Skolkovo Institute of Science and Technology

as a manuscript

Ruslan Rakhimov

ADVANCING GENERALIZATION IN 3D COMPUTER VISION TASKS

PhD Dissertation Summary

for the purpose of obtaining academic degree
Doctor of Philosophy in Computer Science

Academic Supervisor:
Doctor of Science
Evgeny V. Burnaev

Moscow — 2024

Contents

1 Introduction

The exploration of 3D computer vision seeks to bridge the gap between digital and physical worlds, providing a detailed understanding of three-dimensional spaces from two-dimensional data. Despite significant progress, a primary challenge remains: improving the generalization capabilities of 3D computer vision models to perform reliably across diverse, unseen environments. This thesis focuses on this challenge, aiming to advance the field by enhancing the adaptability and efficiency of models across various 3D computer vision tasks. The **goal** of this research is to boost the capabilities of 3D computer vision systems in tasks such as generating synthetic data, creating more accurate 3D reconstructions, rendering new viewpoints more efficiently, and estimating human poses with greater precision.

1.1 Background and motivation

When solving the main interconnected tasks of 3D computer vision, each of which is critically important for interpreting and reconstructing the complex nature of the surrounding three-dimensional world, it is necessary for the corresponding methods to have good generalization capabilities. Among these tasks are the initial data collection and then registration, reconstruction, dynamic interpretation, and visualization of 3D environments. At each step, models need not only to understand and process large volumes of data, but also to work accurately and efficiently in scenarios for which they were not specifically trained.

At the core of 3D computer vision lies the crucial process of reconstruction [24, 83, 58, 59], where raw data is transformed into detailed 3D models, both static and dynamic. The initial step, data acquisition, forms the foundational stage where raw visual information is gathered using various sources such as RGB cameras or synthetic data generation techniques. All further steps and the final results of the analysis and reconstruction depend on the quality of the data.

Following data acquisition, the next critical step is registration, where different data sets are spatially aligned and integrated [5]. This step ensures that the subsequent processing stages, such as 3D reconstruction, are based on a unified dataset that accurately reflects the geometric and spatial relations within the captured scene.

The reconstruction phase begins after registration. In this stage, aligned data is processed to create a 3D digital model. Algorithms interpret and merge the data using techniques like triangulation or surface reconstruction [28, 40], resulting in a detailed three-dimensional representation. Outputs range from point clouds to complex formats like mesh models, and even textured 3D models that offer realistic surface details. An important output of this process is Computer-Aided Design (CAD) models, crucial in precision-focused fields like engineering and architecture. Traditional approaches often struggle with high-resolution and noisy data.

The task of novel view synthesis often occurs either after the reconstruction process or concurrently with it. This involves generating realistic images from viewpoints not originally captured during data acquisition. A significant challenge lies in developing a model capable of

effectively generalizing to unseen scenes and rapidly processing input data for rendering new views.

Accurately interpreting dynamic 3D environments, particularly those involving human interactions, is vital. This is especially relevant in applications like human pose estimation for augmented reality and virtual fitting rooms. Unlike traditional methods, which typically focus on identifying key body joints or landmarks [91], dense human pose estimation [3] provides a comprehensive mapping of the human form, generating a detailed per-pixel map of the human body and assigning each pixel of the person in the image to a corresponding 3D point on a body surface model [51]. This allows for a finer understanding of human posture and movement. However, current models are slow, hindering their application in real-world interactive scenarios.

Lastly, in the realm of human-centric 3D reconstruction, crucial for virtual avatar creation, there is a challenge to perform reconstruction from a single image, departing from traditional methods that rely on multiple images [2, 25, 4]. This requires a model to generalize well across identities.

1.2 Relevance of research

The field of 3D computer vision has seen significant advances yet continues to confront challenges that limit its effectiveness and broader applicability, particularly in generalizing across diverse and complex environments.

To address the challenges in generalizing across diverse environments, there have been significant developments in the use of synthetic data. While generative learning has enabled the creation of realistic synthetic data, video generation remains a resource-intensive task that often fails to achieve the desired quality [52].

In geometric modeling, methods for detecting features of 3D objects (such as sharp feature curves curves, surface lines along which the normal field experiences discontinuities) require careful parameter tuning for each model, thus complicating scalability [90, 16]. Standard strategies, such as surface segmentation and patch fitting, although robust to noise, still lack flexibility and computational efficiency [50, 9]. Similarly, machine learning models for feature classification are ineffective when working with noisy data [27, 31].

Traditional methods in novel view synthesis, including view interpolation and light field rendering, often falter with complex geometries and diverse lighting conditions [47, 76]. Advanced techniques such as Neural Radiance Fields (NeRF) and voxel-based methods face issues with high computational demands and optimization [56, 38]. Neural Point-Based Graphics (NPBG) improves rendering quality but needs extensive optimization for each scene, limiting its usability [1].

Current human pose estimation models, robust in their performance, are unsuitable for mobile deployment due to their significant computational requirements [3, 98]. Although advance-

ments like Slim DensePose and uncertainty estimation techniques exist, they have yet to sufficiently optimize for mobile usage in terms of size and speed [62, 61].

Furthermore, while 2D-focused techniques in head appearance modeling are advanced, 3D modeling often depends on restrictive data like 3D scans [39, 17, 74]. New methods using implicit representations such as NeuS and VolSDF show potential yet struggle with scene adaptation [86, 63, 99, 41].

These challenges validate the need for this research to enhance the robustness, efficiency, and practicality of 3D computer vision technologies, addressing existing limitatiions to better align with the requirements of real-world applications.

1.3 Research Objectives and Scope

The goal of this thesis is to develop and implement new methods and approaches aimed at improving the generalization capabilities of models in 3D computer vision tasks. To achieve this goal, the following objectives were set:

1. Investigate the possibility of improving model generalization for video generation under computational resource constraints during training.
2. Develop a method for predicting sharp geometric features in 3D models with enhanced generalization capabilities when working with new, previously unseen 3D models of different scales and with scanning noise.
3. Develop an approach for novel view synthesis, effectively generalizable to new scenes without requiring intensive optimization.
4. Improve model generalization for dense human pose estimation, achieving high performance and quality under strict model size and speed constraints.
5. Improve the generalization ability of algorithms for 3D head portrait reconstruction so that they work effectively with a single input image.

1.4 Results

The work is based on the use of **methodology and methods** of machine learning, deep learning, and computer vision.

Reliability of the results is ensured by the correct application of validated scientific tools for research and analysis. The developed algorithms were experimentally tested on various tasks using both synthetic and real datasets. Detailed reports on the conducted experiments, open-source code, and access to the data allow for the reproduction of the obtained results. The research has been published in leading scientific journals and presented at computer vision conferences.

Key points presented for defense:

1. Investigation of the possibility of video modeling in a discrete latent space.
2. A regression method for localizing special curves of 3D objects, which reliably handles noisy, high-resolution 3D data and outperforms existing methods.
3. A model for generating new views of a scene from a set of images of that scene, which effectively generalizes to new scene data without additional training.
4. A model for efficiently solving the task of dense human pose estimation, which can be deployed on a mobile device.
5. Adaptation of the 3D head reconstruction algorithm based on a single image for use with unknown camera parameters.

1.5 Importance of work

In this dissertation, we propose new approaches that enhance the generalization of solutions to 3D computer vision tasks at various stages of 3D model construction. We introduce a new method for video generation [68] that performs comparably to existing methods but requires significantly fewer computational resources for model training. We developed a model for predicting sharp features from three-dimensional point clouds [53], trained on synthetic data with minimal retraining on real data, which provides accurate predictions for real 3D objects. We propose a model for novel view synthesis [66] that does not require retraining on data from a new scene and achieves comparable quality and rendering speed up to 22 frames per second, which is significantly higher than the speed of existing approaches. For real-time dense human pose estimation, we developed a model [67] that achieves an optimal balance between performance and quality, allowing the model to be deployed on a mobile device. We have developed a model for three-dimensional reconstruction of a human head, which can operate from the data of a single photograph and effectively generalizes to data from new people [8].

These enhancements not only broaden the practical applications in augmented and virtual reality, robotics, and other sectors but also underscore the importance of this work in pushing the boundaries of generalization within the 3D computer vision field.

2 Publications and approbation of the research

This thesis is based on the following five main research papers, all of which are indexed by SCOPUS and Web of Science.

First-tier publications

1. *Rakhimov, R.**, Ardelean, A. T.*, Lempitsky, V., & Burnaev, E. (2022). *NPBG++: Accelerating Neural Point-Based Graphics.* In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 15969-15979). CVPR 2022. CORE A*. https://doi.org/10.1109/cvpr52688.2022.01550

   Summary: We present, NPBG++, an advanced model for novel view synthesis (NVS) to generate photorealistic views of a scene from limited image sets. It introduces a streamlined approach that predicts neural descriptors from source images in a single pass, eliminating the need for extensive per-scene optimization. This method not only reduces the scene fitting time significantly but also improves rendering quality. Utilizing a combination of U-Net-based feature extraction, permutation-invariant descriptor aggregation, and a refiner network, NPBG++ provides efficient and high-quality rendering. Empirical results show that NPBG++ performs comparably to leading NVS methods, offering faster rendering times and high-quality outputs, positioning it as a promising solution for real-time applications in virtual reality, cinematography, and gaming industries.

   Main Contribution: I played a crucial role in both conceptualizing and developing the model. I developed most of the NPBG++ pipeline and contributed significantly to the experiments.

2. Matveev, A., *Rakhimov, R.*, Artemov, A., Bobrovskikh, G., Egiazarian, V., Bogomolov, E., Panozzo, D., Zorin, D., & Burnaev, E. (2022). *DEF: Deep Estimation of Sharp Geometric Features in 3D Shapes.* Proc. SIGGRAPH 2022 conf. ACM Transactions on Graphics, 41(4) (ACM ToG). CORE A*. https://doi.org/10.1145/3528223.3530140

   Summary: We develop a novel, learning-based framework designed to predict sharp geometric features in 3D shapes by regressing a scalar field representing the distance from point samples to the nearest feature line. The DEF framework utilizes deep estimators on local patches of depth images, employing training datasets derived from both synthetic and real-world sources to ensure robustness and adaptability. Key components of DEF include the construction of training data, a patch-based deep estimation model, and innovative methods for integrating predictions over complete 3D models and extracting parametric feature curves. The proposed method significantly outperforms existing feature detection techniques in terms of precision and generalization across various quality metrics and datasets. The extensive evaluations demonstrate DEF's superior ability to handle

* --- Equal Contribution

large-scale datasets, positioning it as a powerful tool for advancing geometric analysis and 3D computer vision.

Main Contribution: I developed the first of the two major components of the entire pipeline, specifically the method for regressing a scalar field representing the distance from point samples to the nearest feature line. I also made significant contributions to the experiments regarding patch-level comparisons on synthetic data and was responsible for fine-tuning models on real data.

3. *Rakhimov, R.**, Bogomolov, E.*, Notchenko, A., Mao, F., Artemov, A., Zorin, D., & Burnaev, E. (2021). *Making DensePose Fast and Light.* In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1869-1877). WACV 2021. CORE A. https://doi.org/10.1109/wacv48630.2021.00191

   Summary: We introduce Mobile Parsing R-CNN, a new architecture designed for real-time DensePose estimation on mobile devices. Addressing the challenges posed by existing DensePose models that are heavily parameterized and require robust server-side infrastructure, this research achieves a significant reduction in model size (by 17 times) and latency (by 2 times) compared to traditional models. The empirical results demonstrate that the model not only retains good accuracy but also improves operational efficiency, making it a promising solution for applications requiring real-time, on-device human form understanding.

   Main Contribution: I took a leading role in developing improvements in the model's architecture and contributed to the experiments.

4. Burkov, E., *Rakhimov, R.*, Safin, A., Burnaev, E., & Lempitsky, V. (2023). *Multi-NeuS: 3D Head Portraits from Single Image with Neural Implicit Functions.* IEEE Access, Q1 Journal. https://doi.org/10.1109/access.2023.3309412

   Summary: We develop Multi-NeuS, a novel 3D neural implicit model tailored for reconstructing textured 3D human head models from single or few images, enhancing applications in AR, VR, XR, and gaming. Multi-NeuS builds on the NeuS framework by incorporating shared and scene-specific layers, enabling it to efficiently manage multiple objects and scenes. The architecture optimizes for both geometric and textural details, overcoming the limitations of earlier models that depend on extensive datasets or 3D scans. Through a meta-learning approach, Multi-NeuS learns a generalizable representation, which is then fine-tuned for individual scenes. Empirical results demonstrate the model's capability to produce high-quality reconstructions from minimal input, showing comparable or better performance to existing methods.

