Neural Visibility Field for Uncertainty-Driven Active Mapping

Supplementary Material

A. Method Details

A.1. NVF architecture and training details

NVF is an augmentation of a NeRF consisting of two additional MLP heads for predicting RGB variance and visibility. Specifically, we implement NVF on top of a nerfstudio [10] implementation of Instant-NGP [6], where the color MLP head represents $\boldsymbol{\mu}_c$. Alongside the color head is a MLP head for RGB variance, outputting a 3x1 vector $\boldsymbol{Q}_c$. Similarly, the visibility MLP head is attached alongside the density head. For a visualization of the architecture, see Appendix Fig. 1. In practice, we train Instant-NGP, variance, and visibility separately and in sequence. First, we train the NeRF backbone for 5000 iterations using a learning rate of 0.01 and 4096 rays per batch. Next, the variance head is trained for 500 iterations using a learning rate of 0.001 and 4096 rays per batch. Finally, the visibility head is trained for 500 iterations using a learning rate of 0.001 and 65536 samples per batch. We train all modules using the Adam optimizer [5].

A.2. Entropy computation details

Joint Entropy of the Camera Observation. We discuss the details on the computation of the joint entropy $\mathcal{H}(\boldsymbol{Z_p})$ as formulated in in Eq. (13). For simplicity in this discussion, we denote the joint entropy as $\mathcal{H}(\boldsymbol{Z})$ in this section. We model the joint observation of all rays as a Bayesian network, where the observation of each pixel only depends on its adjacent neighboring pixel. Consequently, the joint probability can be factorized as $p(z) = \prod_{mn} p(z_{mn}|z_{m+1,n}, z_{m,n+1})$. Note that for the sake of brevity, boundary terms where a pixel lies at the image edge are omitted here. Then, by applying the chain rule of entropy[1], we obtain:

$$\mathcal{H}(\boldsymbol{Z}) = \sum_{m,n} \mathcal{H}(\boldsymbol{Z}_{mn}|\boldsymbol{Z}_{m+1,n}, \boldsymbol{Z}_{m,n+1}) \tag{1}$$

We then apply the inequality $\mathcal{H}(\boldsymbol{Z}_{mn}|\boldsymbol{Z}_{m+1,n}, \boldsymbol{Z}_{m,n+1}) \leq \mathcal{H}(\boldsymbol{Z}_{mn}|\boldsymbol{Z}_{m+1,n})$ and $\mathcal{H}(\boldsymbol{Z}_{mn}|\boldsymbol{Z}_{m+1,n}, \boldsymbol{Z}_{m,n+1}) \leq \mathcal{H}(\boldsymbol{Z}_{mn}|\boldsymbol{Z}_{m,n+1})$. These allow us to derive an upper bound for $\mathcal{H}(\boldsymbol{Z})$:

$$\mathcal{H}(\boldsymbol{Z}) \leq \sum_{m,n} \frac{1}{2}\big(\mathcal{H}(\boldsymbol{Z}_{m,n}|\boldsymbol{Z}_{m+1,n}) + \mathcal{H}(\boldsymbol{Z}_{m,n}|\boldsymbol{Z}_{m,n+1})\big) \tag{2}$$

Then we connect the conditional entropy with the correlation between the two adjacent rays. We let

$$\rho_{m+1,n} = 1 - \frac{\mathcal{H}(\boldsymbol{Z}_{m,n}|\boldsymbol{Z}_{m+1,n})}{\mathcal{H}(\boldsymbol{Z}_{m+1,n})} \tag{3}$$

where ρ is as a measure of correlation. Specifically, a ρ value closer to 1 indicates $\boldsymbol{Z}_{m,n}$ and $\boldsymbol{Z}_{m+1,n}$ are strongly correlated, whereas a ρ tends to 0 suggests they are not correlated. It is important to note that this definition of correlation, based on entropy, differs from the widely recognized Pearson correlation coefficient, and is commonly used in quantum information[2]. Consequently, we can obtain the upper bound:

$$\mathcal{H}(\boldsymbol{Z}) \leq \sum_{m,n} (1 - \rho_{mn}) \mathcal{H}(\boldsymbol{Z}_{mn}) \tag{4}$$

We assume that the correlation between two points in the scene can be modeled as a function of their spatial distance, a concept commonly referred to as the correlation function in statistical physics [9]. We adopt a truncated least-square form for this correlation function.

$$\rho(x) = \begin{cases} 1 - (\frac{x}{\xi})^2 \text{ , if } x < \xi, \\ 0 \text{ , otherwise} \end{cases} \tag{5}$$

This formula indicates that two points located within a distance threshold ξ of each other are strongly correlated, whereas those beyond this threshold are considered independent. It is noteworthy that this term bears resemblance to the correlation function $\rho(x) = \exp(-\frac{x}{\xi})$, which is commonly applied in statistical physics [9], and ξ represents the correlation length. Empirical evaluations indicate that the use of either correlation function expression significantly outperforms the scenario where all rays are assumed to be independent ($\rho = 0$). Notably, a marginal improvement was observed when utilizing Eq. 5.

