] Ting-Chun Wang et al. “High-resolution image synthesis and semantic manipulation with conditional gans”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8798–8807.

[82] Ting-Chun Wang et al. “Video-to-video synthesis”. In: *arXiv preprint arXiv:1808.06601* (2018).

[83] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.

[84] Marco A Wiering and Martijn Van Otterlo. “Reinforcement learning”. In: *Adaptation, learning, and optimization* 12.3 (2012).

[85] Kyle Wilson and Noah Snavely. “Robust global translations with 1dsfm”. In: *European conference on computer vision*. Springer. 2014, pp. 61–75.

[86] Daniel N Wood et al. “Surface light fields for 3D photography”. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000, pp. 287–296.

[87] Jiajun Wu et al. “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling”. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 82–90.

[88] Wenqi Xian et al. “Space-time Neural Irradiance Fields for Free-Viewpoint Video”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 9421–9431.

[89] Yazhou Xing, Zian Qian, and Qifeng Chen. “Invertible image signal processing”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 6287–6296.

[90] Bo Yang et al. “3d object reconstruction from a single depth view with adversarial learning”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 679–688.

[91] Lin Yen-Chen et al. “iNeRF: Inverting Neural Radiance Fields for Pose Estimation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[92] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *arXiv preprint arXiv:1411.1792* (2014).

[93] Kai Zhang et al. “Nerf++: Analyzing and improving neural radiance fields”. In: *arXiv preprint arXiv:2010.07492* (2020).

[94] Richard Zhang et al. “The unreasonable effectiveness of deep features as a perceptual metric”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.

[95] Tinghui Zhou et al. “Stereo magnification: Learning view synthesis using multiplane images”. In: *arXiv preprint arXiv:1805.09817* (2018).

[96] Xiaojin Zhu and Andrew B Goldberg. “Introduction to semi-supervised learning”. In: *Synthesis lectures on artificial intelligence and machine learning* 3.1 (2009), pp. 1–130.

[97] Xiaojin Jerry Zhu. “Semi-supervised learning literature survey”. In: (2005).