   Main Contribution: I made a significant contribution to various stages of the model development process, such as data preprocessing, optimization of camera parameters, code

* --- Equal Contribution

for metrics calculation, extraction of a polygonal representation of the scene, adaptation of the algorithm to work with images with unknown camera parameters.

Second-tier publications

1. *Rakhimov, R.**, Volkhonskiy, D.*, Artemov, A., Zorin, D., & Burnaev, E. (2021). *Latent Video Transformer.* Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications. VISIGRAPP 2021. CORE B. https://doi.org/10.5220/0010241801010112

   Summary: The Latent Video Transformer (LVT) is a new model introduced to tackle the complexities of video generation, specifically focusing on predicting future video frames from a sequence of initial conditioning frames. This model finds its utility in various applications including self-driving technology, anomaly detection, and animated content creation. LVT leverages a combination of a frame autoencoder, specifically the VQ-VAE architecture for encoding frames into a discrete latent space, and an autoregressive generative model that predicts subsequent frames, reducing computational demands while maintaining quality. It employs a structured approach to generate videos by sequentially creating each frame in a latent space before mapping them back to the pixel space, ensuring efficient video generation with reduced resource requirements. The model has been tested on datasets like BAIR Robot Pushing and Kinetics-600, demonstrating comppetitive results, albeit with some limitations in complex scenarios, highlighting the ongoing challenges and the necessity for further advancements in video generation technology.

   Main Contribution: I came up the initial idea of moving the generation process to the discrete latent space and developed the overall pipeline, particularly the first stage, the frame autoencoder. I also made significant contributions to the experiments.

Reports at conferences and seminars

1. "Making DensePose Fast and Light" talk at the *WACV* conference, Online, 2021;
2. "Latent Video Transformer" talk at the *VISIGRAPP* conference, Online, 2021;
3. "NPBG++: Accelerating Neural Point-Based Graphics", talk at the conference Fall into ML 2022, Moscow, Russia;
4. "Multi-Sensor Large-Scale Dataset for Multi-View 3D Reconstruction", talk at the conference Fall into ML 2023, Moscow, Russia.

The author has also contributed to the following publications

1. Voynov, O., Bobrovskikh, G., Karpyshev, P., Galochkin, S., Ardelean, A. T., Bozhenko, A., Galochkin, S., Karmanova, E., Kopanev, P., Labutin-Rymsho, Y., *Rakhimov, R.*, Safin, A.,

* --- Equal Contribution

Serpiva, V., Artemov, A., Burnaev, E., Tsetserukou, D., & Zorin, D. (2023). *Multi-Sensor Large-Scale Dataset for Multi-View 3D Reconstruction.* In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 21392-21403). CVPR 2023. CORE A*. Indexed by SCOPUS, Web of Science. https://doi.org/10.1109/cvpr52729.2023.02049

3 Content of Works

The dissertation is structured into several sections, each focusing on a distinct research article.

In Section 3.1, we explore the feasibility of modeling videos within a discrete latent space. This section covers the current methodologies, introduces the architecture of the proposed model, and provides a comparative analysis with existing approaches.

In Section 3.2 we introduce a novel regression technique for pinpointing specific curves on 3D objects. We describe the architecture of this new method and compare it against existing techniques.

In Section 3.3, we propose a new model for generating novel species from a collection of input images. This section evaluates the model's quality and efficiency in terms of scene modeling and rendering, comparing these aspects to those of current methods.

In Section 3.4 we describe a model for dense human pose estimation with enhanced processing speed, making it suitable for mobile devices. We review how different components of the model affect both the quality of the results and the speed of operation.

Finally, in Section 3.5 we present a model for creating three-dimensional representations of human heads from a single image. We describe how to apply the model to images with unknown camera parameters.

3.1 Latent Video Transformer

We address the challenge of video generation, specifically predicting future video frames given a few input conditioning frames. This task finds practical applications in diverse fields such as self-driving technology, anomaly detection, time-lapse creation [60], and animated landscape generation [22], where accurate predictions of future video frames are crucial for decision-making and content creation.

Despite recent advances in generative learning that have facilitated the creation of realistic objects with high quality, including images, text, and speech, video generation remains a formidable challenge. Neural networks, even for brief videos consisting of 16 frames at low resolution, demand a substantial computational load, reaching up to 512 Tensor Processing Units (TPUs) [52] for parallel training. Despite these computational demands, the resulting video quality remains low.

To address this challenge, we introduce the Latent Video Transformer (LVT), a model that leverages autoregressive generation in a discrete latent space [84]. Our approach significantly reduces computational demands while preserving the quality of generated videos. By combining representation learning with recurrent video generation, the LVT not only overcomes GPU memory limitations but also accelerates inference speed, offering a promising solution for resource-intensive video generation tasks.

Model Description

The Latent Video Transformer (LVT) predicts subsequent frames given an initial set. We define a video sequence, denoted as X, as a series of T frames $x_{t=1}^{T}$, where each frame $x_t \in \mathbb{R}^{H \times W \times 3}$ has dimensions H and W with 3 RGB channels. Our objective is to generate the remaining frames $(T - T_0)$ given the first T_0 frames. The LVT model comprises two main components: a frame autoencoder and an autoregressive generative model.

For the frame autoencoder, we employ the VQ-VAE [84] architecture, which is a variational autoencoder with a discrete latent space. The VQ-VAE, depicted in Figure 1, is designed to encode an input image $x \in \mathbb{R}^{H \times W \times 3}$ by utilizing a codebook $e \in \mathbb{R}^{K \times D}$. Here, K represents the codebook size, indicating the categorical nature of the latent space, and D signifies the dimensionality of an embedding in the codebook.

Broadly, the VQ-VAE comprises an *encoder* that compresses the image into a more compact representation, $z_e(x) \in \mathbb{R}^{h \times w \times D}$; a *bottleneck* that discretizes each pixel by associating it with its nearest embedding, e_i, from the codebook, producing $z(x) \in [K]^{h \times w \times 1}$; and a *decoder* that takes discrete latent codes, $z(x)$, and maps them to corresponding embeddings, decoding the result, $z_q(x) \in \mathbb{R}^{h \times w \times D}$, back to the input pixel space.

The training objective for VQ-VAE includes a reconstruction loss and a regularization term, expressed by the following equation:

$$L = \|x - \textit{decoder}(z_q(x))\|^2 + \|z_e(x) - \text{sg}[e]\|^2. \tag{1}$$

Here, sg[] represents the stop gradient operator, which outputs its argument during the forward pass and zero gradients during the backward pass. We employ Exponential Moving Average (EMA) updates for codebook variables.

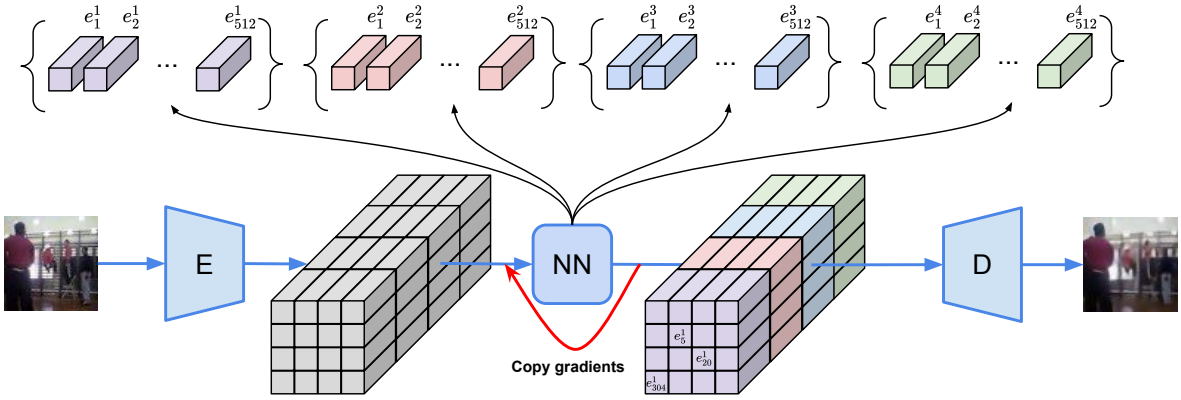

Figure 1: *Architecture of the frame autoencoder.* The encoder divides the input image into $n_c = 4$ parts along the channel dimension. Pixels in each segment are then paired with the nearest embeddings from the codebook, which the decoder uses as input.

The frame encoder transforms the initial T_0 frames into a discrete representation denoted as $Z_0 \in [K]^{T_0 \times h \times w \times n_c}$. The autoregressive model is then employed to generate new frames, totaling $T - T_0$, conditioned on Z_0. We adopt the Video Transformer [92], an autoregressive video generative model, applying it within the latent space as opposed to the pixel space in the original paper. The architecture of the video transformer is detailed in the original paper [92].

The model takes a tensor $Z \in [K]^{T \times h \times w \times n_c}$ as input and initiates the generation process by priming it with the first T_0 latent frames, i.e., $Z_{:T_0,:,:,:} = Z_0$. The remaining latent frames can be randomly initialized as the generation process conditions solely on previously generated or priming pixels. The model employs the concept of subscale [54], generating a latent video as a sequence of non-overlapping slices. Using a subscale factor $\mathbf{s} = (s_t, s_h, s_w)$, the latent video is divided into $s = s_t s_h s_w$ slices, each of size $T/s_t \times h/s_h \times w/s_w$. The generation process unfolds sequentially, slice by slice, pixel by pixel within a slice, and channel by channel for each pixel:

$$p(Z) = \prod_{i=0}^{Thw-1} \prod_{k=0}^{n_c-1} p\left(Z^k_{\pi(i)} | Z_{\pi(<i)}, Z^{<k}_{\pi(i)}\right). \tag{2}$$

where $p(Z)$ represents the probability distribution over the latent video sequence Z. Pixels in each slice $Z_{(a,b,c)}$ are generated in raster-scan order, while slices are generated in subscale order: $Z_{(0,0,0)}, Z_{(0,0,1)}, \ldots, Z_{(s_t-1,s_h-1,s_w-1)}$.

The transformer model comprises an encoder and a decoder. To generate a new pixel value within a slice $Z_{(a,b,c)}$, the encoder first produces the representation of already generated slices

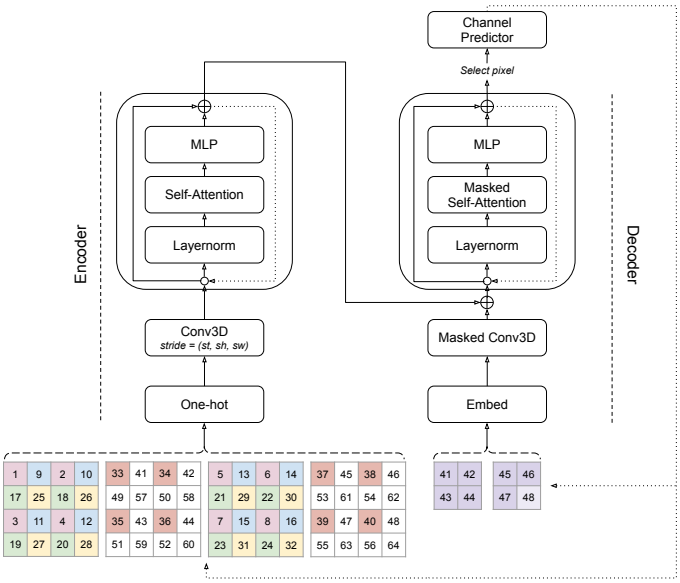

Figure 2: *Latent Video Transformer architecture.* Numbers show generation order, colored pixels represent generated pixels, white pixels are zero-padding, and same-color pixels belong to the same slice. Example: generating the last pixel of slice $Z_{(1,0,1)}$ for a latent video of size $(t, h, w) = (4, 4, 4)$ with subscale factors $(s_t, s_h, s_w) = (2, 2, 2)$.

$Z_{<(a,b,c)}$. This representation is then mixed with the representation of already generated pixels inside the current slice $Z_{(a,b,c)}$. The autoregressive order is maintained through padding within the encoder and masking in convolutions and attention within the decoder. After generating a new pixel value, the corresponding padding is replaced with the generated output, and the generation process recurs. The generation process, in the case of spatiotemporal ($s_t > 0, s_h > 0, s_w > 0$) subscale, is illustrated in Figure 2.

