VSCHH 2023: A Benchmark for the View Synthesis Challenge of Human Heads (Supplementary Material)

Youngkyoon Jang[§†1], Jiali Zheng[§†1], Jifei Song[†2], Helisa Dhamo[†2], Eduardo Pérez-Pellitero[†2], Thomas Tanay[†2], Matteo Maggioni[†2], Richard Shaw[†2], Sibi Catley-Chandar[†2,3], Yiren Zhou[†2], Jiankang Deng[†1], Ruijie Zhu, Jiahao Chang, Ziyang Song, Jiahuan Yu, Tianzhu Zhang, Khanh-Binh Nguyen, Joon-Sung Yang, Andreea Dogaru, Bernhard Egger, Heng Yu, Aarush Gupta, Joel Julin, László A. Jeni, Hyeseong Kim, Jungbin Cho, Dosik Hwang, Deukhee Lee, Doyeon Kim, Dongseong Seo, SeungJin Jeon, YoungDon Choi, Jun Seok Kang, Ahmet Cagatay Seker, Sang Chul Ahn, Aleš Leonardis[†4], and Stefanos Zafeiriou[‡†1]

[1]Imperial College London [2]Huawei Noah's Ark Lab [3]Queen Mary University of London [4]University of Birmingham

1. MPFER-H: Mutliplane Features Encoder-Renderer for Human Heads

Challenge organizing team-1. MPFER-H: MPFER for Heads

Abstract. Our top performing entry uses a Multiplane Features Encoder-Renderer model (MPFER) introduced in our CVPR 2023 paper: *Efficient View Synthesis and 3D-based Multi-Frame Denoising with Multiplane Feature Representations* [24]. MPFER generalizes the concept of multiplane images [23, 29, 7, 16] to feature space. While multiplane images encode the scene as a set of fronto-parallel RGB-α images that are then rendered through alpha-compositing, MPFER predicts feature maps that are concatenated and processed together by a learnt renderer. We describe the method and a number of incremental improvements made for this challenge below. More details can be found in [24].

1.1. MPFER-H: Mutliplane Features Encoder-Renderer for Human Heads

Overview. The general pipeline of our MPFER [24] model is illustrated in Figure 1. It consists of three main operations:

- *Forward warping.* For a given target view, a number of nearby views of size $3\,H\,W$ are warped into a plane sweep volume (PSV) using fixed homographies computed directly from the camera parameters of the nearby and target views. The size of the PSV is $D\,V\,3\,H\,W$ where the number of depth planes D and the number of views V are hyper-parameters.
- *Encoding.* The PSV is then processed by a shared Unet encoder, producing a set of multiplane features (MPF) of size $D\,C\,H\,W$ where the number of channels C is another hyper-parameter. More specifically, the encoding stage consists of D Unet passes with input sizes $(3V)\,H\,W$ (the views are concatenated along the channel dimension) and output sizes $C\,H\,W$.
- *Rendering.* Finally, the MPF is processed by a Unet renderer, producing the rendered view of size $3\,H\,W$. The input size is $(DC)\,H\,W$ (the depths are concatenated along the channel dimension).

Visualizations of the input PSV averaged over the input views and predicted MPF are shown in Figure 2.

Novelty. We made two main modifications to the original pipeline for this challenge.

- *No backward warping.* In [24], a single MPF centered on a reference view can be used to render multiple target views by backward warping the MPF before rendering it. Here, we drop the backward warping operation and compute the MPF directly centered on the target view. This modification simplifies the pipeline during training and allows to train larger models. However, one MPF must now be computed for each target view.
- *Group processing of the depth planes.* In [24], each depth plane of the PSV is processed by an Encoder Unet pass independently. Here, we propose to process the depth planes in groups, allowing the use of significantly more depth planes for a similar computational budget. More precisely for a group size G, we view the PSV tensor as a tensor of size $(D//G)\,(G{\times}3V)\,H\,W$ and process it with

[§]These authors contributed equally to this work.

[‡]S.Zafeiriou (s.zafeiriou@imperial.ac.uk) is the corresponding author.

[†]These authors are the "To NeRF or not to NeRF: VSCHH 2023" organizers, while the other authors are participants in the VSCHH challenge.

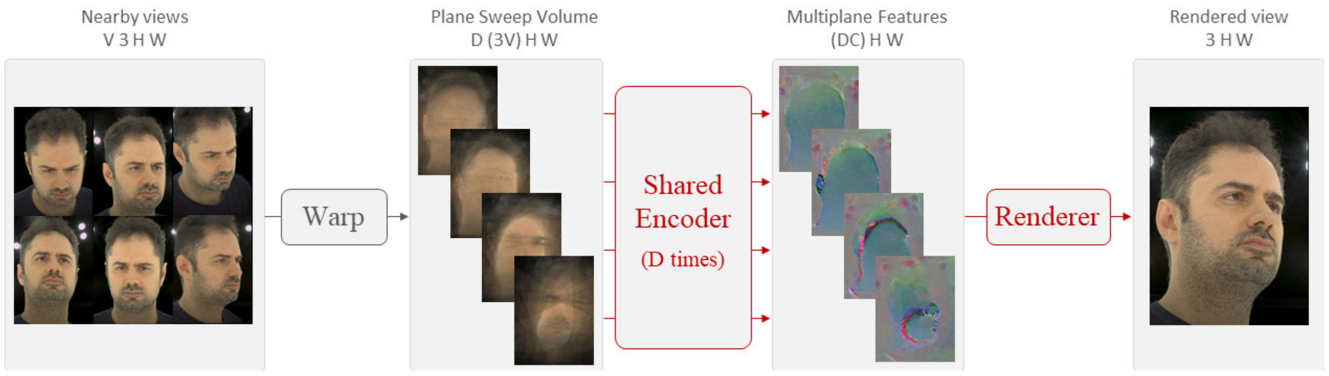


Figure 1: Illustration of our method. Learnable modules are highlighted in red and are implemented with convolutional UNets.

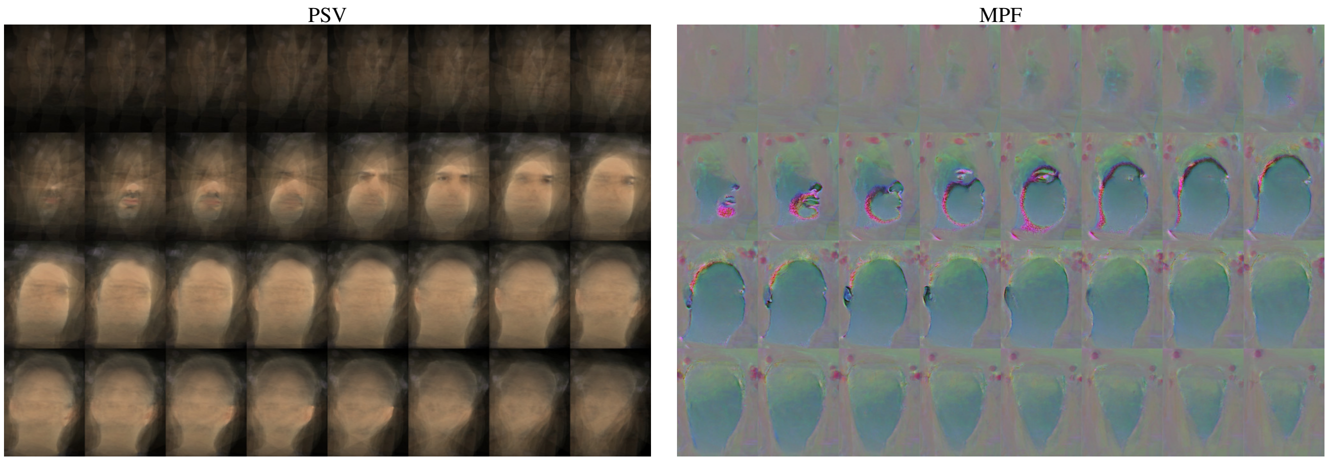


Figure 2: Visualizations of input plane sweep volumes (PSV) averaged over the input views and predicted multiplane features (MPF).

$(D//G)$ Unet passes.

Additional improvements. 4 more elements contributed to the score of our final submission.

- *Bigger model.* The reduced GPU memory usage during training resulting from the previous two modifications allowed us to train MPFER with larger Encoder and Renderer Unets, using 128 filters in their base convolutions, instead of 64 for the baseline.
- *Data augmentation.* To deal with the limited size of the training set, we introduced two simple and effective forms of data augmentation: random shuffling of the input views, and limited jittering of the depth planes. We also tried scaling up and down the input PSV by a small factor, or flipping it left-right, but these two strategies did not provide any additional benefits.
- *Positional encoding.* We help the Encoder and Decoder Unets know the region of the image they are currently processing by feeding them the height and width position of each pixel encoded as a pair of floats scaled in the [0,1] range concatenated to their inputs.
- *Finetuning.* For view synthesis, training MPFER with an L1 loss only produces poor results. The baseline model addresses this issue by using a mixture of VGG loss and L1 loss, but this strategy still tends to produce visible gridding artifacts in the output images. For our final submission, we first trained the model using the mixture of VGG and L1 loss, before finetuning it using an L1 loss only at a low learning rate.

1.2. Experimental Results

Environment. In all our experiments, we train one MPFER model for all the individuals in the ILSH dataset. We train using the Adam optimizer for 100k steps, with a learning rate of 1.5e-4 reduced to 1.5e-5 after 80k steps. The finetuned models are trained for another 4k steps with a learning rate of 1.5e-5, reduced to 1.5e-6 after 2k steps. We use

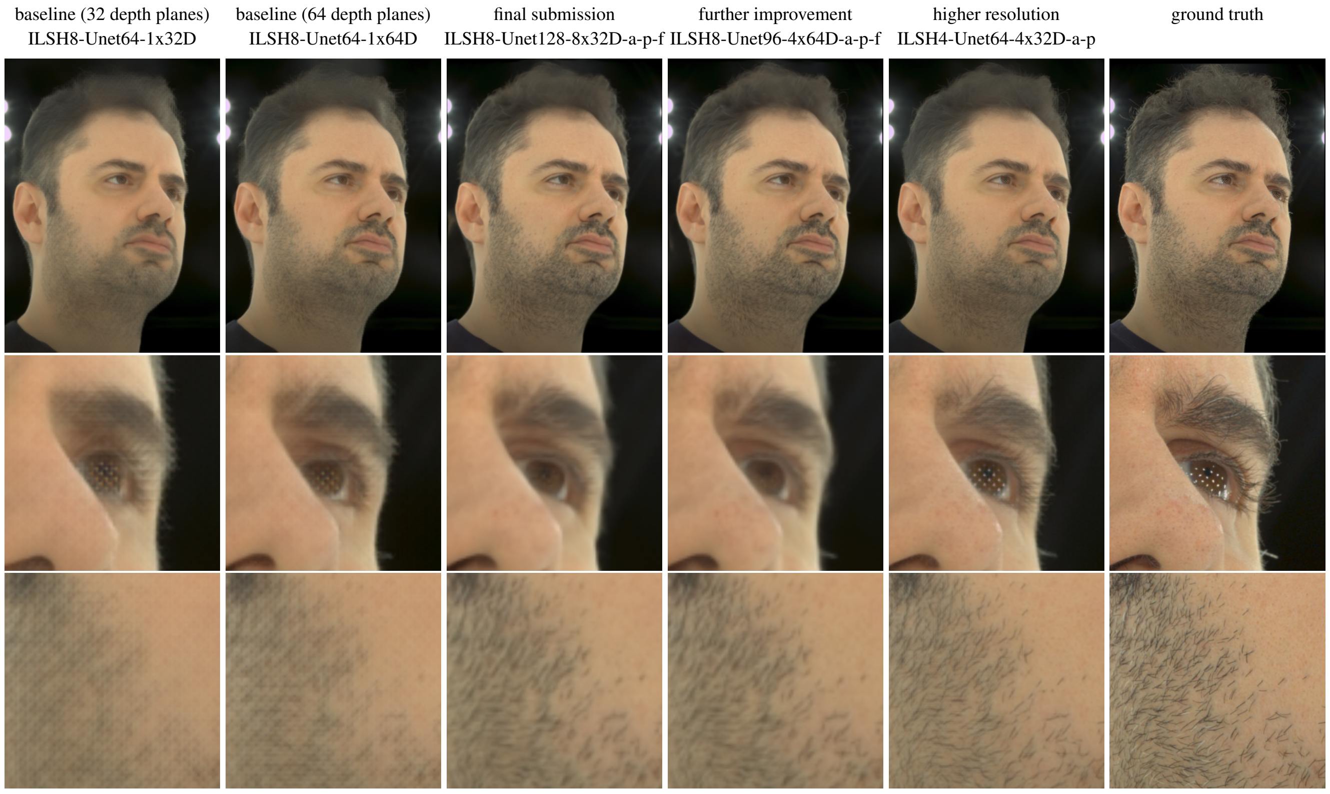


Figure 3: Qualitative results for a target view not seen during training.