Therefore, we can approximate the correlation between two adjacent rays based on their expected depth, expressed as $\rho_{mn} = \rho(d_{mn}\Delta\phi)$, where $\Delta\phi$ is the angular resolution of each pixel, d_{mn} is the expected depth of ray at pixel (m, n). This implies that when the camera is closer to an object, the observations in adjacent pixels of the camera exhibit stronger correlation. Hence the actual total information gain is smaller than the sum of the information gain of each pixel. Accordingly, the correction function f_{corr} in Eq. 13 can be defined as:

$$f_{corr}(\mathcal{H}(\boldsymbol{Z}_{mn}); d_{mn}) = \rho(d_{mn}\Delta\phi)\mathcal{H}(\boldsymbol{Z}_{mn}) \tag{6}$$

In our experiments, we let the correlation length $\xi = kD\Delta\phi$, where D represents the diameter of the coarse bounding box enclosing the object, and k is a hyperparameter, and we let $k = 0.25$.

Entropy of GMM. We then introduce the details to compute the entropy for each ray, which is modeled as

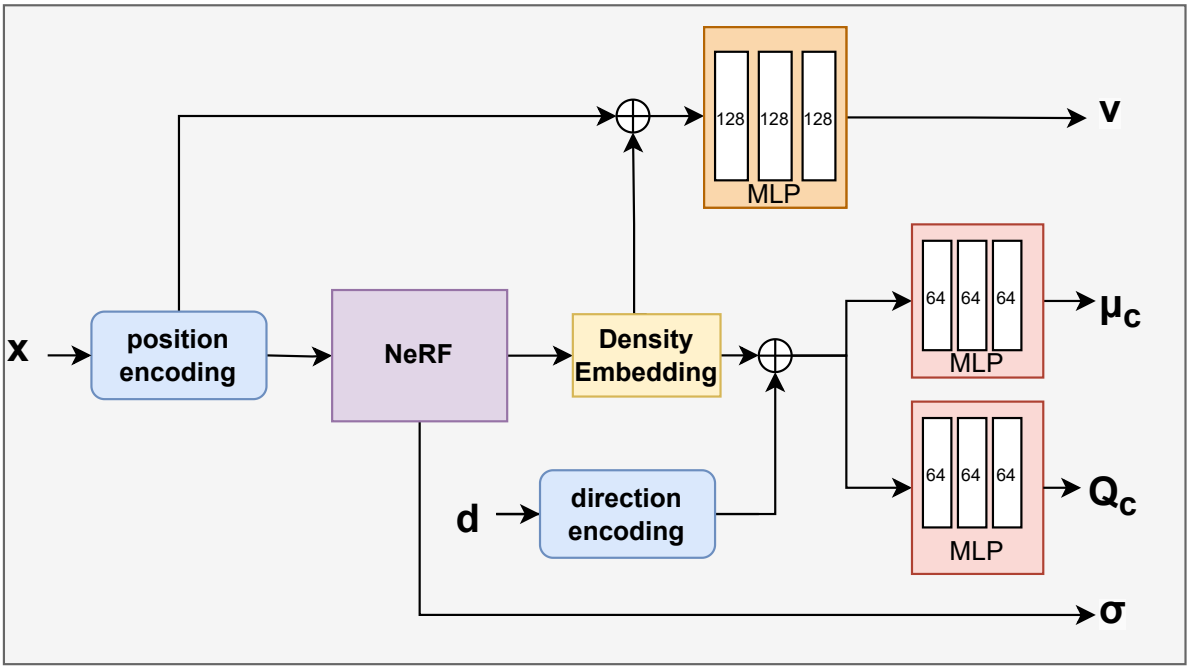

Figure 1. **NVF Architecture:** The MLP block consists of fully connected layers that use the ReLU activation function. The numbers inside the block denote the size of the layer. The final output from the visibility (v) MLP and RGB (μ_c) MLP are passed through the sigmoid activation function while the RGB Variance (Q_c) MLP uses softplus activation

a Gaussian Mixture Model (GMM). For the sake of simplicity, we denote the GMM's distribution as $p(\boldsymbol{x}) = \sum_i w_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{Q}_i)$. We use the upper bound proposed in [4] to closely approximate the entropy of the GMM $\mathcal{H}(\boldsymbol{X})$:

$$\mathcal{H}(\boldsymbol{X}) \leq \sum_i w_i \left(-\log w_i + \frac{1}{2} \log \left((2\pi e)^N |\boldsymbol{Q}_i| \right) \right) \quad (7)$$