Once the generation process is finished, the latent frame decoder takes $Z \in [K]^{T \times h \times w \times n_c}$ as input (where all values are now valid), maps it to the previously learned embeddings $Z_q \in \mathbb{R}^{T \times h \times w \times D}$, and decodes it back frame by frame to the original pixel space $X \in \mathbb{R}^{T \times H \times W \times 3}$.

Empirical Results

We evaluate the video predictions using the Fréchet Video Distance (FVD) [30]; in addition, we include bits per dimension (bits/dim), representing the negative $\log_2$-probability averaged across all generated (latent) pixels and channels. We also provide the baseline solution: what if we take the last ground truth frame and use it as a prediction for all future frames.

We present both quantitative (Tables 1a, 1b) and qualitative outcomes (Figure 3) on two datasets, BAIR Robot Pushing [20] and Kinetics 600 [10]. While our performance matches that of other methods on the BAIR Robot Pushing dataset, we observe inferior results on Kinetics-600, attributing this discrepancy to error accumulation within the Transformer model, which we connect to the dataset's elevated complexity and diversity.

Table 1: *Quantitative evaluations.* We follow the setup of previous approaches [13, 92] and train the video generator conditioning on one frame and report metrics for videos of 16 frames. FVD and bits/dim are computed on videos with five priming frames and one priming frame accordingly.

(a) BAIR Robot Pushing dataset

Method	**bits/dim**(↓)	**FVD**(↓)
Baseline	-	320.90
VideoFlow [44]	1.87	-
SVP-FP [18]	-	315.5
CDNA [23]	-	296.5
LVT (ours, $n_c = 1$)	1.25	275.71 ± 5.41
SV2P [19]	-	262.5
LVT (ours, $n_c = 4$)	1.53	125.8 ± 2.9
SAVP [46]	-	116.4
DVD-GAN-FP [13]	-	109.8
TriVD-GAN-FP [52]	-	103.3
Axial Transformer [32]	1.29	-
Video Transformer [92]	1.35	94 ± 2

(b) Kinetics-600 dataset

Method	**bits/dim**(↓)	**FVD**(↓)
Baseline	-	271.00
LVT (ours)	2.14	224.73
Video Transformer [92]	1.19	170 ± 5
DVD-GAN-FP [13]	-	69.15 ± 1.16
TriVD-GAN-FP [52]	-	25.74 ± 0.66

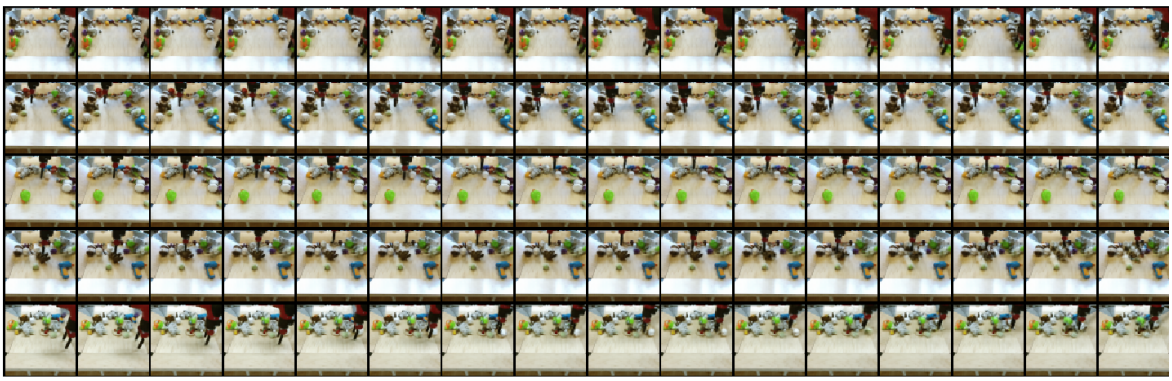

Figure 3: *Results on the BAIR Robot Pushing dataset.* Each row depicts a distinct video, showcasing the initial five frames as real and the subsequent frames as generated.

Conclusion

A video generation model has been developed based on the concept of modeling video in a discrete latent space. The presented model exhibits good generalization ability, meaning it can generate video sequences from previously unseen conditional input frames. Moreover, this is achieved using limited computational resources during the training stage, consisting of 8 V100 GPUs, whereas alternative methods require up to 512 tensor processors for training.

3.2 DEF: Deep Estimation of Sharp Geometric Features in 3D Shapes

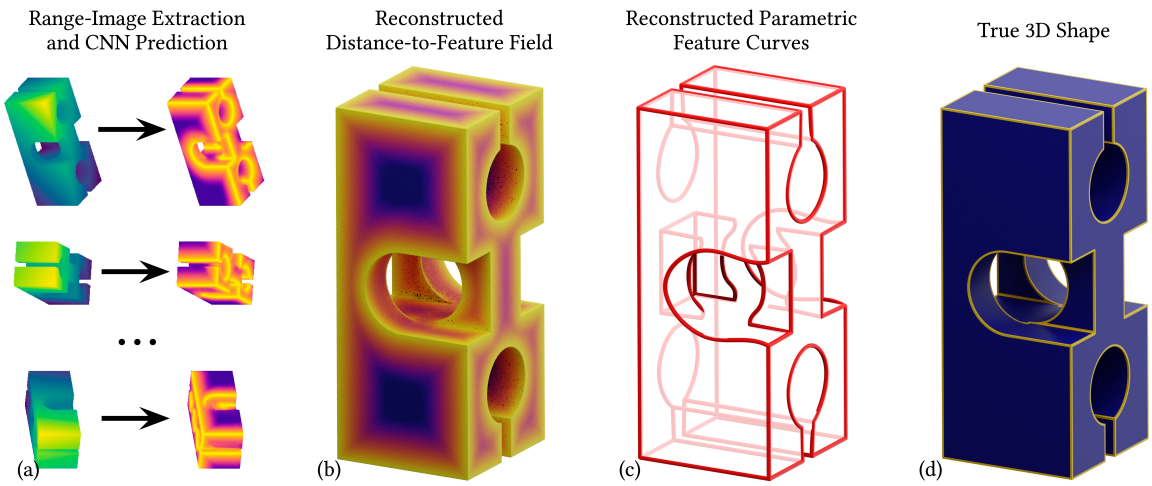

Figure 4: *DEF Overview.* (a) We develop efficient models for distance-to-feature estimation using range scans as an input; (b) Our method integrates these predictions to estimate complete 3D shapes; (c) Out method enables extracting explicit feature curves; (d) This results in precise reconstructions of both straight and curved features, closely aligned with CAD models.

We address the task of predicting sharp geometric features in 3D shapes (surface curves across which the surface normals have a sudden change). Solving this task has a key value for the task of CAD (Computer-Aided Design) model reconstruction while eliminating the need for manual feature definition and parameter tuning.

Existing methods for geometric feature detection include local estimation techniques [90, 16], which focus on computing differential properties in small areas but require extensive parameter tuning for each specific model. Surface segmentation methods [50] aim to identify surface patches and classify their interfaces as features, yet they are ineffective for incomplete models. Patch fitting strategies [9] involve fitting predefined primitives to large mesh regions, offering noise robustness but at the cost of computational efficiency and flexibility due to reliance on predetermined shapes. Meanwhile, the rise of data-driven methods, particularly machine learning models for feature point classification [27, 31], marks a significant shift. However, they struggle with scalability and robustness in the presence of noisy data.

Our proposed method, DEF, introduces a novel distance-to-feature regression on local patches, departing from binary classification and offering scalability, adaptability, and improved performance in detecting sharp geometric features, thus addressing critical limitations in current methodologies.

Model Description

Our algorithm processes depth images, sourced from real-world scans or simulated for synthetic mesh data, as input for a given object. It outputs a truncated distance-to-feature scalar

function for each input point, demonstrated in Figure 4. The method includes four main components.

The first component, *training data construction*, involves creating DEF-Sim (synthetic) and DEF-Scan (real-world) datasets for the following model training. DEF-Sim, based on the ABC dataset [42], uses boundary representation and sharp curve annotations for training, calculating the distance-to-feature for each point p as $d^{\varepsilon}(p) = \min(\|q(p) - p\|_2, \varepsilon)$, where $q(p)$ is the point on the nearest sharp feature curve or edge and ε as the truncation radius. DEF-Scan includes 3D-printed objects scanned with a structured light 3D scanner, aligned with CAD models. These datasets provide varied training environments in resolution, noise levels, and sample sizes crucial for developing precise models for sharp feature detection.

The second component, *Patch-Based Deep Estimators* (DEFs), focuses on estimating distance-to-feature in depth images. Trained initially on the synthetic datasets and fine-tuned with real-world data, these models aim to minimize $\min_{\theta} \frac{1}{N} \sum_i^N L(d_i, f(P_i; \theta))$, where d_i is the actual distance-to-feature for patch P_i, $f(\cdot; \theta)$ is the model with parameters θ, and L is the loss function. CNNs, particularly the U-Net model with a ResNet-152 architecture, were found to be most effective. The Histogram loss [36] significantly improved regression quality by focusing the network on a narrower range of target distances. Network performance stabilizes with datasets over 64000 instances, and DEFs can detect features at various sampling rates, indicating model adaptability.

The third component is *estimation on complete 3D models*. We present a novel method for this task by fusing per-patch distance-to-feature predictions using deep estimators. This process first involves converting an input 3D model into a set of range images, $I_{i=1}^{n_v}$, from multiple directions. Each image patch I_i is independently processed by our neural network, yielding distance predictions sensitive to interior feature curves. The essence of our approach is the transfer of these predictions across patches. For a given pair (s, t) of source and target views, and with the distance-to-feature estimate d_s available in the source view, we utilize a warping-based view synthesis mechanism to produce a warped prediction $\widehat{d}_t^{s \to t}$ for each pixel in the target view by re-projecting predictions from the source view's image plane. The final step involves deriving a coherent global distance estimate, computed as the minimum across warped estimates from different source views $\widehat{d}_t = \min_s \widehat{d}_t^{s \to t}$. This method effectively integrates feature-sensitive information throughout the complete 3D shape, as validated through various ablation studies.

Finally, we *extract parametric feature curves* from point clouds, merging corner detection, graph structure analysis, and spline fitting. This involves classifying and segmenting local points, constructing a curve graph, fitting and optimizing splines, and applying a post-processing procedure, which includes a quality metric and filtering based on curve length.

Empirical Results

We evaluate our feature estimation method using several quality measures, including root mean squared error (RMSE), recall, false positive rate with varying thresholds, for assessing the quality of distance-to-feature regression, feature line estimation precision in 3D shapes.

We compare DEF with five leading methods for extracting feature lines from 3D shapes, covering both traditional and deep learning techniques. VCM [55], a non-learning approach, utilizes Voronoi covariance measures. Sharpness Fields (ShF) [65] employs a CNN for sharpness field prediction. EC-Net [101] leverages a PointNet++[64]-based network for sharp feature line detection, PIE-NET [88] uses a two-stage process for feature curve segmentation and parametric curve proposal generation.

Through extensive evaluations on synthetic and real-world datasets, DEF consistently outperforms its competitors in terms of recall and false positive rates. In the patch-based comparison (Table 2), DEF outperforms competitors like ShF, VCM, and PIE-NET. The evaluation on complete 3D models and real-world scanned shapes (Figure 5) further reinforces DEF's superiority, showcasing its ability to robustly regress distance-to-feature fields and outperforming competitors in terms of precision and generalization. Overall, the results validate DEF as a powerful and versatile framework for sharp geometric feature detection in 3D shapes.

Table 2: *Quantitative Evaluation of Sharp Feature Line Estimation.* Our *local patch-based* networks, dedicated to *distance-to-feature estimation* and feature line *segmentation*, outperform competing methods in several segmentation and regression quality metrics (evaluated using DEF-Sim synthetic image patches). To obtain DEF segmentation results, a threshold of 0.02 is applied to the predicted distance.