4 nvidia V100 GPUs per training with a batch size of 4 (i.e. one instance per GPU). We found experimentally that training on large image regions is crucial for good performance; but it is also difficult as it tends to quickly saturate GPU memory during training. One simple solution is to train on downsampled images and we obtained our best results with images downsampled 8x (i.e. image size: 375×512), with training patches of size 364×364. The predicted images have the same resolution as the input images and are simply upsampled bilinearly to produce the final images. In most of our experiments, we used $V = 16$ nearby views in input as fewer views seemed to produce slightly less reliable 3D geometries. For the depth range, we visually inspected some PSVs and chose values that seemed compatible with all the scenes (see for instance the PSV in Figure 2). More precisely, we fixed the near depth at 1.02m and the far depth at 1.3m for all the individuals and all the views, both during training and inference, with a random jitter of 0.02m during training when using augmentation. We then uniformly sampled depths between these two values for a number of depth planes D varying between 32 and 256.

Quantitative Results. We present an ablation study with 7 different training setups in Table 1. "ILSH8" means that the input images are downsampled 8x. "Unet64" or "128" means that the Unets have a base number of filters of 64 or 128. "16V" means 16 input views. "1x32D" means 32D depth planes processed in groups of 1 (baseline). "8x32D" means 256 depth planes processed in groups of 8. "a" means data augmentation. "p" means positional encoding. "f" means finetuning. We see that all the elements discussed before contribute to the performance of the final submission, with an improvement over the baseline for the masked PSNR of 1.85dB. The reported time includes the forward warping of nearby views, the encoding of the MPF and the rendering of the final target view at the chosen resolution (375×512) on an Nvidia V100 GPU.

Evaluation Region	Full Region		Masked Region		Time
Metric	PSNR	SSIM	**PSNR**	SSIM	Sec.
ILSH8-Unet64-16V-1x32D	25.54	0.80	27.05	0.82	0.36
ILSH8-Unet64-16V-8x32D	26.05	0.81	27.64	0.82	0.75
ILSH8-Unet64-16V-8x32D-a	27.07	0.83	28.06	0.83	0.75
ILSH8-Unet128-16V-8x32D	25.47	0.80	27.32	0.82	1.5
ILSH8-Unet128-16V-8x32D-a	27.30	0.83	28.18	0.83	1.5
ILSH8-Unet128-16V-8x32D-a-p	27.61	0.83	28.54	0.83	1.5
ILSH8-Unet128-16V-8x32D-a-p-f	**28.05**	**0.84**	**28.90**	**0.83**	**1.5**

Table 1: Ablation study for our MPFER method on the ILSH dataset. Our final submission is highlighted in bold.

Qualitative Results. We compare the outputs of various models on a target view not seen during training in Figure 3. Our final submission is better than the baseline at reconstructing 3D geometries (see the eye region) and textures (see the beard region); and it does not suffer from gridding artifacts. Remarkably, both the baseline and our final submission can generalize the positions of the background lights across subjects. This is also apparent in the high PSNR and SSIM scores on the full regions compared to other approaches. We believe that this ability comes in part from the use of a learnt renderer.

1.3. Discussion

Further improvements. We were able to outperform our final submission by ≈0.2dB on the masked region simply by varying the hyperparameters (see Table 2).

Higher resolution. We also explored training on higher resolution images, by using ILSH images downsampled 4x. As discussed before, it is challenging to train on large image regions at this resolution and our final model performs slightly worse on the test set (see Table 2). Although the outputs look visually sharper (see Figure 3), this model is less accurate overall on the 3D geometries.

Evaluation Region	Full Region		Masked Region		Time
Metric	PSNR	SSIM	**PSNR**	SSIM	Sec.
ILSH8-Unet64-16V-1x64D	26.28	0.81	27.82	0.82	0.75
ILSH8-Unet96-16V-4x64D-a-p-f	**28.32**	**0.84**	**29.09**	**0.84**	**1.7**
ILSH4-Unet64-10V-4x32D-a-p	26.41	0.82	28.03	0.83	2.5
ILSH4-Unet64-10V-4x32D-a-p-f	26.70	0.83	28.61	0.84	2.5

Table 2: Additional results for our MPFER method on the ILSH dataset. A new best result is highlighted in bold.

Limitations. While our method performs remarkably well, one current limitation is the difficulty to train on large image regions at high resolution. It might be possible to address this limitation by optimizing memory usage further during training; or by using devices with larger memories; or by using gradient checkpointing. Another related limitation is that multiplane features representations (MPFs) are highly sparse, and a lot of the computation seems to be wasted away. In the MPF of Figure 2 for instance, the first and last 8 planes do not seem to contain much meaningful information at all, and the remaining 16 planes contain little information per plane. One direction we plan to explore in the future is to adapt the use of the computational resources to each scene in a more efficient way, for instance by adjusting the depth range on a per-view basis, or by computing high resolution features only in regions of interest.

2. DINER-SR: Depth-aware Image-based NEural Radiance fields with Super Resolution

Challenge organizing team-2. DINER-SR

Abstract. We present our submission to the ICCV 2023 Challenge "To NeRF or not to NeRF: A View Synthesis Challenge for Human Heads". We train a single generalizable DINER model using all subjects and training views in the ILSH dataset. We initialize the model with pre-trained FaceScape weights, enabling faster convergence and sharper results compared to training on ILSH data from scratch. We employ a super-resolution module to upscale the rendered images to full resolution with a 2D UNet model, trained with L1 and perceptual losses, further improving results. Our method improves upon our baseline by almost 6dB in the masked region and scored second on the leaderboard during the challenge phase.

Baselines: For the baseline method, we use the model DINER: Depth-aware Image-based NEural Radiance fields [21]. We employ their pre-trained model, trained on the FaceScape dataset [30].

As a pre-processing step, DINER requires initial depth maps for each training view, predicted using state-of-the-art depth estimation network TransMVSNet [5]. To achieve good-quality depth maps, we pre-process the data by resizing all images to be 256px width and predict foreground alpha masks for each image using three methods: matting estimation [15], face-parsing [28], and SMPL-fitting [1]. The final foreground mask is obtained by taking the union of the three predicted masks, as shown in Fig. 5. We use multiple methods because face-parsing fails for back-of-the-head views, while matting produces masks with holes and transparent regions. We compute the nearest four views for each training view based on the camera view directions and use these as input to the depth estimation network. Based on the SMPL fittings, we set near-far bounds to be [0.9, 1.4].

With depth maps estimated for each training view, we predict novel views for each test subject using the DINER model with pre-trained weights and the nearest four views as input, downsampled $4\times$ to 768×1024px resolution. Finally, we upsample the resulting image renders to the full resolution of 3000×4096px using bilinear interpolation.

2.1. DINER-SR: Depth-aware Image-based NEural Radiance fields with Super Resolution

To improve over the baseline method, we train a single DINER model for an additional 60,000 steps using all subjects and training views in the ILSH dataset. Training for longer could improve the results, as could fine-tuning the model for each subject individually, but we were limited by the challenge timeframe. We found that initializing the network weights with the pre-trained FaceScape model weights enabled faster convergence and sharper results than training the model on the ILSH dataset from scratch. However, the DINER pre-trained model was trained on FaceScape data with white backgrounds, while the ILSH dataset contains black backgrounds. Therefore, we adjust the model to render black backgrounds, set the initial estimate of background depth to be 1.4 with high confidence (zero standard deviation) and fine-tune the model sufficiently long to re-learn to render the backgrounds correctly black instead of white.

To improve results further, we employ a super-resolution module to upscale the predicted images to the full resolution of 3000×4096px. To do this, we upscale the image renders using bilinear interpolation and process them with a 2D UNet. We train a single UNet model for all subjects using all GT-render pairs in the training set, with a combination of L1 and perceptual (VGG) losses. We find that the UNet model helps reduce artifacts and corrects colour shifts in the NeRF renders. An overview of our method is shown in Fig. 4.

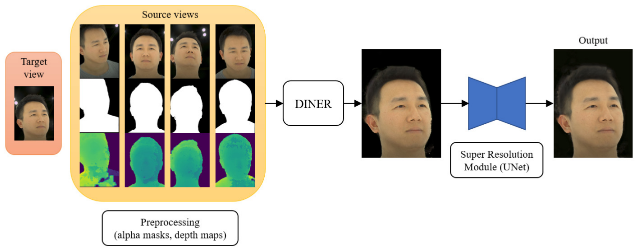

Target view
Source views
DINER
Output
Super Resolution Module (UNet)
Preprocessing (alpha masks, depth maps)

Figure 4: Our method overview. Four nearest source views to the target view are selected as input to the DINER model, along with corresponding predicted alpha masks and initial depth maps. The output renders of the DINER model are upscaled to full resolution using a 2D UNet super-resolution module which reduces rendering artifacts and colour shifts.

2.2. Experimental Results

Experimental environment: First, we train the DINER model on a single NVIDIA V100 32Gb GPU for 60,000 steps, taking approximately two days. We train with Adam optimizer [11], batch size 1 and a learning rate of 1e-4. Rendering time for a single image is roughly 86 seconds. Secondly, we render all training images and train the super-resolution module separately for 150,000 steps, with batch size 16, patch size 512×512px, taking about 12 hours. Inference for the super-resolution module takes 0.88 seconds.
Quantitative Results: The performance metrics for our method obtained for the challenge dataset are given in Table. 3. Our method improves over the baseline result in all metrics, with almost 6dB increase in PSNR for the masked region and 0.05 increase in SSIM. Ablation study results, given in Table 4, show that the super-resolution module provides an almost 1dB increase in masked PSNR.
Qualitative Results: Fig. 6 shows visualizations of our model output vs the baseline result. Close-ups show that our method displays better detail reconstruction and fewer artifacts than the baseline method. We also notice less colour shift in the results.

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metric	PSNR	SSIM	**PSNR**	SSIM	
Base Method [21]	14.81	0.58	22.72	0.78	86.37
Our Method	22.37	0.72	28.50	0.83	87.25

Table 3: Results of baseline and our method on the ILSH dataset.

Methods	Masked-PSNR
DINER [21] w/o masks	27.09
DINER w/ masks	27.61
Final Method (DINER w/ masks + SR)	28.50

Table 4: Ablation study compares using DINER w/ and w/o masks, and final our method using super-resolution (SR) module.

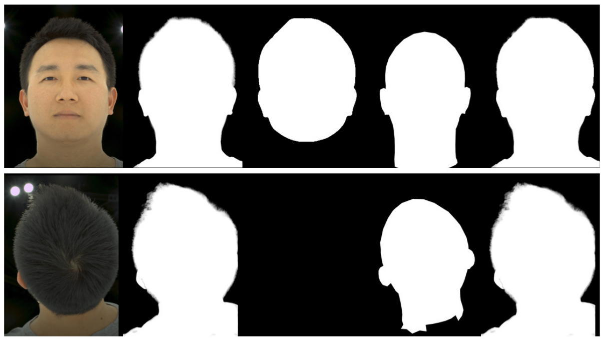


Figure 5: Mask generation, left to right: i) input image, ii) alpha mask [15], iii) face-parsing [28], iv) SMPL fit [1], v) union of masks.

2.3. Discussion

Limitation. Our method is limited by multiple processing steps, including foreground mask prediction, image resizing, depth estimation, DINER training and rendering, and finally super-resolution. Ideally, the model would be trained end-to-end; however, we are restricted by memory size and network capacity. Inference time is also relatively slow. Working with downsampled images may result in blurry outputs compared to using full resolution directly. We also only train on the masked regions, therefore we cannot reconstruct the backgrounds and the performance metrics on the full image are low compared to other methods.
Further Improvement. We could improve results beyond what was achieved during the challenge timeframe by

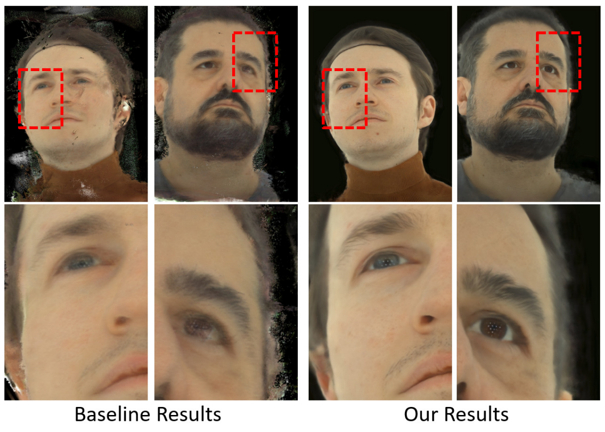


Figure 6: Qualitative results of our final method vs the baseline.

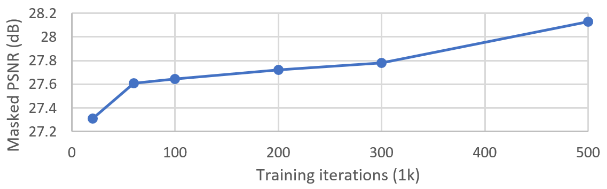


Figure 7: Test performance (Masked PSNR) vs training iterations.

training the DINER model for longer, as this leads to better reconstructions and higher PSNR values, as shown in Fig. 7. Additionally, training the super-resolution module for longer or using a different architecture such as an image transformer would likely improve results.

3. TIFace: Improving Tensorial Radiance Field and Implicit Surfaces for Face Reconstruction

Team-1. OpenSpaceAI

Abstract. This section outlines the methodology and approach used to tackle the challenge. First, the baseline model is introduced to fit the ILSH dataset. Then, we improve face reconstruction quality through tensorial radiance fields (T-Face) and implicit surfaces (I-Face) respectively. Finally, the experimental results demonstrate the effectiveness of the proposed method and superior performance on face reconstruction.