This upper bound is expected to provide a more accurate approximation of the true entropy of the GMM compared to the conventional method which approximates the entropy using a single Gaussian that matches the first two moments of the GMM[3], given by $\mathcal{H}(\boldsymbol{X}) \leq \frac{1}{2} \log \left((2\pi e)^N |\boldsymbol{\Sigma}| \right)$ where $\boldsymbol{\Sigma}$ is calculated as:

$$\boldsymbol{\Sigma} = \sum_i w_i \left(\boldsymbol{Q}_i + (\boldsymbol{\mu_i} - \bar{\boldsymbol{\mu}})(\boldsymbol{\mu_i} - \bar{\boldsymbol{\mu}})^T \right) \quad (8)$$

and $\bar{\boldsymbol{\mu}}$ is the weighted mean of the Gaussian components, defined as $\bar{\boldsymbol{\mu}} = \sum_i w_i \boldsymbol{\mu_i}$. It is worth mentioning that the baseline method [7, 8] use the weighted average of position-based color variance to approximate the rays-based observation variance by employing a single Gaussian whose mean and variance are the weighted averages of the means and variances of all samples along the rays, respectively ; in other words, $\boldsymbol{\Sigma} = \sum_i w_i \boldsymbol{Q}_i$. This approach resembles the first term in Eq. 8 but misses the covariance term $(\boldsymbol{\mu_i} - \bar{\boldsymbol{\mu}})(\boldsymbol{\mu_i} - \bar{\boldsymbol{\mu}})^T$. Additionally, it does not take into account the visibility to the training views.

In summary, we derive an upper bound for the pixel-wise entropy, and consequently, for the joint entropy of each view, this upper bound is utilized to closely approximate the information gain at a given pose. In the planning phase, given a candidate pose, we first apply Appendix Eq. 7 to compute the entropy for each ray, subsequently, we compute the joint entropy of the image observation as per Eq. 13 at that pose, which then serves as a reward function in the planning process.

A.3. Active mapping implementation details

Active Mapping Pipeline. To train NVF within an active mapping framework, we build our pipeline on top of nerfstudio [10] and NerfBridge [12]. Every time a new view is added to NVF, the model is trained from scratch on the collection of its observed views.

After training, we sample candidate poses in the scene, without collision with the object, by filtering all poses within a density threshold. In the Room scene, the sampler additionally thresholds for collisions between view poses and the current pose, to make sure the agent could move to the new pose without collision. After candidate view poses are generated, NVF computes the entropy of each pose. The view with the highest entropy is next rendered in the scene and added to the observations. This procedure repeats until

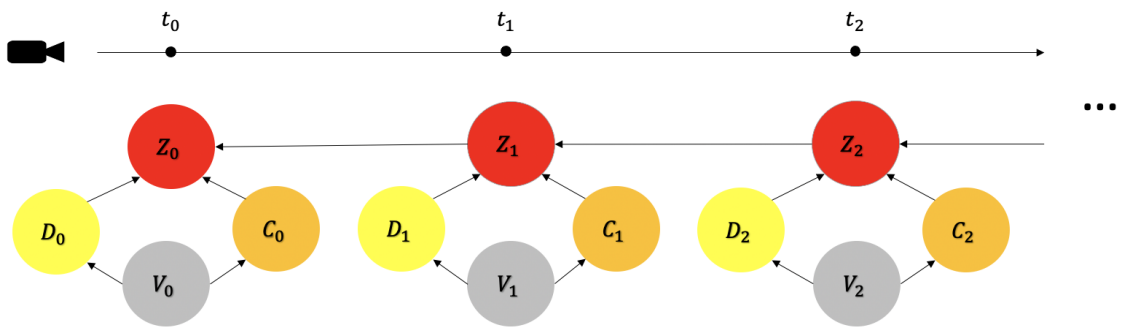

Figure 2. Schematics of the proposed Bayesian Network: Z represents the observed (ray-based) color, C represents the emitted (position-based) color, D represents if the interval is occluded, V represents the visibility,

.

the horizon step is met, as is shown in Alg. 2. In the experiments, we sample $N = 512$ candidate views and run the active mapping for 20 steps; the evaluations are performed after the last planning step.