Method	RMSE$\times 10^{-3}\downarrow$	RMSE-$q_{95}\times 10^{-3}\downarrow$	Recall %$\uparrow$	FPR %$\downarrow$
VCM [55]	---	---	49.1	3.1
EC-Net [101]	---	---	79.2	2.9
DEF *(Trained on EC data)*	124.1	501.1	56.0	0.15
PIE-NET [88]	---	---	32.0	3.8
DEF *(Trained on PIE data)*	86.2	451.8	57.1	0.1
ShF [65]	18.0	95.7	**80.9**	0.3
DEF (Ours)	**11.1**	**42.5**	80.02	**0.02**

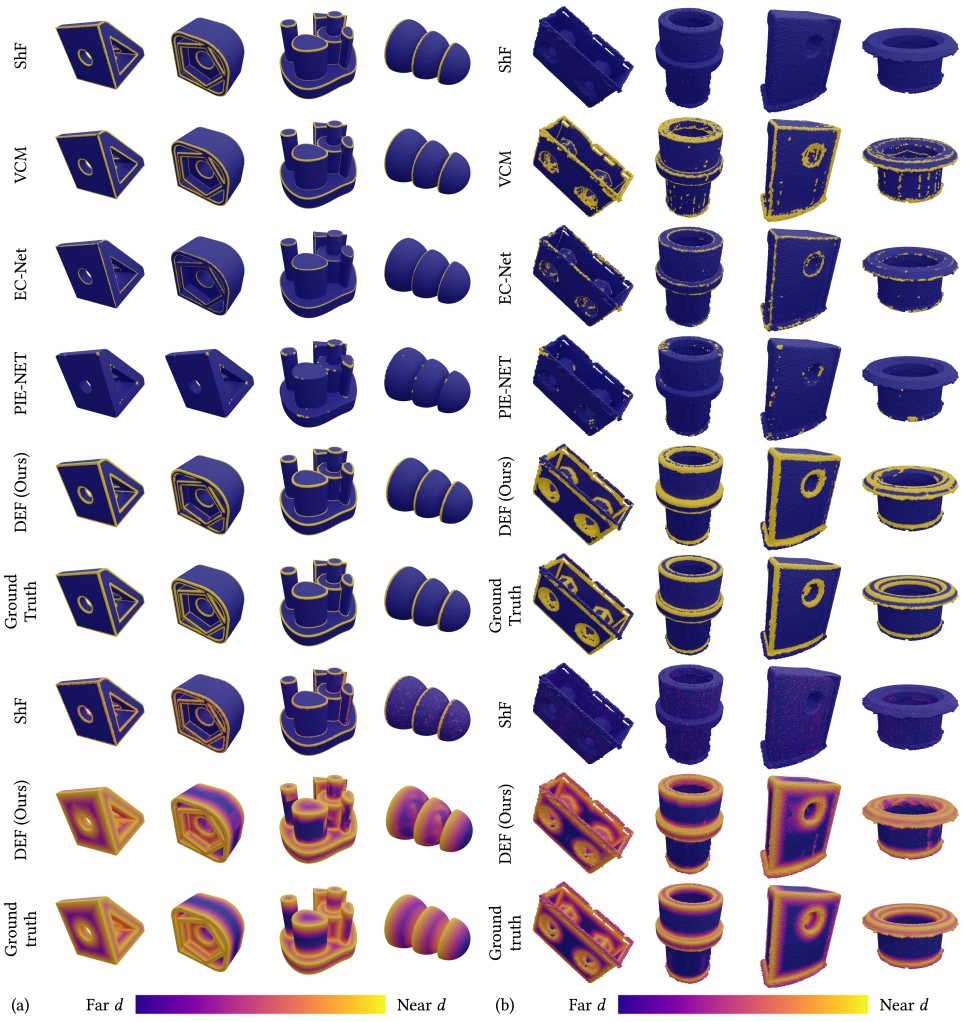

Figure 5: *Qualitative Evaluation of Sharp Feature Line Estimation.* We compare with state-of-the-art methods on (a) high-resolution synthetic full shape datasets and (b) real scanned datasets representing full 3D shapes. Our method robustly reconstructs a point-wise distance-to-feature field and scales to 3D shapes represented by millions of points.

Conclusion

A new approach, DEF, has been proposed for predicting sharp geometric features in 3D models. Traditional methods rely on fitting primitives or estimating the Voronoi covariance measure, which is time-consuming and does not always yield high-quality results. In contrast, DEF trains on large synthetic datasets and a minimal amount of real data, and learns to regress the distance field to features on *local* patches. Due to these two factors, DEF demonstrates excellent generalization and scalability to new, previously unseen 3D shapes of various sizes and variability, even in the presence of scanning noise.

3.3 NPBG++: Accelerating Neural Point-Based Graphics

We address the task of novel view synthesis (NVS), a technology for generating photorealistic views of a scene from a limited set of images. This technique is vital in applications such as virtual and augmented reality, cinematography, and the gaming industry, where it enables the creation of immersive and lifelike environments from sparse data.

Traditional methods like view interpolation and light field rendering [47, 76] have been foundational but often struggle in complex scenarios with detailed geometries and varied lighting. The introduction of Neural Radiance Fields (NeRF) [56] marked a significant advancement, modeling entire scenes with neural networks optimized via differentiable volume rendering. However, NeRF requires substantial computational resources and numerous input views. Voxel-based methods [38] have also been explored, offering a structured representation of 3D scenes but facing challenges in resolution and computational efficiency. Another significant approach is Neural Point-Based Graphics (NPBG) [1], which uses point clouds to model scene geometry and has shown promising results in rendering quality. Yet, these methods, generally necessitate intensive per-scene optimization, which can be time-consuming.

Our model, NPBG++, builds upon and significantly improves the original NPBG framework. By predicting neural descriptors directly from source images in a single pass, our method streamlines the process, eliminating the need for laborious per-scene optimization. This development not only substantially reduces scene fitting time but also enhances rendering quality. NPBG++ offers a significant advancement in efficiency and adaptability for NVS, enabling high-quality rendering in real-time for a diverse range of scenes.

Model Description

Our system generates images from novel views of a static scene using a set of multiview input images, associated camera parameters, and a point cloud. In contrast to NPBG [1], which optimizes neural descriptors for each new scene, our approach adopts a learning-based strategy to predict these descriptors. These neural descriptors represent local geometric and photometric properties and are computed from the input views.

Our system, in contrast to image-based methods that rely on identifying the nearest views from the input image set for generating a novel perspective, constructs a single scene model. This is achieved by processing input views in an online mode, iteratively updating point intermediate states independently of the view count, thereby ensuring constant memory usage. Following the processing of all views, we compute final descriptors from these states. The system comprises two main stages: the modeling stage, where we obtain point descriptors by processing input views, and the rendering process, where descriptors are rasterized and converted into final images using a refiner convolutional network (Figure 6).

In the first, *modeling stage*, we employ a feature extraction process where a U-Net-based network [72] generates a dense feature map for each pixel of the input image while preserving

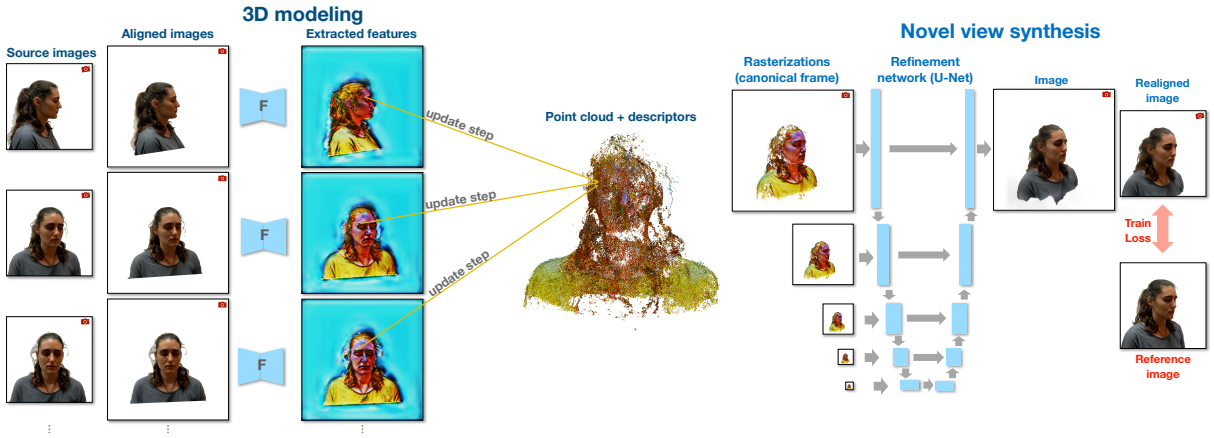

Figure 6: *NPBG++ Overview.* The scene is represented as a point cloud, with each point containing an embedded view-dependent neural descriptor. In the 3D modeling stage, we sequentially process each input view, align the images, extract features, and perform online aggregation to update the neural descriptors, all without requiring fitting. For novel view synthesis, we rasterize the point cloud descriptors and pass the result through a rendering network, followed by post-processing alignment to generate the new view.

its spatial dimensions. We then project points onto the feature map and bilinearly sample the point descriptors. Two important remarks should be noted here. First, the input images undergo an alignment process to ensure consistency across descriptors from different views. This alignment occurs by rotating the input image to a canonical orientation, where the projection of the world's up-axis onto the image plane is vertical (see Figure 6-left). This alignment is crucial because our feature extractor network is not inherently rotation-equivariant. Second, to prevent the updating of descriptors for occluded points, we estimate the visibility of each element by constructing a Z-buffer and rasterizing the point cloud onto a reduced image size. We mark only the points with the minimum Z-value as visible.

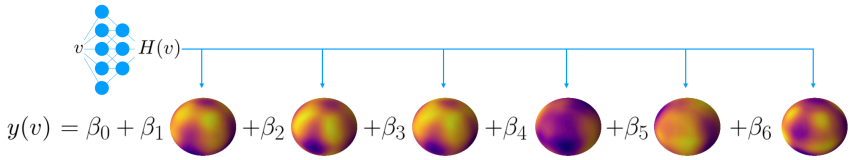

Figure 7: *View-Dependent Neural Descriptor.* The descriptor y: $\mathbb{R}^3 \to \mathbb{R}^c$ is modeled as a linear combination of learnable basis functions over the sphere (H: $\mathbb{R}^3 \to \mathbb{R}^m$), defined by coefficients $\beta_i \in \mathbb{R}^c$ (Equation 3). For each new scene, using a set of source images, we determine β_i for every point.

In the subsequent aggregation phase, we tackle the challenge of processing descriptors from various input views in an online mode, ensuring both memory independence regardless of the number of views and order independence of input frames. We avoid Transformer-based [85], LSTM [33], and GRU [12] networks due to their limitations. Instead, we choose a permutation-invariant method that incorporates view-dependent effects into each point's neural descriptor.

We model this descriptor $y : \mathbb{R}^3 \rightarrow \mathbb{R}^c$ as a linear combination of learnable basis functions over the sphere (Figure 7):

$$\underset{1\times c}{y(v)} = \underset{1\times m}{H(v)}\ \underset{m\times c}{\beta}\ + \underset{1\times c}{\beta_0}\,, \tag{3}$$

where v represents the unit-length view direction, $H : \mathbb{R}^3 \rightarrow \mathbb{R}^m$ is a set of m basis functions (we use $m = 6$), and β and β_0 are coefficients to be determined for each point. This approach is similar to NEX [94], where they model view-dependent RGB values instead of neural descriptors. Unlike NEX, we solve N multivariate linear regression problems to find coefficients β and β_0 for all N points. For each point, we have a set of pairs $\{(v_k, y_k)\}_{k=1}^K$, where K is the number of input views in which we estimate the point to be visible. v_k is a unit-length view direction, and y_k is a sampled descriptor from the input image. Given this, we find the parameters of the descriptor as follows:

$$\underset{1\times c}{\beta_0} = \frac{1}{K}\sum_{k=1}^{K} \underset{1\times c}{y_k}\,, \tag{4}$$

$$\underset{m\times c}{R} := \frac{1}{K}\sum_{k=1}^{K} \underbrace{H(v_k)^T y_k}_{m\times c} - \frac{1}{K}\sum_{k=1}^{K} \underbrace{H(v_k)^T \beta_0}_{m\times c},$$

$$\underset{m\times c}{\beta} = \left(\frac{1}{K}\sum_{k=1}^{K} \underbrace{H(v_k)^T H(v_k)}_{m\times m} + \frac{\alpha}{K}\ \underset{m\times m}{I_m}\right)^{-1} \underset{m\times c}{R}\,, \tag{5}$$

where I_m is the identity matrix, β_0 captures the mean descriptor, and we set the regularizer α=1. When a new descriptor sample y_k arrives we update five intermediate states: K, $\sum_{k=1}^K y_k$, $\sum_{k=1}^K H(v_k)^T y_k$, $\sum_{k=1}^K H(v_k)^T$, $\sum_{k=1}^K H(v_k)^T H(v_k)$. It's important to note that the size of these interim calculations doesn't depend on the number of input views K. For each individual point, we keep updating these calculations until we've processed all the input views. Afterward, we calculate the values for β and β_0. We also remove points from the point cloud that were not visible in any of the input views.