3.1. Methodology

Baseline: To address the View Synthesis Challenge for Human Heads (VSCHH), we use TensoRF [4] as the baseline. TensoRF utilize 4D tensors to model the radiance field of a scene. The key idea mainly focus on the low-rank factorization of 4D tensors to achieve better rendering quality and smaller model size with fast speed. Following the baseline model, we implement a vector-matrix decomposition version of TensoRF with some modifications to fit the ILSH dataset. As shown in Fig. 9, although the baseline model achieves relatively accurate facial reconstruction, the reconstructed results often have floating artifacts, which severely affects the reconstruction quality of the edges of the face. We speculate that this is because there are light sources in the background area of the rendered image, which causes a sudden change in the background color. To avoid this, we use an efficient way to get the masks corresponding to the input images. Furthermore, we improve both explicit and implicit rendering methods and build extra constraints based on the masks to improve rendering quality.

T-Face: Recently, SAM-based image segmentation methods have attracted a lot of attention [12]. Following ViT-Matte [27], we obtain the corresponding masks of input images through a small number of label points as prompts. As shown in Fig. 8, we get the masks fine enough to distinguish the foregrounds and the backgrounds of the images, even in the region of hair strands.

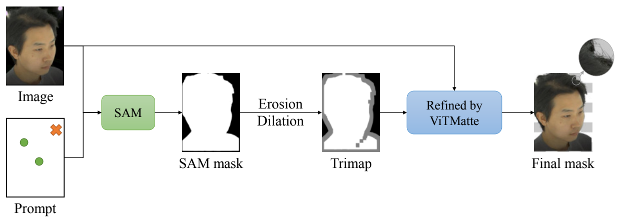


Figure 8: The pipeline of mask generation. To reduce the computational overhead, we first downsample the image and then upsample the predicted mask. Then we label the image with prompt points and use SAM to obtain a initial mask. After erosion and dilation on the image, we generate a trimap mask, where the gray areas represent the areas that need further segmentation. With the help of VitMatte, we obtain a final refined mask.

A natural way to utilize masks is to combine masks with RGB images into RGBA images, as implemented on the Blender dataset. This actually use masks to set the values of background area in the image to a fixed color (e.g. black). Although it appears to remove the effect of cluttered backgrounds on rendering faces, this method still exhibits some artifacts in our experiments, as shown in Fig. 10 and Fig. 11. We speculate that the reason is that the generated mask is not completely accurate, and the model lacks constraints on the background, resulting in not so fine sampling on the surface.

To address this issue, we propose a constraint on the mask and demonstrate its effectiveness through experiments. For each pixel, we march along a ray, sampling Q shading points along the ray and computing the accumulated density weights:

$$T = \sum_{q=1}^{Q} \tau_q (1 - \exp(-\sigma_q \Delta_q)), \tau_q = \exp(-\sum_{p=1}^{q-1} \sigma_q \Delta_q). \tag{1}$$

Here, σ_q is the density at sampled location x_q, Δ_q is the ray step size and τ_q denotes ray transmittance. The mask loss is:

$$\mathcal{L}_{mask} = \frac{\lambda}{|B|} \sum_x T_x^2 \mathbb{I}(x \in B), \tag{2}$$

where x is the pixel corresponding to the ray, B is the background areas according to the masks, $\mathbb{I}(\cdot)$ is the indicator function. Here, we set $\lambda = 0.01$ in our experiments. Note that we also tried to use cross entropy in mask loss function, and experiments showed that it is not as effective as the proposed loss.

I-Face: To further improve the face rendering quality, we explore another way to render photo-realistic faces. We observe that although T-Face shows promising results, it does not perform well in facial reconstruction details. Inspired by InstantNGP [18] and NeuS [25], we use implicit surface rendering for face reconstruction based on Instant-NeuS implementation [9]. Before this, we also tried Instant-NeRF [9] (combination of InstantNGP [18] and NeRF [17]) to complete face reconstruction, but gave up due to the difficulty of imposing constraints on neural radiance fields to remove artifacts. In the experiment, we use the binary cross entropy loss as the mask loss, as mentioned in NeuS [25]. To avoid the floating artifacts, we also add the sparsity loss:

$$\mathcal{L}_{sparsity} = \frac{1}{|S|} \sum_{y \in S} \exp(-\gamma d(y)), \tag{3}$$

where y is the sampled point, $d(y)$ is the corresponding SDF values, S represents the collection of sampling points. Here we set $\gamma = 0.5$. Finally, the total loss is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{color} + \alpha \mathcal{L}_{reg} + \beta \mathcal{L}_{mask} + \gamma \mathcal{L}_{sparsity}, \tag{4}$$

where $\mathcal{L}_{color}$ is a MSE loss on RGB color and $\mathcal{L}_{reg}$ is the Eikonal regularization [8].

TI-Face: To further improve the results, we use a simple and effective linear weighted ensemble method to obtain the final results. In practice, we use a set of linear weights $(0.1, 0.6, 0.3)$ corresponding to baseline, T-Face, I-Face to ensemble the results.

3.2. Experimental Results

This section gives implementation details and presents the results achieved during the challenge phase. We first compare our approach with baseline method and then analyse the effectiveness of the proposed improvements.

Implementation Details. We implement our T-Face and I-Face on PyTorch. In T-Face, we follow the baseline configurations, training our model for 50000 steps with a batch size of 4096 rays on a single NVIDIA RTX 3090 (40-60 minutes per scene). In I-Face, we train our model for 20000 steps with a dynamic batch size (256-8192) on a single NVIDIA RTX 3090 (10-20 minutes per scene).

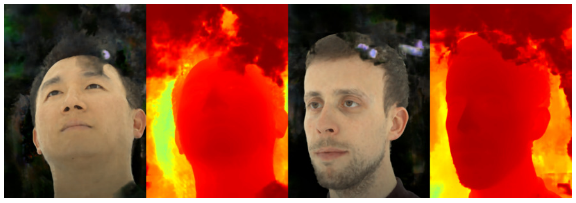


Figure 9: The rendering results of baseline model. Left: rendered image. Right: depth image.

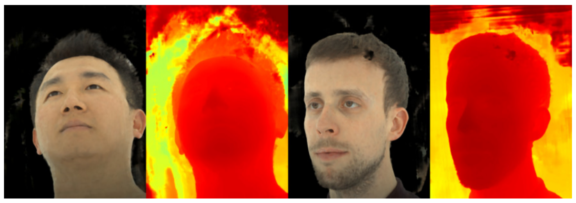


Figure 10: The rendering results of baseline model based on masks. Left: rendered image. Right: depth image.

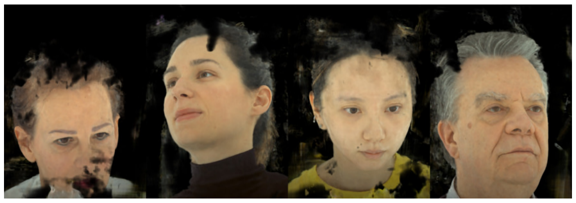


Figure 11: The failure cases of baseline model based on masks.

Comparison with the baseline. We evaluate our TI-Face on the ILSH dataset. As shown in Table. 5, the results obtained by fusing baseline, T-Face and I-Face perform better than those rendered by either method alone. We also selected several examples to qualitatively compare these methods, as shown in Fig. 12 and Fig. 13. Interestingly, although I-Face appears to render clearer images, it does not perform as well as T-Face in actual evaluations. We speculate that the reason is that the results generated by I-Face

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metric	PSNR	SSIM	**PSNR**	SSIM	
TensoRF [4]	20.28	**0.70**	24.70	0.81	31.13
T-Face(Ours)	21.42	0.67	26.63	0.82	**30.88**
I-Face(Ours)	20.99	0.66	25.84	0.81	45.73
TI-Face(Ours)	**21.66**	0.68	**27.02**	**0.83**	76.88

Table 5: Results of baselines and our methods on the ILSH dataset in the challenge phase. TI-Face refers to the final fused results.

Evaluation Region	Masked Region	
Metric	**PSNR**	SSIM
baseline	24.03	0.83
baseline+SAM-mask	25.24	0.83
baseline+mask	25.39	0.84
baseline+mask+$\mathcal{L}_{mask}$(cross entropy)	25.52	0.84
T-Face(baseline+mask+$\mathcal{L}_{mask}$(L2))	**26.77**	**0.84**

Table 6: Abaltion study of T-Face on a subset (the first three) of ILSH dataset in the developing phase.

have an overall deviation from the ground truth, as shown in the Fig. 14, the rendering results of I-Face have unexpected color differences and geometric inconsistency. We have not yet identified the cause of this issue, and it is possible that we will make further improvements in the future.

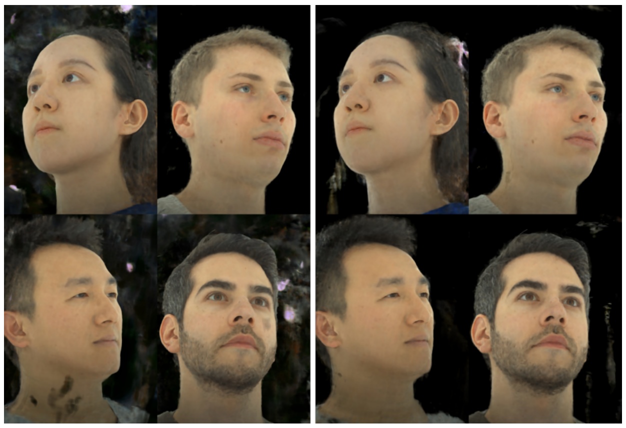


Figure 12: Examples of qualitative results from baseline and T-Face. Left: baseline. Right: T-Face.

Ablation Study. To demonstrate the effectiveness of the proposed improvement, we ablate our methods on the first three scenes of ILSH dataset. As shown in Table. 6, the ablation experiment demonstrated the effectiveness of the proposed mask loss function. Please note that in our experiment where we only add a mask, we set the background area to black in order to eliminate artifacts, but the experiment shows that without appropriate constraints on the mask, floating artifacts still exists. The use of qualitative results to demonstrate the ablation experiment of T-Face is clearly more intuitive. As shown in Fig. 15, the proposed sparsity loss function $\mathcal{L}_{sparsity}$ significantly improves the quality of implicit surface reconstruction. Nevertheless, the reconstruction quality of I-Face is still inferior to T-Face. Therefore, improving the rendering quality of implicit surfaces remains a huge challenge.

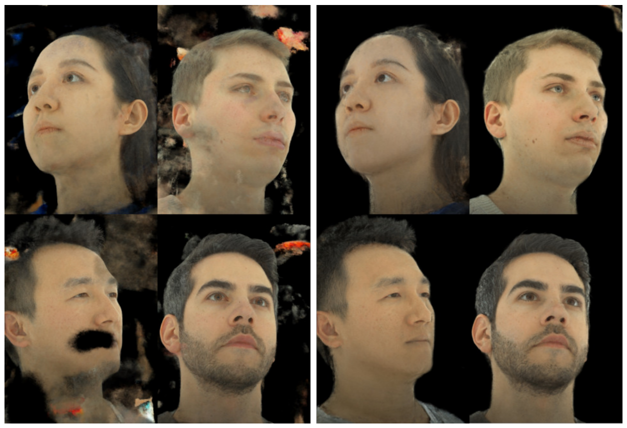


Figure 13: Examples of qualitative results from Instant-NeRF and I-Face. Left: Instant-NeRF. Right: I-Face.

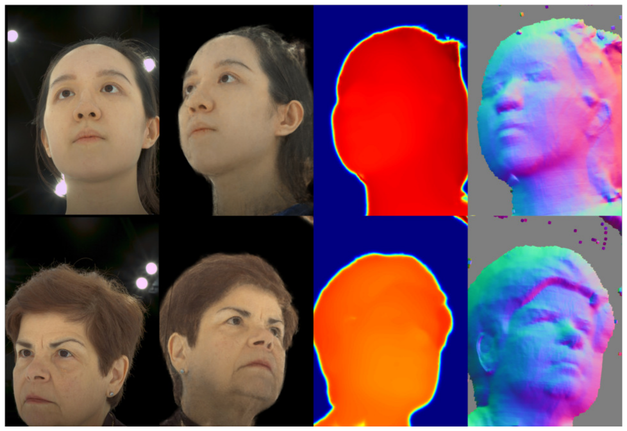


Figure 14: Examples of qualitative results from I-Face. From left to right: GT image (adjacent frames), rendered image, the predicted depth image, the predicted geometry normal.