Gradient-based Optimization for Planning. In addition to the method of finding the best view among a randomly sampled candidate poses set, we also performed experiments on 6 DoF pose-refinement on the camera poses, $\boldsymbol{p} \in SE(3)$, as the entropy function $\mathcal{H}$ is a fully differentiable differentiable function of $\boldsymbol{p}$. We find the optimal $\boldsymbol{p}$ such that

$$\boldsymbol{p}^* = \underset{\boldsymbol{p} \in SE(3)}{\arg\max}\, \mathcal{H}(\boldsymbol{Z_p}) \tag{9}$$

We first find the top k poses with the highest entropy $\mathcal{P}_k$ and perform gradient-based optimization to refine the poses. To reduce the size of the computation graph and the memory requirements, a subset of pixels $\boldsymbol{Z}_i \subset \boldsymbol{Z_p}$ with an image is used to estimate the expected entropy, instead of the full image. We perform backpropogation on this estimated entropy using an Adam optimizer with a learning rate of $1e-4$, to find the optimum pose.

Algorithm 1 Active Mapping with NVF

Input:
  $\mathcal{P} \leftarrow$ initial poses
  $\boldsymbol{Z} \leftarrow$ initial images
for $i = 1$ to $n_{horizon}$ **do**
  $F_\Theta \leftarrow$ trainNVF($\mathcal{P}$, $\boldsymbol{Z}$) ▷ train NVF
  $\mathcal{P}_c \leftarrow$ samplePoses(F_Θ) ▷ sample candidate poses
  $\boldsymbol{p}_i \leftarrow \underset{\boldsymbol{p} \in \mathcal{P}_c}{\arg\max}\, \mathcal{H}(\boldsymbol{Z_p}|F_\Theta)$
  $\mathcal{P} \leftarrow \{\boldsymbol{p}_i\} \cup \mathcal{P}$
  $\boldsymbol{Z} \leftarrow$ takeImageAt($\{\boldsymbol{p}_i\}$) $\cup\, \boldsymbol{Z}$ ▷ update training set
return F_Θ

Algorithm 2 Gradient-Based Optimization for Planning

Input:
  $\mathcal{P} \leftarrow$ sampled poses
$\mathcal{P}_k \leftarrow$ getTopKPoses($\mathcal{P}, \mathcal{H}$)
for $i = 1$ to $n_{iterations}$ **do**
  for $\boldsymbol{p}$ in $\mathcal{P}_k$ **do**
    $\boldsymbol{Z_p} \leftarrow$sampleRays($\boldsymbol{p}$)
    $\boldsymbol{p} \leftarrow \boldsymbol{p} + \eta \frac{\partial(\mathcal{H}(\boldsymbol{Z_p}|F_\Theta))}{\partial \boldsymbol{p}}$
$\tilde{\boldsymbol{p}} = \underset{\boldsymbol{p} \in \mathcal{P}_k}{\arg\max}\, \mathcal{H}(\boldsymbol{Z_p}|F_\Theta)$
return $\tilde{\boldsymbol{p}}$

B. Experiments Details

B.1. Uncertainty Estimation details

As for the entropy comparison experiments shown in Fig. 3 of the main paper, Appendix Fig. 3 provides an illustration of the pose of the training views and evaluation views.

B.2. Mesh metrics implementation details

For computing Accuracy, Completion, and Completion Ratio metrics, ground truth points are sampled from the ground truth scene meshes. Points from NVF's reconstructed mesh are sampled from the observation view rays. Accuracy measures the mean distance of sampled points from the reconstructed mesh to the nearest corresponding points in the ground truth mesh. Completion instead measures the mean distance of sampled ground truth points to the nearest reconstructed mesh points. Completion Ratio calculates the percentage of completion distances being below a threshold. For the original NeRF assets and Hubble scene, the threshold is set to 0.01. For the Room scene, as the scale is larger, the threshold is set to 0.1.