In the second stage, *novel view synthesis stage* of our process, we employ three distinct steps to generate the final image based on specific camera parameters. Initially, we rasterize the calculated descriptors, following a similar approach as NPBG [1]. Then, a refiner network, utilizing a U-Net architecture with gated convolutions [100], processes the rasterization output to address issues such as surface bleeding. Finally, as the last step, we introduce an output image alignment process. This process initially renders the image in a canonical orientation and then rotates it to align with the original query orientation, as depicted in Figure 6-right. This step ensures a consistent appearance, regardless of the camera's y-axis orientation, a factor that has been previously overlooked in methods employing neural descriptors [1, 45, 93].

Our training loss combines a VGG-19 perceptual loss [77], an $\mathcal{L}_1$ loss between down-sampled output and target images to preserve color and prevent detail smoothing, and a novel self-

supervised regularization loss that compares the ground truth image with a rendered image using ground-truth descriptors obtained from the target image.

Empirical Results

In our experimental evaluation, we assessed the effectiveness of our proposed method using several standard metrics for image quality assessment: Structural Similarity Index (SSIM), Peak Signal-to-Noise Ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) [102]. We compare our method's performance with several state-of-the-art neural rendering algorithms, including NPBG [1], NeRF [56], SVS [71], and IBRNet [87]. These comparisons were illustrated quantitatively in Table 3 and qualitatively in Figure 8. We found that our method could produce superior renderings to SVS and on-par results with IBRNet, the leading NVS method for fast generalization to new scenes. In the fine-tuning case, our method outperformed NPBG on all datasets, obtaining leading scores on DTU and H3DS scenes and competing closely on ScanNet and NeRF-Synthetic datasets.

Table 3: *Quantitative evaluations.* For each dataset, we compute the metrics on holdout frames averaged across holdout scenes. Subscript *ft* indicates finetuned versions of the methods. In the case of NPBG++$_{ft}$ we directly finetune coefficients (β, β_0) and the refiner. In the case of NPBG++$_{ft\text{-}system}$ we finetune the feature extractor, aggregator (MLP: neural basis functions), and refiner.

Method	Per scene optimization	Nerf-Synthetic			ScanNet			DTU			H3DS		
		PSNR↑	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**
SVS[71]	✗	22.81	0.919	0.104	23.32	**0.771**	**0.445**	20.98	0.897	0.162	18.96	0.798	0.210
IBRNet[87]	✗	**29.47**	**0.955**	0.157	**23.34**	0.760	0.494	**25.81**	**0.924**	0.231	20.30	0.791	0.279
NPBG++ (Ours)	✗	26.06	0.936	**0.071**	23.11	0.766	0.502	23.23	0.915	**0.154**	**21.80**	**0.818**	**0.177**
NPBG[1]	✓	28.62	0.946	0.058	25.09	0.737	0.459	26.00	0.913	0.125	24.68	0.827	0.146
NeRF[56]	✓	32.49	0.970	**0.041**	**25.74**	**0.780**	0.537	**26.92**	0.913	0.198	23.88	0.833	0.178
SVS$_{ft}$[71]	✓	23.37	0.919	0.101	22.31	0.610	0.543	20.72	0.864	0.190	20.12	0.770	0.197
IBRNet$_{ft}$[87]	✓	**32.51**	**0.972**	0.144	24.42	0.774	0.493	23.80	0.917	0.222	24.68	**0.850**	0.195
NPBG++$_{ft\text{-}system}$ (Ours)	✓	26.24	0.940	0.064	23.48	0.768	0.490	24.05	0.919	0.147	23.79	0.836	0.155
NPBG++$_{ft}$ (Ours)	✓	28.67	0.952	0.050	25.27	0.772	**0.448**	26.08	**0.928**	**0.123**	**24.91**	0.845	**0.137**

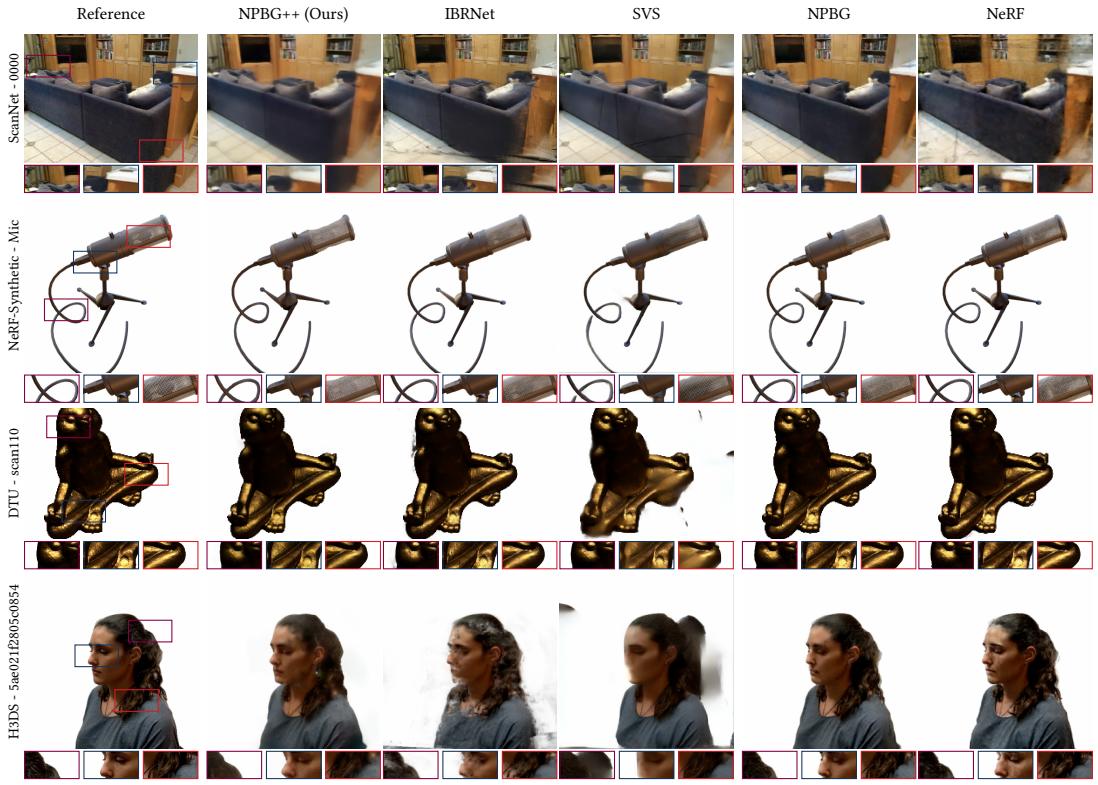

Figure 8: *Qualitative evaluations.* Comparisons with otimization-based approaches (NPBG[1], NeRF[56]) and learning based approaches (IBRNet[87], SVS[71]) on ScanNet[15], NeRF-Synthetic[56], DTU[37], H3DS[69] scenes.

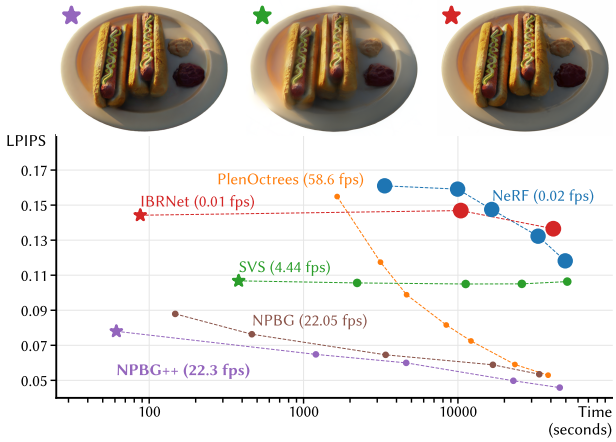

Figure 9: *Runtime vs. Image Quality.* Comparison of several methods, computed on the `hot-dog` scene from the NeRF-synthetic dataset. The time axis represents the time-to-rendering, i.e., fitting time + rendering time for one image. For methods marked with $\star$, the first scores are reported without per-scene optimizations. Fitting time consists of feature extraction for IBRNet, geometry estimation + 3D modeling stage for NPBG++, and geometry estimation + meshing for SVS (the renderings on top offer qualitative comparisons between these configurations). The remaining scores are computed at different points in the fine-tuning processes. Circle areas are proportional to logarithms of the rendering times (smaller is better) and highlight the methods' rendering speed.

We also conducted runtime analysis, comparing the speed of several advanced methods across two stages of inference, as illustrated in Figure 9. The first stage involves algorithms capturing data from source images. This includes training neural representations for some methods, running feature extractors for IBRNet, and 3D modeling for our approach. SVS, NPBG, and NPBG++ also construct 3D representations at this stage. The time for these processes is counted in our comparisons. This stage occurs once per scene. The second stage involves rendering novel views. NeRF and IBRNet, in particular, show high rendering times. IBRNet's rendering time exceeds the entire fitting process of our method. NeRF, PlenOctrees, and NPBG require per-scene fitting, leading to longer durations for quality results. Our model, without optimization, has the shortest overall time-to-render, outperforming SVS and IBRNet, which are hindered by surface estimation and rendering time, respectively.

Conclusion

In conclusion, the proposed NPBG++ model significantly improves generalization in the task of novel view synthesis, meaning that the model can perform well on diverse, previously unseen scenes without per-scene optimization. By predicting neural descriptors directly from the original images in a single pass, NPBG++ avoids the laborious optimization process for new scenes. This innovation allows the model to quickly adapt to new environments and enables the rapid creation of high-quality renderings while maintaining high rendering speed.

3.4 Making DensePose fast and light

We address the task of DensePose estimation [3], which involves understanding human forms in images through dense image-to-surface correspondences. This task is crucial for a variety of computer vision applications, such as augmented reality and virtual cloth fitting, where precise human body modeling is essential. The DensePose task involves predicting UV coordinates for each pixel of the human form, mapping it to a 3D model like the Skinned Multi-Person Linear (SMPL) model [51].

However, existing solutions in this field, such as the DensePose R-CNN [3] and Parsing R-CNN [98], are heavily parameterized, making them unsuitable for deployment on mobile or embedded devices. These models require robust server-side infrastructure and stable internet connectivity, limiting their practical applicability. Moreover, the followup works [62, 61] have improved the quality of the results, but none have specifically focused on optimizing model size and speed for mobile deployment. These limitations highlight a critical gap in making DensePose estimation more accessible and widely usable in real-time applications.

In response to these challenges, our work introduces a novel architecture, Mobile Parsing R-CNN, that is designed to be both lightweight and efficient, enabling real-time DensePose estimation on mobile devices. We have meticulously restructured the DensePose R-CNN model, incorporating several deep learning innovations and model quantization methods. Our archi-

tecture represents a significant advancement in the field, achieving a $17\times$ reduction in model size and a $2\times$ improvement in latency compared to the baseline model. This breakthrough opens up new possibilities for deploying advanced computer vision applications directly on end-user devices, without the need for extensive hardware or internet dependency.

Model Description

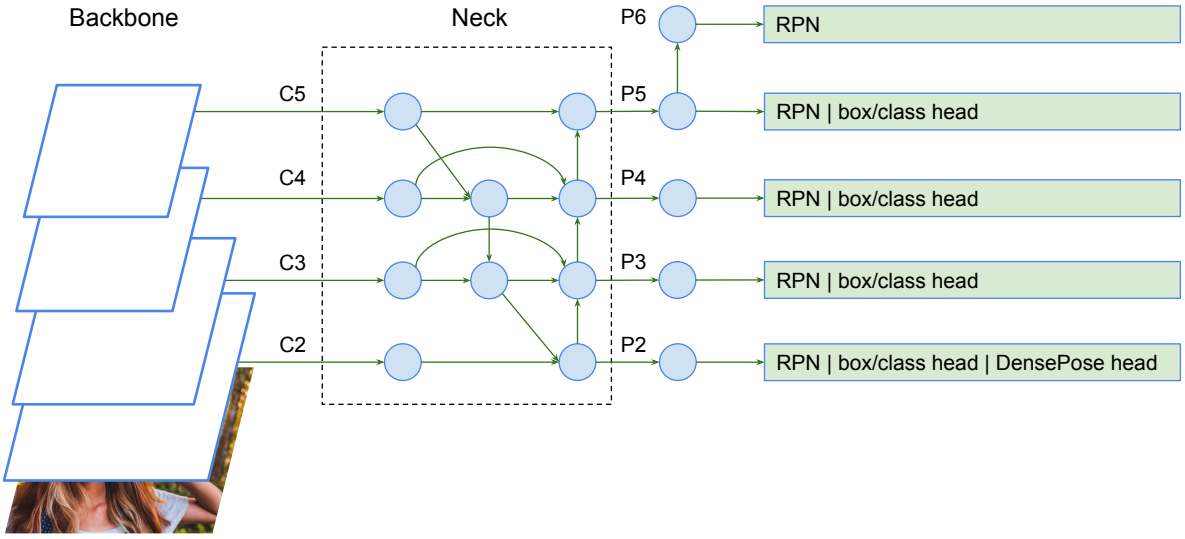

Figure 10: *High-level Structure.* C_i, P_i represent feature levels with a resolution of $1/2^i$ of the input image. P_6 is obtained via stride-2 pooling on P_5.