4. Recovering Better Texture on Faces by better sampling strategy.

Team-2. NoNeRF

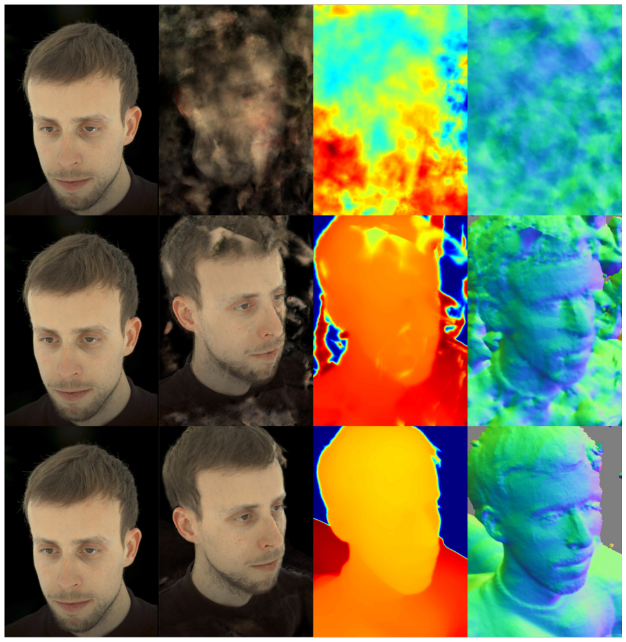


Figure 15: Ablation study of I-Face. From left to right: GT image (adjacent frames), rendered image, the predicted depth image, the predicted geometry normal. From top to bottom: Instant-NeuS, Instant-NeuS+mask, I-Face(Instant-Neus+mask+$\mathcal{L}_{sparsity}$).

Abstract. Neural Radiance Fields (NeRFs) are a powerful representation for modeling a 3D scene as a continuous function. Though NeRF is able to render complex 3D scenes with view-dependent effects, few efforts have been devoted to exploring its limits in a high-resolution setting. Specifically, existing NeRF-based methods face several limitations when reconstructing high-resolution real scenes, including a very large number of parameters, misaligned input data, and overly smooth details. Especially for 360-degree facial scenes, it is difficult to render due to the asymmetry and textile features. In this work, we conduct our analysis on the VSCHH challenge dataset. Through observation, we notice that conventional methods such as TensoRF fail to render high-quality facial scenes due to the light source in the background. Furthermore, to increase the inference speed and reduce the training time, a strategy of skipping rays is used, thus reducing the quality of the outcomes. To overcome those drawbacks, we conducted a simple method to enhance the output quality dubbed as NoNeRF (No skipping rays NeRF).

4.1. Methodology

This section presents an overview of our method for TO NERF OR NOT TO NERF: VSCHH at ICCV 2023 [†]. Based on the detailed analysis of the dataset, a better ray marching sampling strategy produces better texture outputs than the conventional method. Additional studies further demonstrate the effectiveness of the proposed pipeline, and we will show more details of our experiment.

Baselines: In this work, we utilize TensoRF [4]) and the NerfAcc [13] module. We implement the feature decoding function S as either an MLP or SH function and use $P = 27$ features for both. For SH, this corresponds to 3rd-order SH coefficients with RGB channels. For neural features, we use a small MLP with two FC layers (with 128-channel hidden layers) and ReLU activation. We use the Adam [11] optimizer with initial learning rates of 0.02 for tensor factors and (when using neural features) 0.001 for the MLP decoder. We optimize our model for T steps with a batch size of 4096 pixel rays on a single NVIDIA RTX 3090 GPU (24GB). We apply a feature grid with a total number of N_0^3 voxels; the actual resolution of each dimension is computed based on the shape of the bounding box. To achieve coarse-to-fine reconstruction, we start from an initial low-resolution grid with N_0^3 voxels with $N_0 = 128$; we then upsample the vectors and matrices linearly and bilinearly at steps 2000, 3000, 4000, 5500, 7000 with the numbers of voxels interpolated between N_0^3 and N^3 linearly in logarithmic space. The maximum voxel resolution is set to 300. The models are trained for 50000 iterations with 96 components.

Recovering Better Texture on Faces by better sampling strategy: For better quality results, we adopted NerfAcc module. Specifically, we include the Occupancy Grid Estimator for a faster and better sampling strategy. We set the resolution for the estimator to 256. Since we aim for better results, we do not need the compression effects from TensoRF, therefore, we set the thresholds to 0. In addition, we only estimate the PSNR loss for the valid masked RGBs. For better quality, the maximum voxel resolution is set to 1024. The models are trained for 50000 iterations with 192 components. We increase the number of upsamples at steps 2000, 3000, 4000, 5500, 7000, 8500, 10000, 12000, 14000, 16000, 18000, 20000, 22000, 24000, 26000, 28000, 30000.

4.2. Experimental Results

Experimental environment: We evaluated both the baseline and our method on a single NVIDIA RTX 3090 GPU. A Pytorch library version 1.13.1 is used, with CUDA version 12.2.

This section includes a detailed explanation of the qualitative and quantitative results between the baseline model

[†]https://sites.google.com/view/vschh/home

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metric	PSNR	SSIM	**PSNR**	SSIM	
BaseMethod [4]	20.13	0.70	24.37	0.81	72.55
NoNeRF	20.37	0.69	**26.43**	**0.82**	175.58

Table 7: Results of baselines and NoNeRF on the ILSH dataset.

Methods	Masked-PSNR
Baseline	24.37
Increase the # of components	25.11
Increase the voxel resolution	26.17
Use NerfAcc sampling strategy	26.43
Final Method (Ensemble)	26.43

Table 8: The results from each individual improvement and the final result which ensembles all of them.

and our method:

Quantitative Results: For the quantitative results, we reported two metrics: PSNR and SSIM for both the full region and masked region. As we can see, our proposed method has a slight improvement over the baseline method on the full region performance. This could be due to our training strategy that only optimizes the RGBs for valid masked RGBs. However, for the masked region performance, our method provides a higher SSIM and higher PSNR. NoNeRF achieves a 2.06 (dB) improvement on PSNR over the baseline. Which shows the effect of a better sampling strategy on the output quality. It should be noted that since we increase the voxel resolution, adopted more components, and do not skip any rays, therefore the runtime is much higher than the baseline.

Qualitative Results: We visualized the model output, as shown in Fig. 16. Compared with the baseline model TensoRF alone, the outputs are blurry and have mixed RGBs between different regions, resulting in low-quality outputs despite the similar full region performance. Especially, for the conventional method and sample strategy, most RGBs are skipped to increase the speed and save resources. However, this causes a reduction in the quality of the outputs, as we could see in Fig. 16 with many black RGBs on the face region.

4.3. Ablation study

In Fig. 17, 18, 19, we visualize more examples from the challenge phase.

5. Sphere-Guided Neural Implicit Face Reconstruction

Team-3. CogCoVi

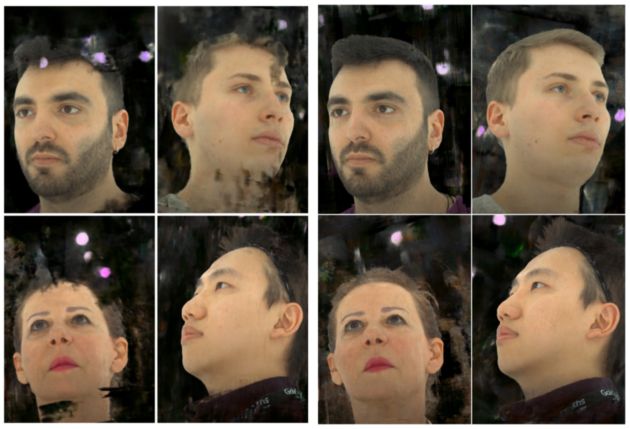


Figure 16: Some examples visualizations from challenge phase outputs between the baseline TensoRF (left) and the NoNeRF (right).

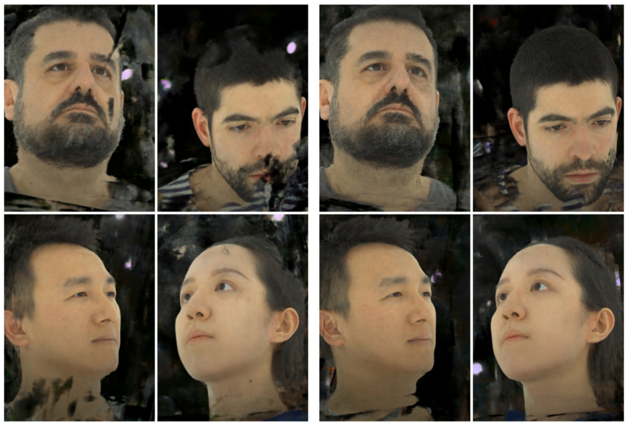


Figure 17: Some examples visualizations from challenge phase outputs between the baseline TensoRF (left) and the NoNeRF (right).

Abstract. Most lines of work in recent years employ methods based on Neural Radiance Fields [17] for the task of Novel View Synthesis. These methods show impressive capabilities of viewpoint generalization within the input cameras' positions manifold. However, in sparse camera setups, their performance decreases considerably. To this end, several methods [10, 20] introduced additional regularization terms to prevent overfitting to the input. We follow a similar idea and build our solution around the NeuS model [25], which minimizes an Eikonal penalty [8] in addition to the rendering loss. A side benefit of this regularization is the reduction of floating artifacts, making the model more suitable for the sparse camera setup. Still, the NeuS model samples the points for volume rendering along the entire camera ray, which has a slow convergence and is prone to

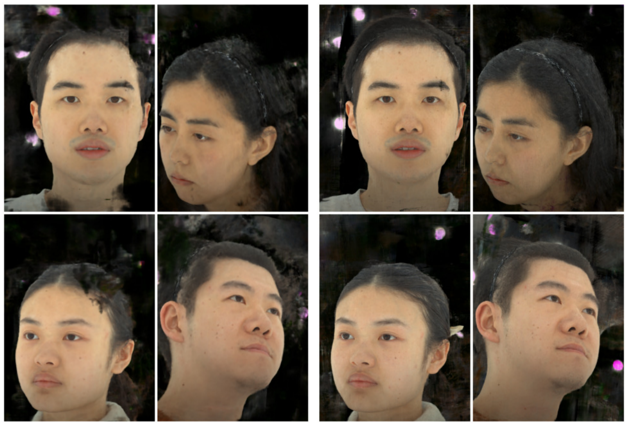


Figure 18: Some examples visualizations from challenge phase outputs between the baseline TensoRF (left) and the NoNeRF (right).

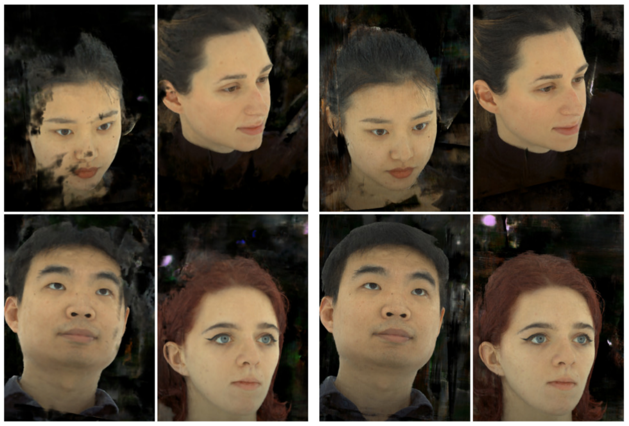


Figure 19: Some examples visualizations from challenge phase outputs between the baseline TensoRF (left) and the NoNeRF (right).

degenerate solutions. Therefore, our solution is based on sphere-guided trained NeuS [6] that uses optimizable primitives to bound the sampling space. Moreover, we propose an improved primitive initialization scheme for the task of face reconstruction.

5.1. Methodology

Baseline: NeuS [25] is one of the first methods that introduce additional surface constraints to NeRF [17] by representing the learned geometry as a signed distance function. Although NeuS underperforms compared to NeRF for the task of NVS under a dense-camera setup, we choose to exploit this model because of the Eikonal penalty. This regularization encourages the gradients of the geometry network to be of unit 2-norm and acts as a smoothness prior, preventing the appearance of noisy surfaces in the space. We produce our baseline results using the NeuS model[†] trained without segmentation masks or a background network and adjust the following hyperparameters: $batch\_size = 1024$, $igr\_weight = 0.05$, $sdf\_network.multires = 8$. We stop the training after 100k iterations to prevent overfitting the training views.

Sphere-Guided NeuS: One of the main issues in sparse camera setups is the hallucination of floating artifacts that satisfy the input views but do not generalize to new viewpoints. The problem is amplified in larger scene spaces, as they are more challenging to reconstruct accurately. For that reason, we follow the method introduced in [6] to gradually prune the empty space of the scene and guide the sampling to the area of interest using spherical primitives.

As this approach builds around NeuS, we follow the same training setting as for the baseline. Differently from the original implementation of sphere-guided training,[†] which randomly samples the origins of the spheres throughout the scene volume, we propose to improve the initialization of the primitives using a fitted FLAME mesh for each subject [14]. This enables us to reduce the starting radius of the primitives, further concentrating the training samples around the target surface. We fit the FLAME model minimizing the landmark distance between the model and unprojected 2D facial landmarks detected in the frontal input images. To reflect the enhanced initialization, we adjust the following hyperparameters for spheres' optimization: $n\_spheres = 12000$, $radius\_scheduler.max = 0.15$, $i\_warmup\_spheres = 10000$. Fig. 20 provides an overview of the proposed method.

5.2. Experimental Results

We report the numerical results for the baseline and our final model in Table 9. Additionally, we ablate the proposed FLAME-based initialization and evaluate the model with randomly-initialized primitives. By using sphere-guided training, we observe a significant improvement of 7.07 % in the PSNR evaluated inside the face region. Our enhanced initialization further boosts the performance. In Fig. 21, it can be observed how the sphere guidance amends the artifacts around the boundaries of the face, and that superior results are obtained with our final model. The metrics on 'Full Region' are poor for all three models, as we do not explicitly address the background region. The visual results highlighted in Fig. 22 confirm that our approach does not overfit the training views but succeeds in reconstructing the subject, as novel views do not display severe artifacts.