In our development of the Mobile Parsing R-CNN model, we are inspired by the Parsing R-CNN model, renowned for winning the COCO 2018 Challenge DensePose Estimation task. The network architecture fundamentally relies on a two-stage R-CNN detection pipeline, which includes a backbone for feature extraction, a neck for further refining these features, a Region Proposal Network (RPN) to generate object proposals, a Box Head for object classification and bounding box regression, and a DensePose Head dedicated to detailed pose estimation (see Figure 10).

For the backbone of our model, we emphasize efficient design, aligning with structures akin to MobileNetV1 and V2 [35, 73], marked by depth-wise separable convolutions. Our exploration includes various architectures such as MobileNetV3 [34], which integrates Squeeze and excitation block and non-linearities; MixNet [81], offering a multi-kernel variant; Differentiable neural architecture search (NAS), where models like MnasNet [79], FBNet [95], and Single-Path [78] are considered; EfficientNets [80], balancing accuracy and network size; and CondConv [97], featuring dynamic kernel weights in convolutional layers.

In the neck section, our choice is the bidirectional FPN (BiFPN) [82] for multi-scale feature fusion, which has shown superior performance in object detection tasks while remaining lightweight and efficient. This is partly attributed to the use of separable convolutions.

The Densepose head sees an enhancement in the region of interest (RoI) resolution, increased from 14×14 to 32×32 as suggested in [98]. Here, we employ the atrous spatial pyramid pooling (ASPP) module [11], followed by convolutional layers, and deliberately exclude the non-local convolutional layer [89] to reduce network latency.

Empirical Results

In our experimental evaluation of the Mobile Parsing R-CNN models, we primarily used the Average Precision (AP) at various geodesic point similarity (GPS) thresholds, alongside box average precision, to assess the performance. We implemented and modified the Parsing R-CNN model using PyTorch and Detectron2 [96]. Our experiments involved an extensive ablation study on different model components, including the backbone network, neck type, and the number of channels in the model, as detailed in Tables 4, 5, and 6.

Table 4: *The main differences between the models presented.* Results on DensePose-COCO minival. 3x LR refers to 3 times longer training compared to the default setting. P_i represents a feature level with a resolution of $1/2^i$ of the input images. #Channels represent the number of channels inside *neck* and *heads*.

	DensePose R-CNN (baseline) [3]	Parsing R-CNN [98]	Mobile Parsing R-CNN (A)	Mobile Parsing R-CNN (B)
Backbone	ResNet-50 [29]	ResNet-50 [29]	Single-Path [78]	Single-Path [78]
Neck	FPN[49]	FPN[49]	FPN[49]	BiFPN[82]
RoI resolution	14×14	32×32	32×32	32×32
Pooling Type	RoIPool	RoIPool	RoIAlign	RoIAlign
Box/class head	2 linear layers	2 linear layers	2 conv layers	2 conv layers
Feature level for prediction	P_2,P_3,P_4,P_5	P_2	P_2	P_2
DensePose head	8 conv layers	ASPP[11]+NL[89]+4 conv layers	ASPP[11]+4 conv layers	ASPP[11]+4 conv layers
#Channels	512	512	256	64
#Params	59.73M	54.36M	11.35M	3.35M
GPU FPS	13.16	10.15	12.03	22.77 (3x LR: **23.55**)
CPU FPS	1.62	1.39	1.42	2.02 (3x LR: **2.10**)
box AP	57.8	59.609	56.370	55.39 (3x LR: 56.83)
densepose AP	49.8	54.676	49.512	46.79 (3x LR: 51.08)

We found that Mobile Parsing R-CNN variants (A) and (B) balance average precision and computational efficiency, showing significant FPS improvements on CPU and GPU, as demonstrated in Figure 11.

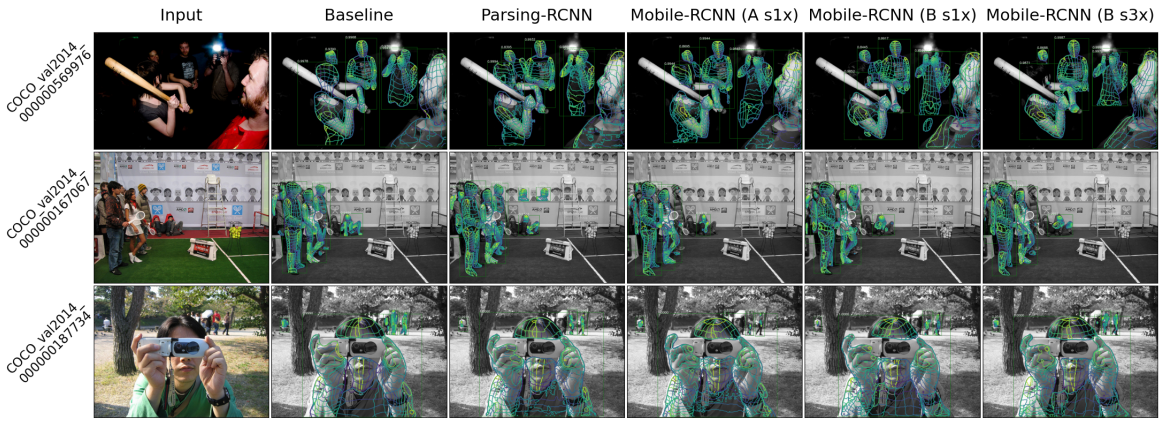

Figure 11: *Qualitative comparison of different models.* We depict contours with color-coded U and V coordinates as an output of the model.

Table 5: *Ablation on the backbone network used in Mobile Parsing R-CNN (A).* The backbones are sorted by top-1 accuracy. Results on DensePose-COCO *minival*.

Backbone	Top-1 Accuracy (%)	#Params	box AP	dp. AP	GPU FPS	CPU FPS
ResNet-50 [29]	77.15	33.61M	60.0	**54.7**	11.05	1.34
EfficientNet-B3 [80]	81.636	16.03M	59.027	53.084	8.31	1.37
EfficientNet-EdgeTPU-L [21]	80.534	17.89M	60.069	53.378	8.11	1.34
MixNet-XL [81]	80.120	19.10M	58.444	51.475	8.54	1.32
EfficientNet-B2 [80]	79.688	13.68M	58.041	51.800	9.33	1.38
MixNet-L [81]	78.976	14.62M	57.481	50.649	8.52	1.34
EfficientNet-EdgeTPU-M [21]	78.742	14.57M	58.825	52.302	9.21	1.37
EfficientNet-B1 [80]	78.692	13.03M	57.654	51.053	9.49	1.39
CondConv-EfficientNet-B0 [21, 97]	77.304	18.32M	56.779	49.231	10.63	1.40
EfficientNet-EdgeTPU-S [21]	77.264	13.12M	58.296	51.606	10.03	1.39
MixNet-M [81]	77.256	12.39M	56.834	48.371	9.39	1.35
EfficientNet-B0 [80]	76.912	12.10M	56.271	49.647	10.53	1.39
MixNet-S [81]	75.988	11.52M	55.132	46.685	10.34	1.37
MobileNetV3-Large-1.0 [34]	75.516	12.04M	54.537	47.195	11.54	1.40
MnasNet-A1 [81]	75.448	10.94M	54.648	47.036	11.21	1.38
FBNet-C [95]	75.124	11.49M	55.399	47.983	10.97	1.37
MnasNet-B1 [79]	74.658	11.31M	52.280	47.658	11.24	1.37
Single-Path [78]	74.084	11.35M	56.370	49.512	**12.03**	**1.42**
MobileNetV3-Large-0.75 [34]	73.442	10.92M	52.763	44.736	11.02	1.36
MobileNetV3-Large-1.0 (minimal) [34]	72.244	10.48M	52.464	44.632	11.33	1.36
MobileNetV3-Small-1.0 [34]	67.918	10.07M	49.614	35.808	10.62	1.35
MobileNetV3-Small-0.75 [34]	65.718	9.74M	44.224	32.650	10.16	1.33
MobileNetV3-Small-1.0 (minimal) [34]	62.898	9.58M	45.989	36.522	10.34	1.34

Table 6: *Ablation on neck type and number of channels.* The number of channels is the same in neck and heads. Results on DensePose-COCO *minival*.

	Neck	#channels	#Params	box AP	dp. AP	GPU FPS	CPU FPS
Mobile Parsing R-CNN (A)	FPN	256	11.35M	56.371	49.512	12.03	1.42
	BiFPN	256	10.53M	58.106	52.80	12.05	1.41
	BiFPN	112	4.41M	56.41	49.64	19.04	1.78
	BiFPN	88	3.82M	56.08	48.19	20.43	1.87
Mobile Parsing R-CNN (B)	BiFPN	64	3.35M	55.39	46.79	22.77	2.02

Conclusion

The study is dedicated to improving the generalization ability of the DensePose model for dense human pose estimation under strict constraints on model size and speed. The improvement was achieved through careful selection of components in various parts of the model, such as the choice of the optimal backbone network for feature extraction, the neck architecture, and the architectures of the heads for human detection and DensePose prediction. Thanks to these innovations, the model became faster and more efficient, which subsequently allowed it to run locally on a mobile device.

3.5 Multi-NeuS: 3D Head Portraits from Single Image with Neural Implicit Functions

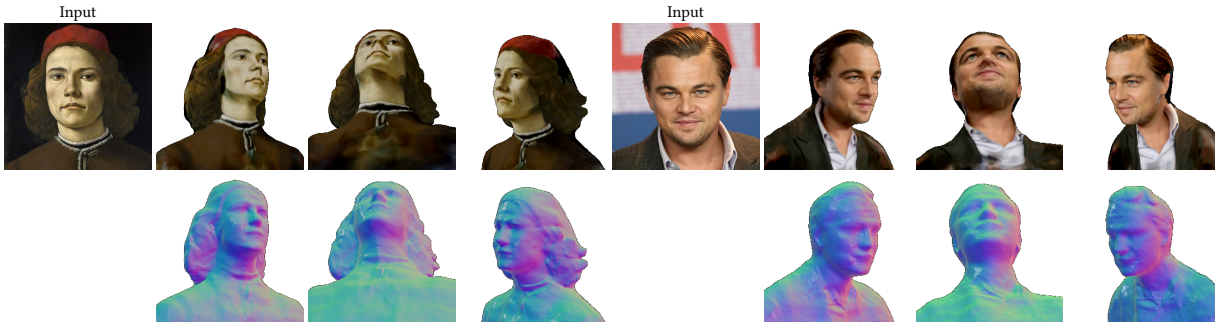

Figure 12: *3D Head Portrait Reconstruction from Single Image.* Multi-NeuS can create realistic 3D head portraits from single in-the-wild photos or paintings.

We address the challenge of 3D portraiture, specifically the automatic acquisition of textured 3D human head models. This task is crucial in various fields such as filmmaking, augmented reality (AR), virtual reality (VR), extended reality (XR), and the gaming industry. It aims to capture both the geometric and textural details of human heads, bypassing the need for labor-intensive and time-consuming manual model creation. The importance of this task lies in its potential to revolutionize content creation in these domains, providing more realistic and immersive experiences.

In the realm of head appearance modeling, several methods have been developed, focusing mainly on 2D representations [39, 17, 74]. For 3D modeling, existing approaches like H3D-Net and NeuralHeadAvatars [70, 26] often rely on 3D scans or synthetic data, limiting their practicality. These methods, while groundbreaking, exhibit limitations in data acquisition and generalizability. The recent introduction of NeuS and related methods like UNISURF and VolSDF [86, 63, 99, 41] have opened new avenues in implicit representations for shape and appearance, but still face challenges in adapting to individual instances effectively.