All the considered models are trained from scratch for every scene for 100k iterations. The fitting time per scene is around 5h45min for the baseline and 6h30min for the final

[†]github.com/Totoro97/NeuS
[†]github.com/AndreeaDogaru/SphereGuided

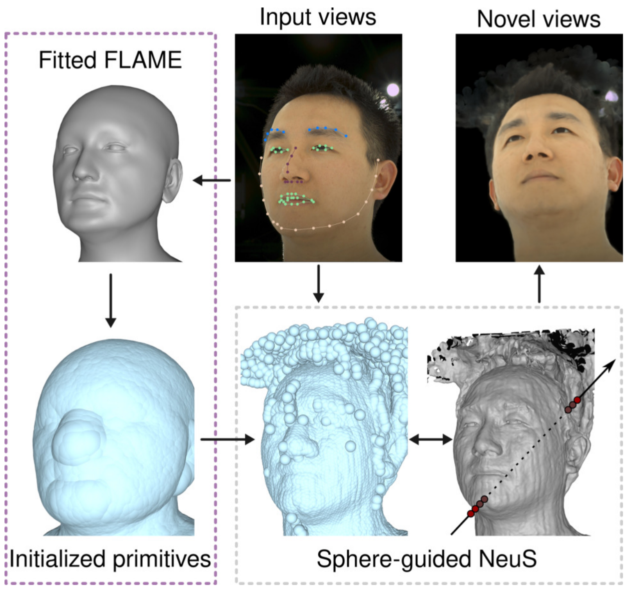


Figure 20: Method overview. Firstly, we detect 2D facial landmarks in the input frontal views and use them to fit the FLAME model. Next, we sample the primitives' origins from the model's surface. The initialized sphere-based surface is then optimized jointly with the NeuS model. Finally, we volume render novel views of the trained neural implicit surface.

Evaluation Region	Full Region		Masked Region	
Metric	PSNR	SSIM	**PSNR**	SSIM
NeuS [25]	21.02	**0.72**	24.33	0.80
Sphere-Guided NeuS [6] w/ rand. init.	21.46	0.70	26.05	**0.82**
Sphere-Guided NeuS w/ FLAME init.	**21.49**	0.70	**26.33**	**0.82**

Table 9: Quantitative comparison on the ILSH dataset.

model. Table 10 reports the averaged rendering times. We consider the default setting with 128 points sampled per ray and a faster alternative with 64, which does not affect the rendering quality of the sphere-guided models.

5.3. Discussion

Though the proposed approach obtains good results in the face region, there are still challenges to be considered. The lack of background modeling induces artifacts that cannot be fully handled through sphere-guided training. Because of the sparse view setting, using a dedicated background network as proposed in [25] does not solve the issue, and other artifacts appear, as illustrated in Fig. 23.

Additional limitations to be addressed in future works include poor reconstruction of the shoulders and hair areas.

Methods	Points per ray	Masked PSNR	Time (sec.)
Baseline	128	24.33	944
Ablation	128	26.05	862
Ours	128	**26.33**	806
Baseline	64	24.14	510
Ablation	64	26.04	470
Ours	64	**26.32**	456

Table 10: Rendering time per frame averaged across scenes.

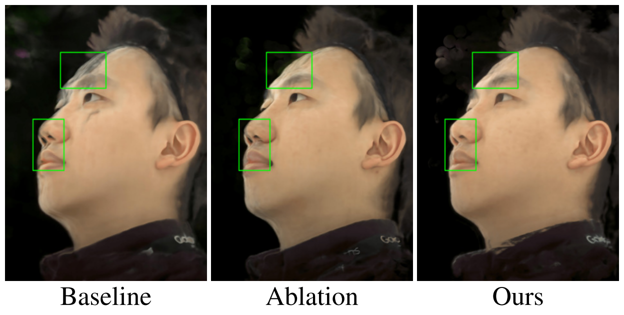


Figure 21: Illustrative qualitative results on the ILSH dataset.

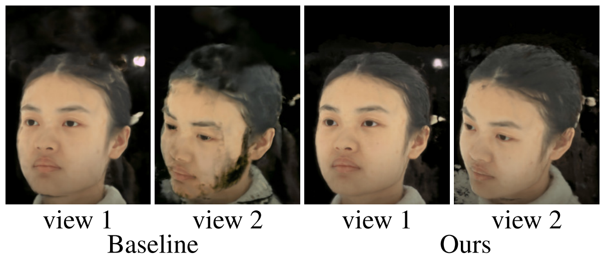


Figure 22: Multi-view rendering comparison. In the first row, it is noticeable that NeuS yields a decent rendering for 'view 1' but fails for 'view 2', displaying unrealistic artifacts. On the other hand, the renderings resulting from our method are consistent and more faithful to the input images.

5.4. Acknowledgment

This project is funded via the German Federal Ministry of Education and Research (BMBF) via the project "Scaling Inverse Rendering to the Real World".

6. Artifact Avoidance by Reducing Problem Complexity

Team-4. CUBE

Novel view synthesis is a complex problem that has been the subject of much research. Many approaches that seek to solve such a problem assume a large supply of

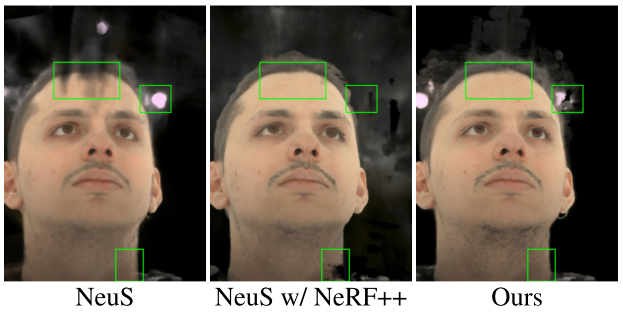


Figure 23: Background modeling is challenging in sparse setups.

standard-resolution data capturing a wide range of viewing angles. The ILSH dataset, and accompanying challenge, increase the problem's complexity with only 22 training images per subject at a high-resolution of 3096×000. With these deviations from the typical data structure, to the best of our knowledge, all preexisting methods are either unable to effectively capture the scenes geometry, have substantially reduced visual clarity, or suffer from significant rendering artifacts. In the few methods capable of capturing the scene's geometry, we found that many visual artifacts are a result of improper parameter settings and background interference. As a result, we propose a method to avoid such problems through a more intelligent selection of parameters and the complete removal of the oftentimes complex background structures.

6.1. Methodology

Baseline: Mip-NeRF 360 [2] acted as our baseline method for this challenge. To reproduce our baseline, we use the standard blender parameters from [2] with 512 hidden units, and change the near/far values to effectively capture the scene's geometry. According to the challenge forum, we change the near and far values to 0.4 and 2.8 respectively. We also use the provided masks from the ILSH dataset to remove the camera calibration borders around the images.
Artifact Avoidance by Reducing Problem Complexity: Our method directly extends the baseline to better perform on the challenge of novel view synthesis for human heads using a set of sparse, high-resolution images. In specific, we achieve better reconstruction over the baseline by confining the problem to maximize important metrics (the head) and by fine-tuning certain parameters to promote better geometry and visual clarity.

During the development of our method, we noticed that many reconstruction artifacts are caused by not the subject's face, but by the background. Given the sparse set of images, the model is simply unable to effectively capture everything (face + background). Due to this, we confine the problem

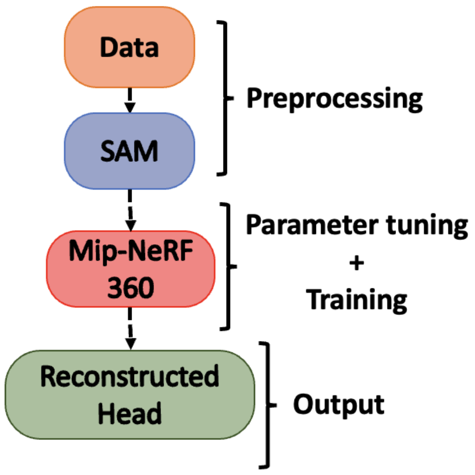
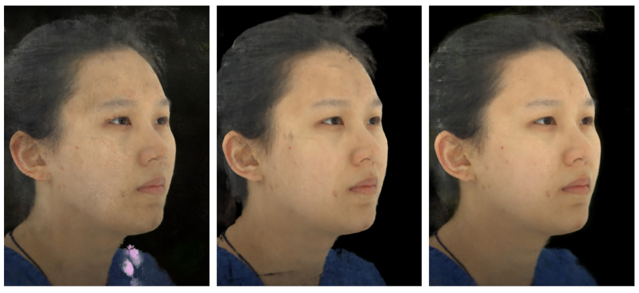


Figure 24: An overview of our method.

with SAM [12], by applying a dilated binary mask to set all background pixels in our training data to black. Dilation of the mask is necessary to ensure there are no masking errors (e.g. missing parts of the face). Removing background structure and lighting significantly minimizes, or in most cases, removes, artifacts caused by the background while maximizing the performance of head reconstruction. For a more visual understanding of SAM's benefits and the need for dilation, refer to Fig. 25



(a) Baseline (b) Ours-1 (c) Ours-2

Figure 25: Effects of masking the background with (Ours-1) and without dilation (Ours-2) compared to the baseline. We see significant improvement from the baseline to Ours-1 with the removal of the lighting artifact on the shoulder and better color/texture for the face. Ours-2 further improves over Ours-1 by removing the black line on the neck caused by masking errors.

In addition to confining the problem, our method also uses more intelligently selected parameters to further decrease the reconstruction artifacts found in the baseline. The primary parameter we found to make the most impact on

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metric	PSNR	SSIM	**PSNR**	SSIM	
Mip-NeRF 360 [2]	20.59	0.71	24.13	0.80	78.00
Ours	21.07	0.66	25.72	0.81	95.00

Table 11: Quantitative comparison between Mip-NeRF 360 and our method on the ILSH dataset.

our results was the number of samples taken from each ray. There was inconsistency across subjects. At times, 32 samples resulted in empty, discolored, or incorrectly textured spots on the face, whereas 64 samples seemed to resolve it as shown in Fig. 26. Conversely, in other cases, the opposite was true and 32 samples were ideal. The full overview of our method can be seen in Fig. 24.

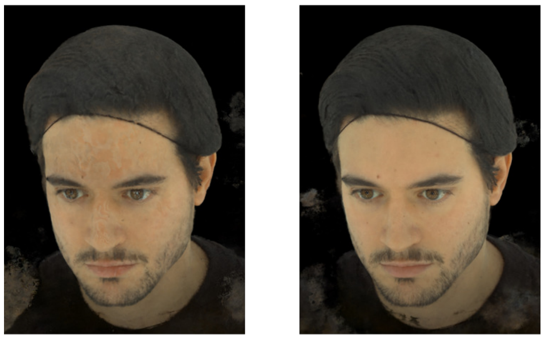

(a) Ours-1 (b) Ours-2

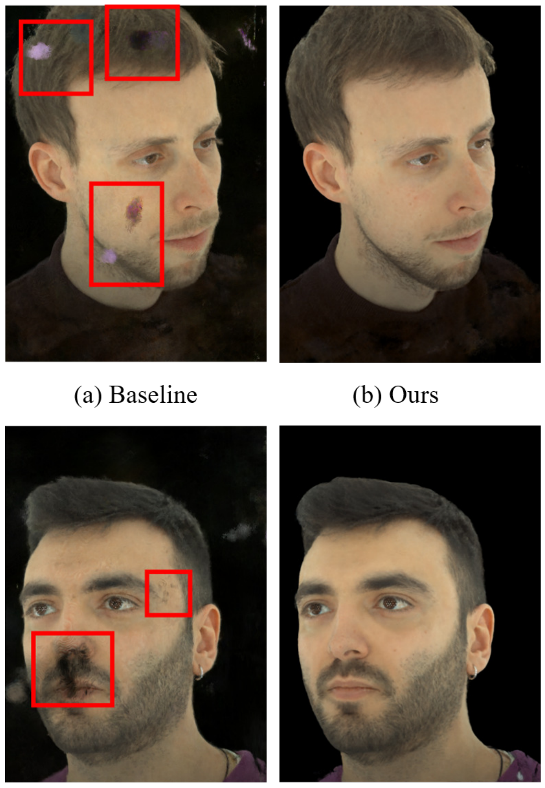
Figure 26: The number of points sampled along each ray is critical to accurate renderings. Ours with 32 samples (Ours-1) struggles with representing the texture and color on the subject's forehead and cheeks, where 64 samples (Ours-2) resolves this problem.

6.2. Experimental Results

This section presents the results achieved during the challenge phase. All experiments were performed on single A100 and H100 GPUs.

Quantitative Results: Quantitatively, our method has noticeable gains in performance over the baseline, as shown in Table. 11. With the most important metric being for the masked region, our method has a PSNR increase of 1.59 dB over Mip-NeRF 360 [2]. Even though the performance on the full region is not nearly as important, we still see a slight increase in PSNR.

Qualitative Results: As demonstrated by Fig. 27, our method is substantially better at reconstructing the human head. In particular, by setting the background to black, and reducing overall complexity, we see much better performance on the face. Furthermore, the benefits of fine-tuning the sampling parameter are also shown.