To overcome these limitations, we propose *Multi-NeuS*, an innovative neural implicit architecture that efficiently fits multiple objects of the same class (human heads) and reconstructs their surfaces from sets of multi-view photos. Multi-NeuS extends the NeuS framework [86], employing a shared parameter subset across various training videos. This unique approach facilitates rapid learning from a minimal dataset and demonstrates impressive data-efficiency and speed. Unlike its predecessors, Multi-NeuS can generate high-quality 3D head portraits from a single or few photographs, marking a significant advancement in the field of 3D portraiture (Figure 12).

Model Description

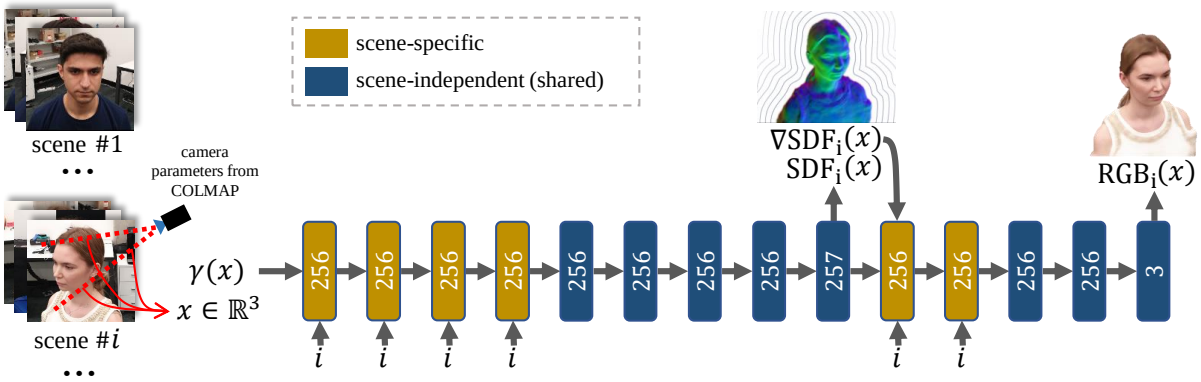

Figure 13: *Multi-NeuS Overview.* Our model, a 3D neural implicit function, can represent multiple objects of the same class. Shared layers (blue) capture class priors for few-shot reconstruction in novel scenes of the same class. It's trained using volumetric rendering and pixelwise loss on a dataset of multiple scenes. When fitting to a new object, scene-specific layers (yellow) are adjusted first, followed by fine-tuning all layers.

In our study, we introduce Multi-NeuS, a novel 3D neural implicit function designed for reconstructing multiple objects of a specific class simultaneously. Building upon the principles of NeuS [86] and NeRF [57], Multi-NeuS aims to overcome the limitations in scenarios where only single or few images are available for reconstruction. We focus on the class of human heads for our implementation.

We begin by reviewing the NeuS reconstruction method, a modification of NeRF tailored for non-transparent objects, specifically focusing on modeling the object's surface. NeuS rep-

resents the object's surface as the zero-level set of a signed distance function (SDF), defined as $\{x \in \mathbb{R}^3 \mid \text{SDF}(x) = 0\}$. Two neural networks are used to model the SDF and RGB radiance at any 3D point, with density defined as a bell-shaped function of the SDF, peaking at zero, which corresponds to the object's surface. These networks are optimized through differentiable volume rendering, enabling both 3D surface reconstruction and novel view synthesis.

Our Multi-NeuS method expands NeuS to manage multiple scenes, incorporating shared and scene-specific layers into its architecture (refer to Figure 13). This design aids in transferring learned class priors to new scenes, improving few-shot reconstruction.

For the training process, Multi-NeuS undergoes two main stages: meta-learning and fitting (Figure 14). During meta-learning, we pre-train the model on a dataset of multi-view images from multiple scenes, allowing Multi-NeuS to learn a general representation of the class. Subsequently, in the fitting stage, we add and optimize scene-specific layers for new, unseen objects, starting with an initialization representing an 'average' object from the dataset.

Next, we address the task of applying the model to in-the-wild images, such as those taken from the Internet. The model requires both extrinsic and intrinsic camera parameters, which are typically unknown for random pictures.

First, we manually initialize an intrinsics matrix by averaging the parameters of the cameras used during the meta-learning stage. To reduce potential initialization errors, all images from the training, validation, and in-the-wild sets are cropped around the face area with a margin. To find the extrinsic camera parameters, we use the 3D coordinates of facial landmarks, relative to which all training examples were aligned during the meta-learning stage. For in-the-wild photographs, we predict the 2D coordinates of facial landmarks using an off-the-shelf detector [7]. Given the 2D positions of the landmarks, the corresponding 3D coordinates, and the intrinsics matrix, the extrinsic camera parameters can be estimated using PnP [14].

The obtained estimate is quite coarse, so during training, the extrinsics matrix is multiplied by a correction matrix parameterized using $\mathfrak{se}(3)$ Lie algebra, following [48]. To correct the intrinsic camera parameters, we set learnable coefficients that multiply the focal lengths of the matrix. All new parameters are optimized alongside the network parameters in each training iteration using stochastic optimization.

Empirical Results

The training of our model utilized a subset of the SmartPortraits dataset [43], consisting of smartphone videos. We extracted frames and camera parameters using COLMAP software [75], ensuring that the scenes were aligned for consistent 3D coordinates.

We evaluate single-view mesh reconstruction and provide both quantitative (Table 7) and qualitative (Figure 15) comparisons. Our method, Multi-NeuS, exhibited comparable performance to H3D-Net [70], even though it was trained on a different dataset with far less data and without access to 3D scans. Results on in-the-wild photographs are showcased in Figure 16.

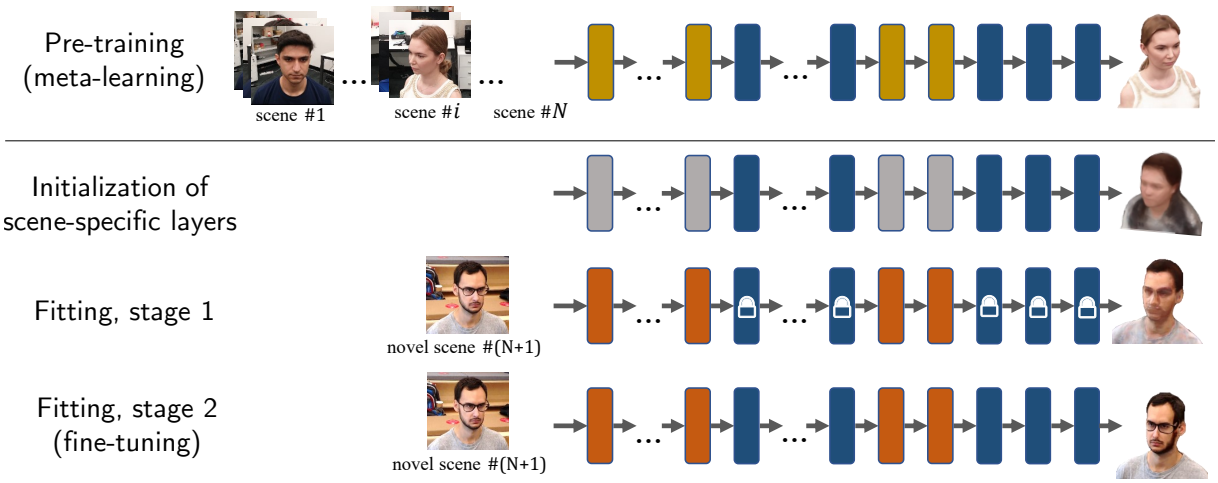

Figure 14: *The training stages of Multi-NeuS.* Row 1: The model is trained to represent N scenes, including scene-specific layers. Row 2: Scene-specific layers are reset using weighted aggregation for fitting a new scene. Row 3: Only scene-specific layers are fine-tuned for new individuals with limited images. Row 4: All layers are fine-tuned with reduced learning rate.

Figure 15: *Single-view Mesh Reconstruction Comparison.* We evaluate on the first four scenes of the H3DS dataset. H3D-Net [70], originally designed for three-view reconstruction, can also be assessed in one-shot mode. H3D-Net was trained on 10,000 3D scans from the same distribution as these test examples, while our method, trained on a hundred smartphone videos, achieves comparable quality. Furthermore, our approach demonstrates a smaller identity gap and a less pronounced regression-to-mean effect.

Table 7: *Mesh Reconstruction Quantitative Results.* We compare with H3D-Net [70] on the H3DS dataset [70]. We calculate the unidirectional Chamfer distance in millimeters, measured post alignment with the ground truth via the Iterative Closest Point (ICP) method [6]. This metric was applied to both facial areas and full head meshes. Lower values indicate better performance. 'F/L/R' represents the input views: 'frontal/left/right'.

	face				head			
Input view	*F*	*L*	*R*	*mean*	*F*	*L*	*R*	*mean*
H3D-Net 3-view	-	-	-	1.34	-	-	-	10.53
H3D-Net 1-view	**1.82**	1.83	1.91	1.85	13.83	**13.01**	12.51	13.12
Ours 1-view	1.89	**1.77**	**1.86**	**1.84**	**13.00**	13.27	**11.95**	**12.74**

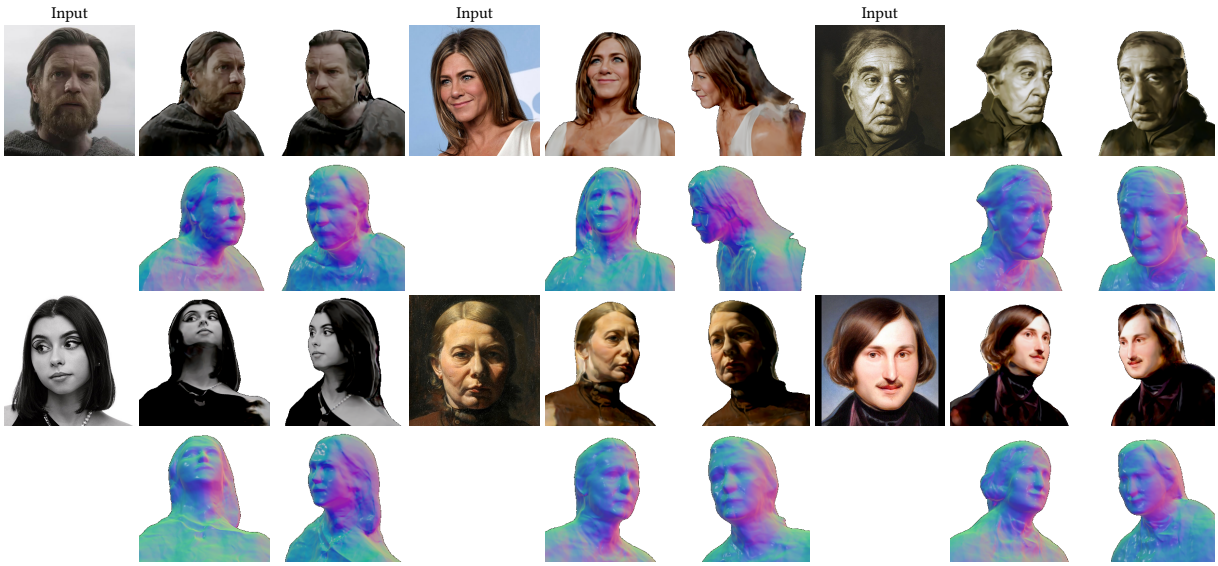

Figure 16: *3D reconstruction of in-the-wild photographs and paintings.* Our method handles diverse hair styles and performs well on images beyond the SmartPortraits dataset. Potential artifacts on the back are due to limited training data angles.

Conclusion

This work presents *Multi-NeuS*, a new approach for reconstructing 3D head portraits from one or more images, improving generalization in 3D computer vision tasks. Generalization, or the model's ability to accurately reconstruct *new* faces from one or a few photographs, is achieved by incorporating priors through pre-training on a large dataset of various individuals' images. This pre-training step enables the model to capture class-specific features, reducing the need for lengthy optimization for each scene. By combining fitting general parameters with adaptation to specific scenes, Multi-NeuS effectively reconstructs textured surfaces. The method has limitations, primarily due to the limited diversity of the training dataset. In the future, expanding the dataset and making architectural improvements will further enhance the approach's capabilities.