(a) Baseline (b) Ours

(c) Baseline (d) Ours

Figure 27: Our method produces results with far less artifacts that are caused by the background and improper sampling.

6.3. Discussion

While our proposed method improves both quantitatively and qualitatively over the baseline method, there is still certainly more development to be done. At the moment, selecting the best number of points to sample along each ray is not adaptive, requiring us to examine the renderings and adjust the setting accordingly. This trial and error process of determining how many points to sample (32 or 64) can increase the already lengthy processing time for our method. Assuming the point sampling is correct, the total training time per subject is about 5 hours. However, if it is not, the process of finding the correct one can extend the total training time to approximately 6 hours. In addition to adaptive sampling and training time, our method is also limited in its ability to represent all high-resolution details. These are all potential avenues for future improvements.

We hope that our method will help avoid future artifacts and lead to more accurately reconstructed human heads.

7. Mitigating Floating Artifacts through Imposing Occlusion Regularization and Backview Exclusion

Team-5. Y-KIST-NeRF: Yonsei-KIST NeRF

Abstract. The ILSH dataset consists of 52 subjects with 24 views each which can be thought of as a sparse view problem for a novel view synthesis task. Sparse views lead to floating artifacts, degrading the quality of novel views. To overcome this problem, we exclude certain views during training, and adopt an occlusion regularization term allowing us to render high quality novel views.

7.1. Methodology

Baseline: To reproduce our baseline method TensoRF [4], we introduce the parameter values used to produce the baseline results based on the ILSH dataset. Firstly, create a configs file for the ILSH dataset with the following parameters: dataset name ”llff,” ndc ray 0, n iters 50000, n lamb sigma [16,4,4], n lamb sh [48,12,12], shadingMode MLP Fea, fea2denseAct relu, view pe 0, fea pe 0, TV weight density 1.0, TV weight app 1.0. Secondly, set the scene bounds near far to [3.5,7.0], and the object bounds near far to [0.4,2.8]. Thirdly, as we set ndc ray 0, comment out some relevant lines in the LLFF loader. Lastly, disable pose centering in the LLFF loader.

Removing floating artifacts: To surpass the baseline performance, we introduce occlusion regularization to reduce the floating artifacts commonly encountered in few-shot neural rendering tasks. The key idea of this regularization term is to penalize the density fields near the camera. Additionally, we find it beneficial to exclude all the backside views in the preprocessing step to avoid floating artifacts.

We conjecture that the reason why the baseline method doesn’t work well for the ILSH dataset is the lack of training views, which leads to the presence of ”walls” or ”floaters” near the camera. Regions with minimal overlap in the training views can result in geometric inconsistencies, causing unexpected dense volumetric floaters in close proximity to the camera. To address this, we introduce an effective occlusion regularization technique that penalizes density values near the camera. This regularization significantly improves the baseline method. Our implementation extensively leverages the FreeNeRF [26] codebase to incorporate the occlusion regularization.

The approach involves defining a binary mask vector, denoted as m_k, which determines whether a point should be penalized or not. Additionally, we sample K points along the ray in order of proximity to the origin (from near to far), and σ_K represents the density values of these points. To minimize the occurrence of floaters near the camera, we set the values of m_k up to a specific index M (referred to as the regularization range) to 1, while the rest are set to 0. By doing so, we effectively control the penalization for the points in close proximity to the camera.

$$\mathcal{L}_{occ} = \frac{[\sigma_K]^T \cdot m_K}{K} = \frac{1}{K}\sum_K \sigma_k \cdot m_k \tag{5}$$

The final loss function consists of four terms, L_2 loss between pixel values $\mathcal{L}_p$, occlusion regularization $\mathcal{L}_{occ}$, and $\mathcal{L}_{TV_\sigma}$, $\mathcal{L}_{TV_c}$, which denotes TV (total variation) loss of density and appearance on our vector and matrix factors, respectively [4].

$$\mathcal{L} = \mathcal{L}_p + \lambda_{occ}\mathcal{L}_{occ} + \mathcal{L}_{TV_\sigma} + \mathcal{L}_{TV_c} \tag{6}$$

We empirically find that using 0.1 for λ_{occ} and 20 for M works best with our model.

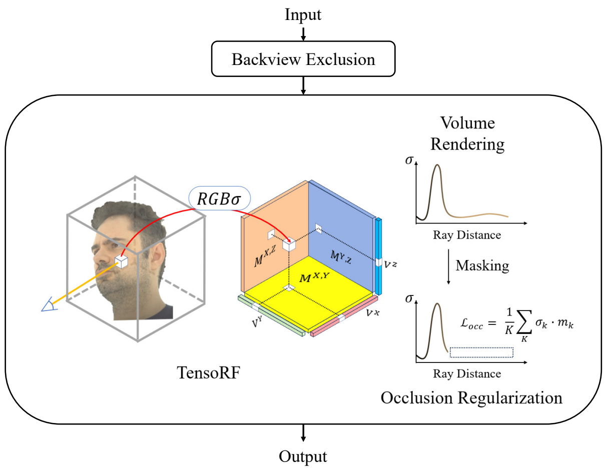


Figure 28: Our method overview figure, inspired by TensoRF [4]

Evaluation Region	Full Region		Masked Region		Inference
Metric	PSNR	SSIM	**PSNR**	SSIM	Time (Sec.)
Baseline Method [4]	19.87	0.69	24.07	0.81	14.50
Our Method	**20.73**	**0.71**	**25.54**	**0.82**	15.10

Table 12: **Quantitative comparison on ILSH dataset.** Results of our baseline and our method. Better results represented in bold.

7.2. Experimental Results

Experimental environment: We train our model using a single NVIDIA RTX 6000 Ada Generation GPU with 48GB of GDDR6 ECC memory and a batch size of 4096.

Phase	Resolution		Time
	Input	Output	
Training	3000×4096	3000×4096	24.20m
Rendering	3000×4096	3000×4096	15.10s

Table 13: **Processing time for training and rendering**

.

Methods	Masked-PSNR
Baseline	24.07
w/o. backviews	24.49
w/t. $\mathcal{L}_{occ}$	24.96
Final Method	**25.54**

Table 14: **Ablation study.** We experiment on how each idea contributes to the rendering quality of our final method. By using both ideas, we obtain the highest result.

Results : Table 12 shows the image synthesis metrics on the ILSH dataset. Our approach outperforms the baseline method in PSNR and SSIM scores, for both full region and masked region evaluation. Table 14 shows how our method improves our baseline. In Fig. 31, we present results for some of the captures used in the quantitative evaluation. The difficulty in estimating depth between a background and black hair leads to a significant number of floaters, primarily occurring in regions where the background and hair are close together. Our method effectively removes these artifacts positioned near the camera, thereby eliminating black artifacts from the rendered image. Ablation study on our ideas are shown in Fig. 29 and Fig. 30.

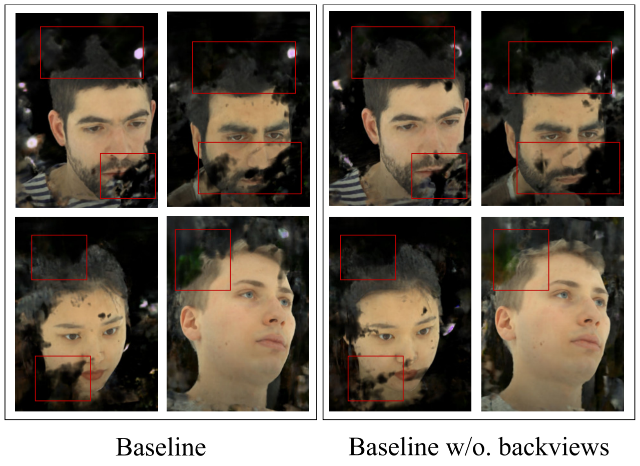


Figure 29: **Improvement when trained without backviews.** For sparse view settings such as ILSH, excluding certain views is beneficial for rendering novel views.

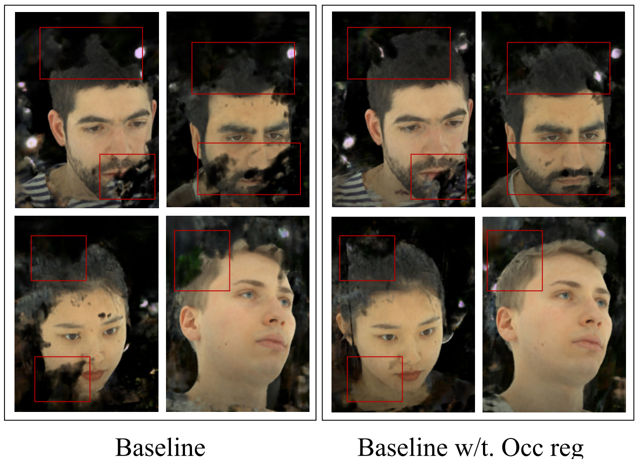


Figure 30: **Improvements using occlusion regularization.** Occlusion regularization is effective at removing floaters in the vicinity of the camera.

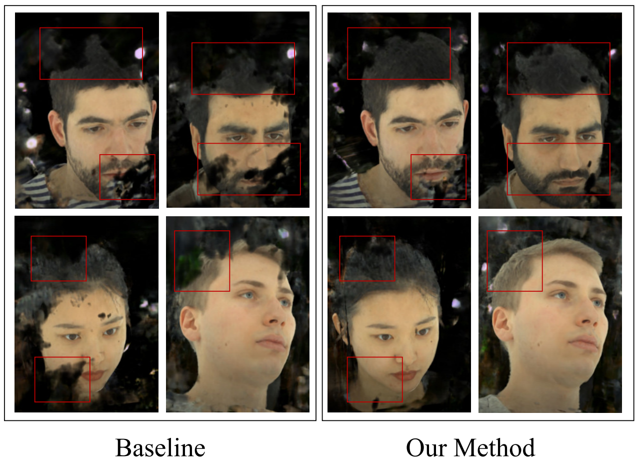


Figure 31: **Qualitative comparison on ILSH dataset.** Near-camera floaters are prominent in our baseline results while our method reconstructs images with fewer floaters.

7.3. Discussion

Limitation. Our method takes care of floating artifacts, leading to better geometry for the model to learn. However, our method struggles to capture high frequency details of the human face.

7.4. Acknowledgements

This work was supported by Artificial Intelligence Graduate School Program at Yonsei University [No. 2020-0-01361], KIST Institutional Program (Project No.2E32271-23-078) and Yonsei Signature Research Cluster Program of 2023 (2023-22-0008).

8. MS-TensoRF: Close to the GT partially

Team-6. xoft

Abstract: We defined that the core challenge of this competition is finding approaches for rendering novel view images with high-resolution images (3000*4069) and a few shots (22 images). First, we chose TensoRF as a baseline, which showed SOTA performances for rendering quality and memory efficiency. However, TensoRF had issues with rendering high-frequency features such as hair and skin texture. Therefore, we applied masked positional encoding of FreeNeRF and multisampling of Zip-NeRF to overcome high-quality problems. As a result, we improved qualitative results on the partial problems of faces, such as hair and skin texture.

8.1. Methodology

Baselines: The baseline is TensoRF [4] which was suggested by this competition organizer and had great capabilities: 1) superior rendering quality, and 2) a significantly lower memory footprint. For baseline application, TensoRF was trained based on subject-specific images and evaluated each subject individually. For a training step, we applied the near/far of the scene bound to [3.5, 5.5] without pose centering, the grid's size to 400^3 and others to the defaults as parameter setting. For a rendering step to find optimal near/far value to remove artifacts considering a distance from a camera, we set [3.55-3.6] for the near-scene bound. However, the baseline still had issues that were not able to render hair, skin texture, pupils and beards that required more detailed representation than other facial regions compared to ground true in high-frequency features.

MS-TensoRF: To overcome the issues of TensoRF, we developed a MultiSampling NeRF(called MS-NeRF (Fig. 32)) that was integrated with TensoRF's volume density (Fig. 28(c)), Zip-NeRF's multisampling [3] (Fig. 32(a)), and masked positional encoding of FreeNeRF[26](Fig. 32(b)). We combined Zip-NeRF, which used multisampling in each conical frustum by a hexagonal pattern similar to Gaussian, with TensoRF for improving color rendering of TensoRF. To further explain through Fig. 32, TensoRF computed the color for the interval points along a ray. But Zip-NeRF sampled points as the form of Gaussian between intervals and took the color's weighted means that represents the high frequency feature. Additionally, we applied masked positional encoding for improving overfitting issues to the sparse inputs. As a result, rendering novel views using 1) volume density of TensoRF and 2) color value which is sum of masked positional encoding result, and multisampling result, contributed for better representation of high frequency features compared to the baseline.

To be more specific, although TensoRF's article described that this approach had excellent performance for rendering quality, the results of baseline shown that rendering novel view images with high resolution images (3000*4069) and a few shots (22 images) had issues that did not represent the details of faces such as hair, skin texture, pupils and beard. In Fig. 33, since TensoRF updates the gradient only at points t_0 and t_1, it cannot model the GT (blue line) in detail by interpolating (orange arrow) for the color of the points between t. We thought that the causes were that TensoRF was able to overfit to the low frequency features and represented better performance for rendering quality, but did not represent the particular regions that required more detailed description in high frequency features. For example, the details of faces are blurry and couldn't get a representation of the details.

Firstly, we analyzed the results of baseline in a novel view image of Fig. 34 and depth image of Fig. 35, then we understood that volume density from TensoRF was enough for novel view representation. Therefore, we focused on the improvement of color representation in high frequency

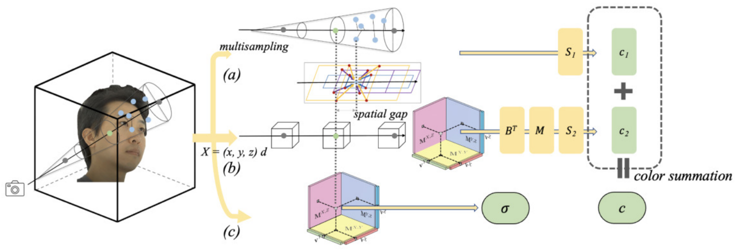


Figure 32: Overview of MS-TensoRF. B^T is the same process with the original TensoRF [4]. M is a process of masked positional encoding. S_1 and S_2 are MLPs composed of 3 hidden layers each.

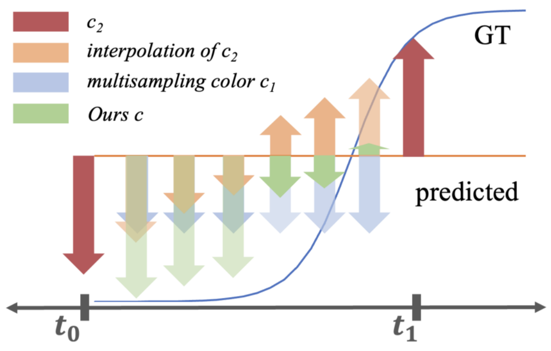


Figure 33: Detailed description of color summation from Fig. 32. We visualized for the boundary between the face and the background. The y-axis means color and the arrow means the gradient of color.

features. The increase of sampling using a lot of grids is a simple approach to improve the current issue, however there are pros and cons that this approach requires a lot of computing resources and time. Therefore, we combined multisampling of Zip-NeRF and volume density of TensoRF. Yet, partially the result of multisampling with baseline was close to the ground truth to represent skin texture and hair (Fig. 34 baseline + multisampling), but this created another problem with black holes like in Fig. 34. This issue is caused by spatial differences between sampling points from TensoRF (the sampling points on the ray used for volume density calculation) and Zip-NeRF (the multiple sampling points inside a conical frustum used for color calculation). To elaborate through Fig. 33, due to the spatial Gap of the sampling points of TensoRF and Zip-NeRF, the gradient calculated at t_0 is backpropogated to multisampled points, and the multisampled points belonging to the skin are trained as the background. Because of this reason, for an example, the color values of sampling points in the background (black) were affected to the color values of sampling points in the skin texture, then background color appeared in the skin texture regions. In other words, the volume density in 3D areas where actual volume density should have existed became empty because of spatial differences, generating artifacts such as black holes and blurry areas on the faces irregularly.

Next, to mitigate these irregular issues, we summed the color values of multisampling from Zip-NeRF and the color values of sampling points from TensoRF which are the same points for color and volume density calculation. To explain with Fig. 33, if the gradient for interpolation of TensoRF and the gradient of multisampling are summated, the final gradient can be updated to approximate GT. Through this, irregular issues have been greatly improved. Furthermore, we applied masked positional encoding from FreeNerf which used it to start with low frequency features and gradually train high frequency features to mitigate more the previous irregular issues, then we slightly improved the issues about irregular artifacts and blurry areas.

In conclusion, our approach used TensoRF for volume density representation. Multisampling contributed rendering of high frequency features to represent the details of faces. Then, for sampling points for color representation, we combined TensoRF, Zip-NeRF, and FreeNeRF to mitigate irregular black holes and blurry areas. As a results, we were able to achieve high quality images that better represent high frequency feature compared to the baseline.

8.2. Experimental Results

Experimental environment: We trained 22 images and created our model at 50000 iterations for novel view synthesis on an A10 GPU, then rendered novel views using camera pose information on the same GPU.

Steps	Baseline	Ours (sec.)
ray sampling	149	149
volume density rendering	22	22
Zip-NeRF sampling	0	18
Zip-NeRF RGB rendering	0	13
TensoRF RGB rendering	90	90
volume rendering	466	466
Total	727	758

Table 15: Processing time for each step.

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metrics	PSNR	SSIM	**PSNR**	SSIM	
Baseline [4]	20.09	0.65	25.30	0.81	727
Ours	20.01	0.64	25.02	0.80	758

Table 16: Results of baseline and ours on the ILSH dataset

Quantitative Results: Our approach (MS-TensoRF) presented slightly underperformed PSNR and SSIM scores in both the full and masked regions. Even though we tried to find optimal approaches for sampling points of volume density and color representation using MS-TensoRF, we could not outperform on a quantitative basis respect to the baseline. While PSNR and SSIM scores were lower than the baseline, we adopted MS-TensoRF in context of coordinating quantitative scores based on PSNR and SSIM and qualitative evaluation based on human eyes. In the next part, we'll specify why our algorithm didn't scored better in terms of quantitative results.

Qualitative Results: All qualitative results we implemented are shown in the Fig. 34. In the second column, TensoRF rendered the blurry images. This methodology made the beard flat and couldn't represent the lighting in the pupils at all. The result images in the third column which we applied multisampling on TensoRF are expressed high frequency features better than baseline, but it causes translucent areas that blend into the background and fuzzy areas resulted from spatial gap. Additionally, we could understand and verify more details about the generation of irregular artifacts from depth image in the Fig. 35. And our results in the last column of Fig. 34 rendered same level of the details with the former one with only very localized areas of quality loss. Although that remained degraded areas are difficult to see and our images seem more realistic with the naked eyes, that regions made the PSNR and SSIM of our images lower than the images from baseline.

8.3. Discussion

Limitation: MS-TensoRF was developed to outperform TensoRF's capabilities through integrating advantages of

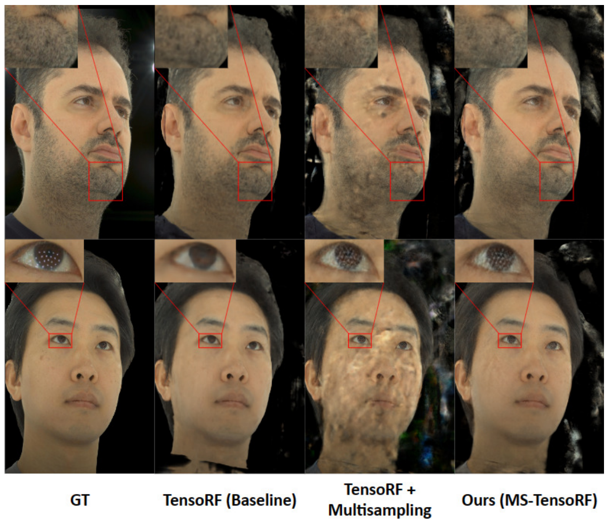


Figure 34: Qualitative result of the two portraits of Imperial Light-Stage Head (ILSH) Datasets. Note how detailed and realistic the beard(top) and pupils(bottom) are in our approach compared with TensoRF.

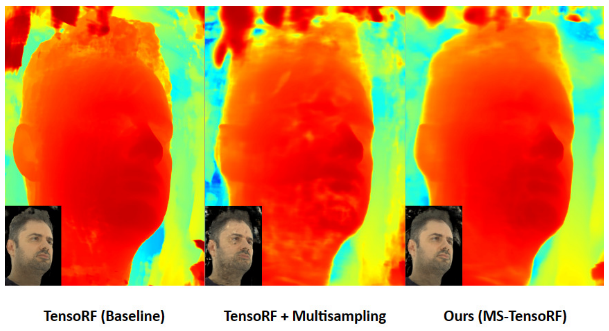


Figure 35: Depth images of the methodologies we implemented. If the color is close to red, the the distance of that pixel is near and if the color is close to blue the distance of that pixel is far.

TensoRF, Zip-NeRF, and FreeNeRF. Theoretically, our idea had the possibility to surpass the baseline results, however, we couldn't find the best solution fundamentally to remove spatial differences from sampling points of volume density and color. Therefore, partially our results looked better than baseline results to human eyes recognition. Yet, quantitative results of our results as PSNR and SSIM underperformed baseline results because of irregular black holes and blurry areas from spatial differences of sampling points.

Further Improvement: In this competition, we focused on the development of MS-TensoRF especially with lower memory usage compared to the original Zip-NeRF approach. Currently, there are two different sampling structures based on the grid of TensoRF and Zip-NeRF. As a future work, we will integrate these two different sampling structures into one with memory efficiency of TensoRF and high quality of Zip-NeRF into MS-TensoRF. We will move this research in a direction where general and productive results such as realizing metaverse move to the real world beyond the research area.

8.4. Acknowledgement.

We gratefully acknowledge the computing resources provided on AWS (Amazon Web Service) supported by AI Research Laboratory of K-water (Korea Water Resources Corporation) Research Institute.

9. Realistic 3D Head Avatar Generation using Neural Implicit Function based Estimation of Warping Field and Textures

Team-7. KHAG: KIST-Head Avatar Generator

Abstract This is the factsheet of the Realistic 3D Head Avatar Generation using Neural Implicit Function based Estimation of Warping Field and Textures. We used the Nvdiffrec [19] as our baseline. We suggest a new method to get realistic 3D head avatars from a small number of head images. We obtain a warping field that adjusts the vertices of a base mesh instead of directly optimizing an SDF and mesh vertex position to get a clean geometry mesh. Additionally, we use a relative Laplacian loss, a relative normal loss, an elastic loss, and a volume maximization loss to prevent distortion, folding, and noise on the mesh. Also, we use a uniform light assumption and hash-grid MLP-based texture estimation instead of directly optimizing light map and texture maps to remove the color artifacts. Moreover, we apply the range limit on textures to remove the remaining color artifacts. Unlike other NeRF-like methods, our method aims to get not only good rendered images, but also get a useful mesh model. The output model of our method preserves the topology of the base head mesh. If we had the blend shapes for the base head mesh, we could easily control the mesh model to make various expressions. Also, our system can easily generate texture maps(diffuse, material, and normal maps) using the MLP model. So, by simply modifying the texture map, we can change the final model's color or material properties. The output mesh model is compatible with common graphics applications like Blender.

9.1. Methodology

Baselines (Nvdiffrec [19]): Nvdiffrec is a method that reconstructs an arbitrary mesh model and texture maps from the multiple images using SDF estimation, texture

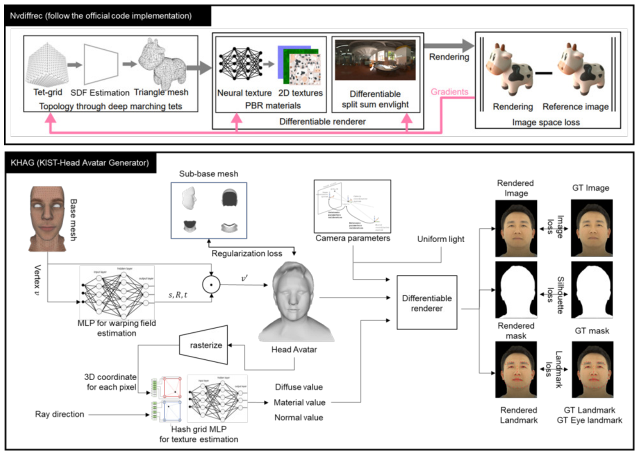


Figure 36: System architecture of baseline(Nvdiffrec [19]) and Ours.

	Nvdiffrec [19]		Ours
	1st stage	2nd stage	All stages
Geo-metry	SDF estimation	Mesh vertex direct optimization	Base Mesh based Warping field(transform) estimation
Color	Hash grid MLP based estimation	Texture map optimization	Hash grid MLP based estimation w/ ray direction input
	Light map optimization		Uniform light assumption
Loss	Img based loss Light regularization		Img based loss Light regularization Relative Laplacian Relative normal Elastic Volume maximization
	SDF regularization	Relative Laplacian	

Table 17: Key differences between Nvdiffrec and our method.

MLP, mesh optimizatoin, and texture/light map optimization with differentiable rasterisation/rendering. For performance comparison, we use official GitHub code to run the Nvdiffrec. Note that, unlike the paper, the official implementation only supports explicit SDF estimation, excluding estimation using MLP. Nvdiffrec requires a foreground mask to train the model. We acquire foreground masks using the RITM segmentation [22]. Nvdiffrec consists of two stages. In the first stage, SDF, textures, and a light map are learned. Due to memory-related issues, 1/4 resolution images are used instead of the original images. For other parameters, we use default values. In the second stage, mesh models and texture maps are generated from the acquired SDF and textures, respectively, and then they are refined. Due to memory-related issues, the original images were cropped and used for learning (the resolution was not changed). Parameters in learning are as follows: 'batch size: 4', 'iteration: 7000', 'dmtet_grid: 64, 'learning rate: [0.01, 0.001]', 'mesh_scale: 0.5', 'train_res: [1024,720], and 'texture_res: [4096,4096]. To make the background, we extract the background from each image using the foreground mask. We get the mean background image from the extracted background images.