4 Conclusion

In this dissertation, we propose methods to improve the generalization capabilities of models in 3D computer vision tasks. All presented methods aim to enhance the efficiency and accuracy of model performance in diverse, previously unseen conditions, which is a key factor for the successful application of these technologies in real-world scenarios.

In the first study, we introduce a video generation model based on modeling videos in a discrete latent space. The uniqueness of this approach lies in its ability to generate video sequences from previously unseen input conditional frames, achieved with significantly fewer computational resources compared to existing methods. Using only eight V100 graphics processors for model training, while alternative approaches require up to 512 tensor processors, demonstrates a significant improvement in efficiency without compromising generalization quality.

In the second study, we propose a new DEF method for predicting sharp geometric features in 3D models. Unlike traditional methods that rely on fitting primitives or estimating the Voronoi covariant measure, DEF uses training on large synthetic datasets with minimal real data. The method learns to regress the distance field to features on local patches, which enhances generalization ability and scalability to new, previously unseen 3D shapes, even in the presence of scanning noise.

In the third study, we focus on the NPBG++ model, which significantly improves generalization in the task of novel view synthesis. This model predicts neural descriptors directly from input images in a single pass, avoiding laborious optimization on a new scene. This innovation allows the model to quickly adapt to new environments, creating high-quality renderings at a high visualization speed, making it efficient compared to existing approaches.

In the fourth study, we achieve significant improvement in the generalization ability of the DensePose model for dense human pose estimation under strict size and speed constraints. By optimizing various components of the model, such as the feature extraction backbone, the neck and head architectures for human detection and DensePose prediction, we improve model performance and quality, ultimately enabling it to run locally on a mobile device.

Finally, in the fifth study, we present the Multi-NeuS approach for reconstructing 3D head portraits from one or several images. Improvement in generalization ability is achieved through pre-training the model on a large set of images of different people, allowing it to capture class-specific features and reduce the need for prolonged optimization for each scene. By combining the optimization of general parameters with scene-specific adaptation, Multi-NeuS effectively reconstructs textured surfaces.

Thus, all the methods we present in this work demonstrate significant improvements in the generalization capabilities of models in 3D computer vision tasks. Each proposed solution not only surpasses existing approaches in efficiency and accuracy but also ensures broader applicability in various practical tasks, such as synthetic data generation, accurate 3D reconstruction, efficient novel view synthesis, and human pose estimation. These achievements underscore

the importance and significance of the developed methods, opening new possibilities for the further development of 3D computer vision technologies.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23--28, 2020, Proceedings, Part XXII 16*, pages 696--712. Springer, 2020.

[2] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Detailed human avatars from monocular video. In *2018 International Conference on 3D Vision (3DV)*, pages 98--109. IEEE, 2018.

[3] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297--7306, 2018.

[4] Linchao Bao, Xiangkai Lin, Yajing Chen, Haoxian Zhang, Sheng Wang, Xuefei Zhe, Di Kang, Haozhi Huang, Xinwei Jiang, Jue Wang, et al. High-fidelity 3d digital human head creation from rgb-d selfies. *ACM Transactions on Graphics (TOG)*, 41(1):1--21, 2021.

[5] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586--606. Spie, 1992.

[6] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239--256, 1992.

[7] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In *International Conference on Computer Vision*, 2017.

[8] Egor Burkov, Ruslan Rakhimov, Aleksandr Safin, Evgeny Burnaev, and Victor Lempitsky. Multi-neus: 3d head portraits from single image with neural implicit functions. *IEEE Access*, 2023.

[9] Yuanhao Cao, Liangliang Nan, and Peter Wonka. Curve networks for surface reconstruction. *arXiv preprint arXiv:1603.08753*, 2016.

[10] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834--848, 2017.

[12] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[13] A Clark, J Donahue, and K Simonyan. Adversarial video generation on complex datasets. *arXiv preprint arXiv:1907.06571*, 2019.

[14] Toby Collins and Adrien Bartoli. Infinitesimal plane-based pose estimation. *International Journal of Computer Vision*, 109(3):252–286, sep 2014.

[15] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[16] Kris Demarsin, Denis Vanderstraeten, Tim Volodine, and Dirk Roose. Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Computer-Aided Design*, 39(4):276--283, 2007.

[17] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690--4699, 2019.

[18] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018.

[19] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018.

[20] Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*, 2017.

[21] Efficientnet-edgetpu: Creating accelerator-optimized neural networks with automl. https://ai.googleblog.com/2019/08/efficientnet-edgetpu-creating.html.

[22] Yuki Endo, Yoshihiro Kanamori, and Shigeru Kuriyama. Animating landscape: self-supervised learning of decoupled motion and appearance for single-image video synthesis. *arXiv preprint arXiv:1910.07192*, 2019.

[23] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64--72, 2016.

[24] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1--8. IEEE, 2007.

[25] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18653--18664, 2022.

[26] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Niessner, and Justus Thies. Neural head avatars from monocular rgb videos. In *Proc. CVPR*, 2022.

[27] T. Hackel, J. D. Wegner, and K. Schindler. Contour detection in unstructured 3d point clouds. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1610--1618, 2016.

[28] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770--778, 2016.

[30] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626--6637, 2017.

[31] Chems-Eddine Himeur, Thibault Lejemble, Thomas Pellegrini, Mathias Paulin, Loic Barthe, and Nicolas Mellado. Pcednet: A lightweight neural network for fast and interactive edge detection in 3d point clouds. *ACM Transactions on Graphics (TOG)*, 41(1):1--21, 2021.

[32] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[33] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735--1780, 1997.

[34] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1314--1324, 2019.

[35] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[36] Ehsan Imani and Martha White. Improving regression performance with distributional losses. In *International conference on machine learning*, pages 2157--2166. PMLR, 2018.

[37] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406--413, 2014.

[38] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[39] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020.

[40] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, page 0, 2006.

[41] Petr Kellnhofer, Lars C. Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *Proc. CVPR*, June 2021.

[42] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9601--9611, 2019.

[43] Anastasiia Kornilova, Marsel Faizullin, Konstantin Pakulev, Andrey Sadkov, Denis Kukushkin, Azat Akhmetyanov, Timur Akhtyamov, Hekmat Taherinejad, and Gonzalo Ferrer. Smartportraits: Depth powered handheld smartphone dataset of human portraits for state estimation, reconstruction and synthesis. In *Proc. CVPR*, June 2022.

[44] Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. Videoflow: A flow-based generative model for video. *arXiv preprint arXiv:1903.01434*, 2(5), 2019.

[45] Christoph Lassner and Michael Zollhofer. Pulsar: Efficient sphere-based neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1440--1449, 2021.

[46] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[47] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31--42, 1996.

[48] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proc. ICCV*, 2021.

[49] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117--2125, 2017.

[50] Y. Lin, C. Wang, B. Chen, D. Zai, and J. Li. Facet segmentation-based line segment extraction for large-scale point clouds. *IEEE Transactions on Geoscience and Remote Sensing*, 55(9):4839--4854, 2017.

[51] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1--248:16, October 2015.

[52] Pauline Luc, Aidan Clark, Sander Dieleman, Diego de Las Casas, Yotam Doron, Albin Cassirer, and Karen Simonyan. Transformation-based adversarial video prediction on large-scale data. *arXiv preprint arXiv:2003.04035*, 2020.

[53] Albert Matveev, Ruslan Rakhimov, Alexey Artemov, Gleb Bobrovskikh, Vage Egiazarian, Emil Bogomolov, Daniele Panozzo, Denis Zorin, and Evgeny Burnaev. Def: Deep estimation of sharp geometric features in 3d shapes. *ACM Transactions on Graphics*, 41(4), 2022.

[54] Jacob Menick and Nal Kalchbrenner. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. *arXiv preprint arXiv:1812.01608*, 2018.

[55] Quentin Mérigot, Maks Ovsjanikov, and Leonidas J Guibas. Voronoi-based curvature and feature estimation from point clouds. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):743--756, 2010.

[56] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405--421. Springer, 2020.

[57] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, 2020.

[58] Theo Moons, Luc Van Gool, Maarten Vergauwen, et al. 3d reconstruction from multiple images part 1: Principles. *Foundations and Trends® in Computer Graphics and Vision*, 4(4):287--404, 2010.

[59] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255--1262, 2017.

[60] Seonghyeon Nam, Chongyang Ma, Menglei Chai, William Brendel, Ning Xu, and Seon Joo Kim. End-to-end time-lapse video synthesis from a single outdoor image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1409--1418, 2019.

[61] Natalia Neverova, David Novotny, and Andrea Vedaldi. Correlated uncertainty for learning dense correspondences from noisy labels. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 920--928. Curran Associates, Inc., 2019.

[62] Natalia Neverova, James Thewlis, Riza Alp Guler, Iasonas Kokkinos, and Andrea Vedaldi. Slim densepose: Thrifty learning from sparse annotations and motion cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10915--10923, 2019.

[63] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proc. ICCV*, 2021.

[64] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099--5108, 2017.

[65] Prashant Raina, Sudhir Mudur, and Tiberiu Popa. Sharpness fields in point clouds using deep learning. *Computers & Graphics*, 78:37--53, 2019.

[66] Ruslan Rakhimov, Andrei-Timotei Ardelean, Victor Lempitsky, and Evgeny Burnaev. Npbg++: Accelerating neural point-based graphics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15969--15979, 2022.

[67] Ruslan Rakhimov, Emil Bogomolov, Alexandr Notchenko, Fung Mao, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Making densepose fast and light. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1869--1877, 2021.

[68] Ruslan Rakhimov*, Denis Volkhonskiy*, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Latent video transformer. *VISAPP 2021: 16th International Conference on Computer Vision Theory and Applications*, 2021.

[69] Eduard Ramon, Gil Triginer, Janna Escur, Albert Pumarola, Jaime Garcia, Xavier Giro-i Nieto, and Francesc Moreno-Noguer. H3d-net: Few-shot high-fidelity 3d head reconstruction. *arXiv preprint arXiv:2107.12512*, 2021.

[70] Eduard Ramon, Gil Triginer, Janna Escur, Albert Pumarola, Jaime Garcia, Xavier Giro-i Nieto, and Francesc Moreno-Noguer. H3d-net: Few-shot high-fidelity 3d head reconstruction. In *Proc. ICCV*, 2021.

[71] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12216--12225, 2021.

[72] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234--241. Springer, 2015.

[73] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510--4520, 2018.

[74] Sergio Saponara, Abdussalam Elhanashi, and Alessio Gagliardi. Reconstruct fingerprint images using deep learning and sparse autoencoder algorithms. In *Real-Time Image Processing and Deep Learning 2021*, volume 11736, pages 9--18. SPIE, 2021.

[75] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016.

[76] Harry Shum and Sing Bing Kang. Review of image-based rendering techniques. In *Visual Communications and Image Processing 2000*, volume 4067, pages 2--13. International Society for Optics and Photonics, 2000.

[77] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[78] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. Single-path nas: Designing hardware-efficient convnets in less than 4 hours. *arXiv preprint arXiv:1904.02877*, 2019.

[79] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820--2828, 2019.

[80] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[81] Mingxing Tan and Quoc V Le. Mixconv: Mixed depthwise convolutional kernels. *arXiv preprint arXiv:1907.09595*, 2019.

[82] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. *arXiv preprint arXiv:1911.09070*, 2019.

[83] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21--22, 1999 Proceedings*, pages 298--372. Springer, 2000.

[84] Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306--6315, 2017.

[85] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998--6008, 2017.

[86] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Proc. NeurIPS*, 2021.

[87] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690--4699, 2021.

[88] Xiaogang Wang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang. Pie-net: Parametric inference of point cloud edges. *Advances in Neural Information Processing Systems*, 33, 2020.

[89] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794--7803, 2018.

[90] Christopher Weber, Stefanie Hahmann, and Hans Hagen. Sharp feature detection in point clouds. In *2010 Shape Modeling International Conference*, pages 175--186. IEEE, 2010.

[91] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724--4732, 2016.

[92] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. *arXiv preprint arXiv:1906.02634*, 2019.

[93] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467--7477, 2020.

[94] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534--8543, 2021.

[95] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734--10742, 2019.

[96] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[97] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems*, pages 1305--1316, 2019.

[98] Lu Yang, Qing Song, Zhihui Wang, and Ming Jiang. Parsing r-cnn for instance-level human analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 364--373, 2019.

[99] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Proc. NeurIPS*, 2021.

[100] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4471--4480, 2019.

[101] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Ec-net: an edge-aware point set consolidation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 386--402, 2018.

[102] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.