Ours (Getting realistic head avatar): We found that the learned model in Nvdiffrec had a noisy geometry by optimizing the SDF and mesh vertex position directly. So, instead of using direct optimization, we obtain a warping field that adjusts the vertices of a base mesh. The warping field is represented by transforms that are defined by scale, rotation, and translation parameters. We estimate the warping field by using a MLP model that takes the vertex position as the input. Also, we found that color estimation by directly optimizing the texture maps and the light map could generate many artifacts when there were only a small number of training images. So, assuming uniform light, we use a hash-grid based MLP model that takes a vertex position and a ray direction as inputs for estimating diffuse, material and normal values. Moreover, we apply the range limit on material and normal values to remove the remaining color artifacts. We also found that we needed to use more regularization loss to prevent distortion, folding and noise on the mesh. Thus, a relative Laplacian loss, a relative normal loss, an elastic loss and a volume maximization loss are used. Fig. 36 shows the system architecture of the baseline and ours. Table 17 summarizes the key differences in methodology between the baseline and our method.

Ablation studies: To check the effect of our component, we perform ablation studies. We divide the cases into two, which are related to geometry and color, respectively. Ours$_{rw}$ represents the case of using the base mesh without regularization loss and Warping field estimation. Ours$_{w}$ represents the case where the base mesh and regularization are adopted without Warping field estimation. In case of Ours$_{rw}$ and Ours$_{w}$, the vertex positions of the base mesh are directly optimized to reconstruct the geometry. Ours$_{rurl}$ represents the case of using the texture MLP without ray direction, uniform light, and texture map range limit. Ours$_{rl}$ represents the case of using the texture MLP, ray direction and uniform light without texture range limit. Table 20 summarizes the differences between each ablation case ('Ray d', 'U-light', and 'T-RL' represent usage of ray direction uniform light estimation, and texture range limit, respectively).

9.2. Experimental Results

Experimental environment: Table 18 describes our experimental environment. We denote the detailed processing time in Table 19. The preparing environment step is run only once for each subject. Therefore, when generating a

CPU	Intel Core i9-10920X
RAM:	64 GB
GPU	NVIDIA RTX A6000/PCIe/SSE2
OS	Ubuntu 18.04.6 LTS 64-bit
Cuda version	11.3
Python version	3.8.12

Table 18: Experimental environment.

Steps	Resolution		Time (Sec.)
	Input	Output	
Prepare environment (ex. build light)	-	-	2.352
Neural Rendering	3000×4096	3000×4096	0.227
Total	3000×4096	3000×4096	2.579

Table 19: Processing time for each step.

Methods	Base mesh	Regular -ization	Warp -ing	Texture MLP	Ray d U-light	T-RL	Masked -PSNR
Ours$_{rw}$	o	x	x	o	o	o	20.75
Ours$_{w}$	o	o	x	o	o	o	22.24
Ours$_{rurl}$	o	o	o	o	x	x	23.86
Ours$_{rl}$	o	o	o	o	o	x	22.39
Ours	o	o	o	o	o	o	22.78

Table 20: Setting and a masked-PSNR score for each ablation study on sub-dataset (for the first three subjects).

Evaluation Region	Full Region		Masked Region		Time (Sec.)
Metric	PSNR	SSIM	**PSNR**	SSIM	
Nvdiffrec [19]	20.03	0.62	22.96	0.78	0.222
Ours	22.14	0.64	23.39	0.79	2.579

Table 21: Quantitative results of the baseline and our method.

plurality of novel view images for one subject, the preparation step may be skipped after the first image is generated.

Quantitative and Qualitative Results: Table 21 shows that our method gets better quantitative scores than the baseline in every case. Fig. 37 shows qualitative results. It can be noticed that our method has better results than the baseline. We can especially see that our results have much cleaner surfaces and no artifacts on textures.

Quantitative and Qualitative Results (Ablation studies): Table 20 denotes the PSNR of each ablation study. Fig. 38 and 39 show the rendered result of each ablation study. In Fig. 38, we clearly see that each component can help to improve the geometry quality. The figures show that using a base mesh instead of a SDF removes clutter from empty space. Adding the regularization reduces the noise on the

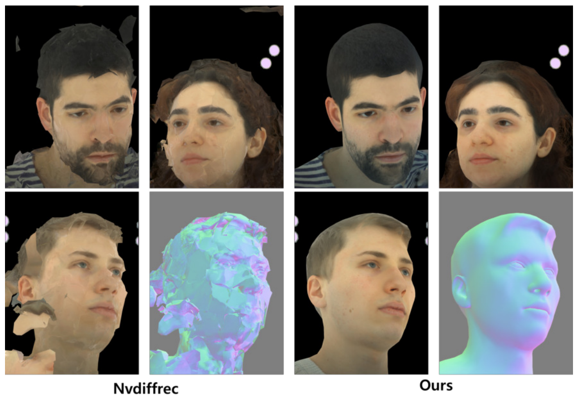


Figure 37: A few examples for qualitative comparison.

surface. By using warping field estimation instead of direct vertex optimization, we can get more clean surfaces. As a result, we can gain the PSNR. In Fig. 39, we can notice that the usage of a texture range limit raises the PSNR. We can see some color artifacts (refer the red box in Fig. 39) without the texture range limit removed. Note that the result without uniform light assumption and ray direction input can have better PSNR score as can be seen in Table 20 (Ours$_{rurl}$ case). However, we can see the artifacts in the output figure (Ours$_{rurl}$ case). We think that the uniform light assumption and ray direction input solve the color artifacts, but they can lead to underfitting.

9.3. Advantages of Our methods

Unlike other NeRF-like methods [17], our method aims to get not only good rendered images but also a useful mesh model. The output model of our method preserves the topology of the base head mesh. If we had the blend shapes for the base head mesh, we could easily control the mesh model to make various expressions. Also, our system can easily generate the texture maps (diffuse, material, and normal maps) using the MLP model. So, by simply modifying the texture map, we can change the final model's color or material properties. The output mesh model is compatible with common graphics applications like Blender.

9.4. Discussion

Limitation. Our method requires a base head mesh to reconstruct the head model. If we want to reconstruct other objects, a base mesh with a different shape is required. Unlike NeRF, which performs volume-based reconstruction, we use mesh-based reconstruction. So it is difficult to restore geometry that is significantly different from the topology of the base mesh (ex., thin hair, very complex surfaces).

Also, we found that the uniform light assumption and using ray direction as an input to the model decreased the PSNR. So, we think that we need more advanced methods

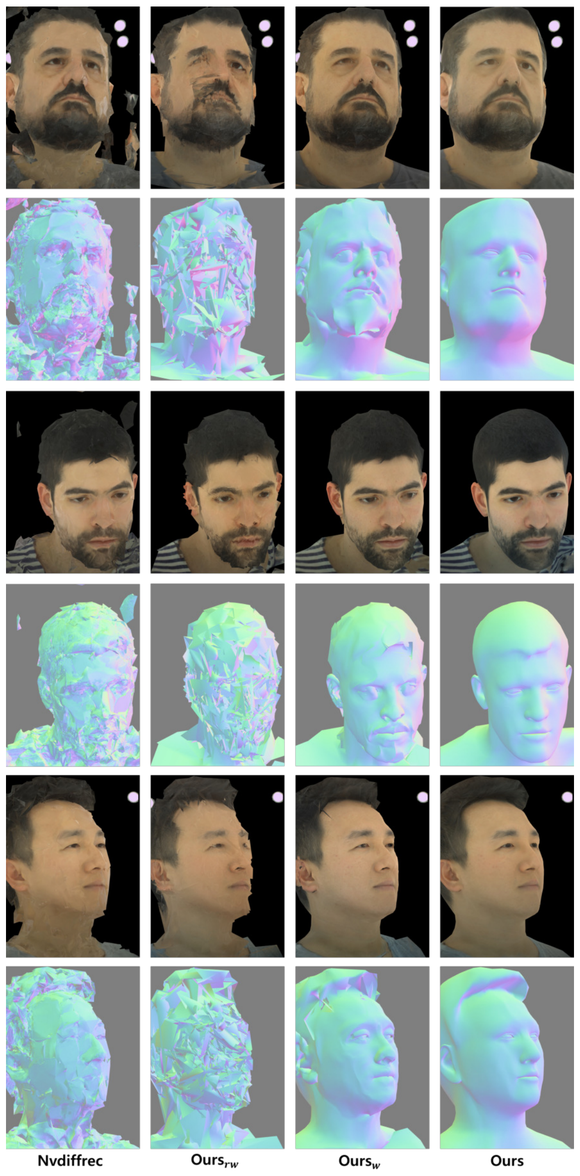


Figure 38: A few examples for qualitative comparison of ablation study (geometry).

for estimating the light when there are only a small number of training images.

9.5. Acknowledgment

This work was supported by the Korea Institute of Science and Technology (KIST) Institutional Program (ProjectNo. 2E32291).

References

[1] EasyMoCap - Make human motion capture easier. Github, 2021. 4, 5

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 13, 14

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023. 17

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 6, 8, 9, 10, 15, 17, 18

[5] Yikang Ding, Wentao Yuan, Qingtian Zhu, Haotian Zhang, Xiangyue Liu, et al. TransMVSNet: Global Context-aware Multi-view Stereo Network with Transformers. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 4

[6] Andreea Dogaru, Andrei-Timotei Ardelean, Savva Ignatyev, Egor Zakharov, and Evgeny Burnaev. Sphere-guided training of neural implicit surfaces. In *CVPR*, 2023. 11, 12

[7] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. pages 2367–2376, 2019. 1

[8] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proceedings of Machine Learning and Systems 2020*, pages 3569–3579. 2020. 7, 10

[9] Yuan-Chen Guo. Instant neural surface reconstruction, 2022. https://github.com/bennyguo/instant-nsr-pl. 7

[10] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *CVPR*, 2022. 10

[11] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2014. 5, 9

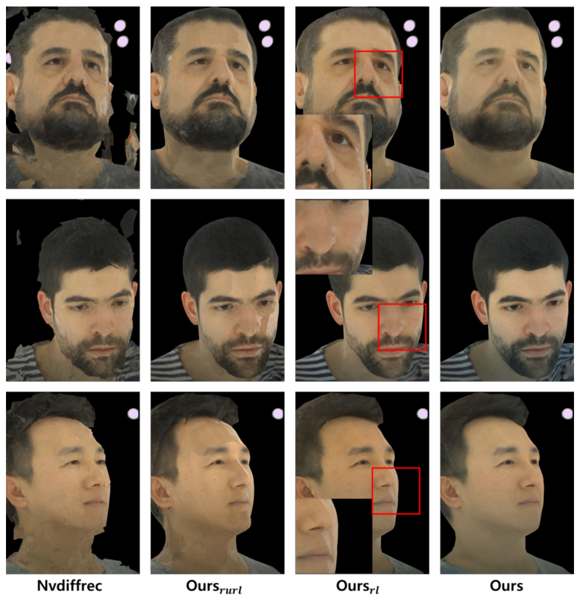


Figure 39: A few examples for qualitative comparison of ablation study (color).

[12] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 6, 13

[13] Ruilong Li, Hang Gao, Matthew Tancik, and Angjoo Kanazawa. Nerfacc: Efficient sampling accelerates nerfs. *arXiv preprint arXiv:2305.04966*, 2023. 9

[14] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *SIGGRAPH Asia*, 2017. 11

[15] Shanchuan Lin, Linjie Yang, Imran Saleemi, and Soumyadip Sengupta. Robust High-Resolution Video Matting with Temporal Guidance. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022. 4, 5

[16] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Gr.*, 38(4):1–14, 2019. 1

[17] 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 *ECCV*, 2020. 7, 10, 11, 21

[18] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 7

[19] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8280–8290, June 2022. 19, 20, 21

[20] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022. 10

[21] Malte Prinzler, Otmar Hilliges, and Justus Thies. Diner: Depth-aware Image-based NEural Radiance fields. In *Computer Vision and Pattern Recognition (CVPR)*, 2023. 4, 5

[22] Konstantin Sofiiuk, Ilya A Petrov, and Anton Konushin. Reviving iterative training with mask guidance for interactive segmentation. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3141–3145. IEEE, 2022. 20

[23] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. pages 517–524, 1998. 1

[24] Thomas Tanay, Aleš Leonardis, and Matteo Maggioni. Efficient view synthesis and 3d-based multi-frame denoising with multiplane feature representations. *CVPR*, 2023. 1

[25] 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. *NeurIPS*, 2021. 7, 10, 11, 12

[26] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15, 17

[27] Jingfeng Yao, Xinggang Wang, Shusheng Yang, and Baoyuan Wang. Vitmatte: Boosting image matting with pretrained plain vision transformers. *arXiv preprint arXiv:2305.15272*, 2023. 6

[28] Yinglin Zheng, Hao Yang, Ting Zhang, Jianmin Bao, Dongdong Chen, Yangyu Huang, Lu Yuan, Dong Chen, Ming Zeng, and Fang Wen. General facial representation learning in a visual-linguistic manner. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 18697–18709, 2022. 4, 5

[29] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Trans. Gr.*, 37(4):1–12, 2018. 1

[30] Hao Zhu, Haotian Yang, Longwei Guo, Yidi Zhang, Yanru Wang, et al. FaceScape: 3D Facial Dataset and Benchmark for Single-View 3D Face Reconstruction. *arXiv preprint arXiv:2111.01082*, 2021. 4