Visual coverage quantifies the surface area a trajectory of views covers a scene. We compute this with rasterization. Given a ground truth mesh of the scene, we project the mesh

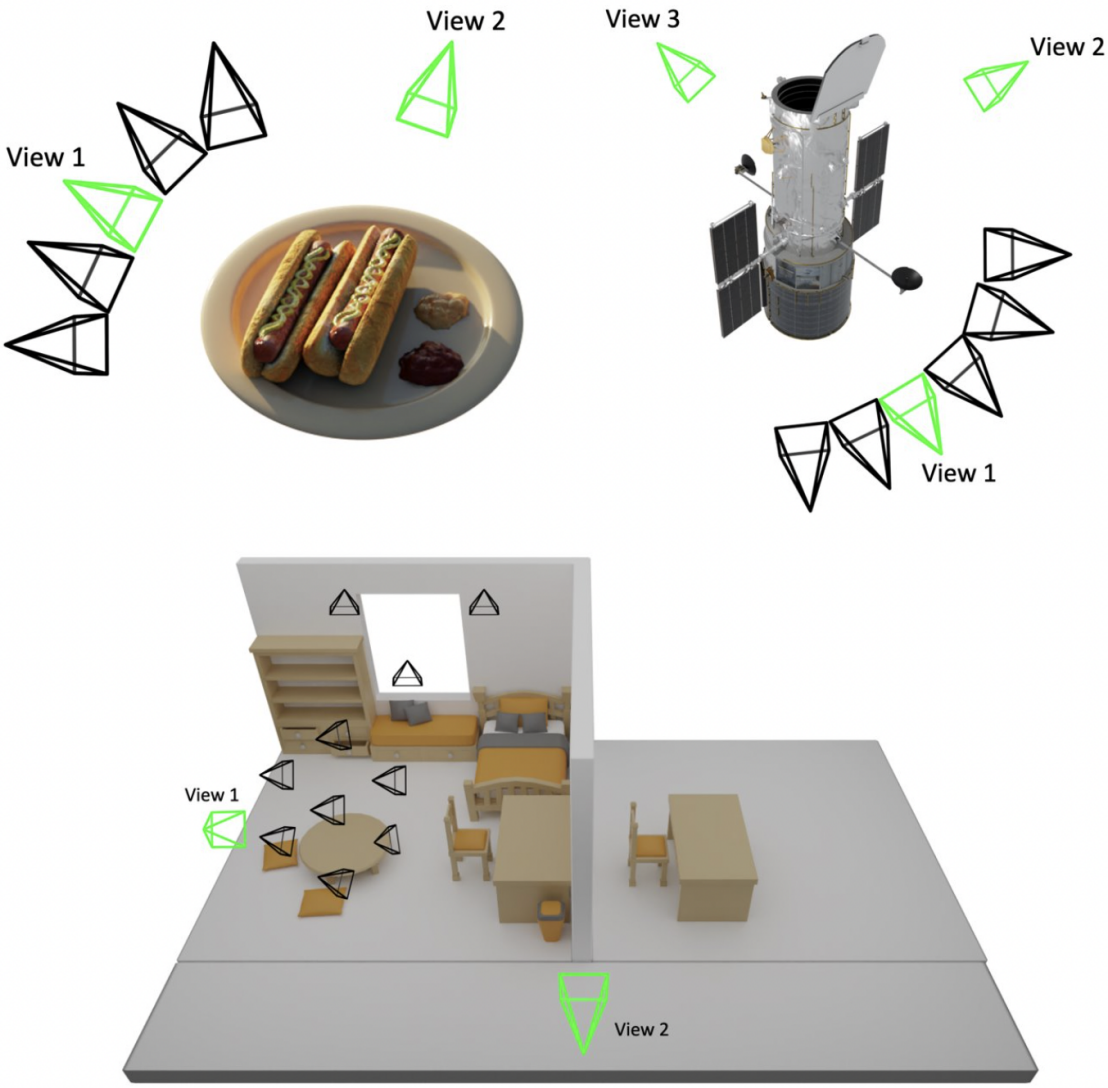

Figure 3. **Uncertainty Experiment Scene Setups:** Illustration of the training views and evaluation views in Fig. 3. The black frustums correspond to the training views, the green frustums are the evaluation views. For more video results, please refer to https://sites.google.com/view/nvf-cvpr24/

.

onto all of the observation views. In each rendered image, we record the number of mesh faces visible to the corresponding view. We append all observed faces to a visible set. Computing visual coverage is then the ratio of the length of the visible set to the total number of faces in the mesh.

C. More Qualitative Results

C.1. Active Mapping

In addition to the results in Tab. 1, more qualitative results are presented in Appendix Fig. 4. As shown, our method achieves better novel view synthesis quality compared to baseline methods.

C.2. Gradient-based Pose-Optimization results

Certain methods compare uncertainty among a finite set of pre-defined scene-specific view candidates. This limits their applicability to previously unseen scenes as well as their ability to reach an optimal solution. Gradient-based pose estimation aims to find the next-best-view (NBV) on a continuous manifold which broadens its applicability to different scenarios and results in optimal view selection.

The results in Tab. 1 highlight our approach's ability to select the optimal view from proposed candidates, intention-

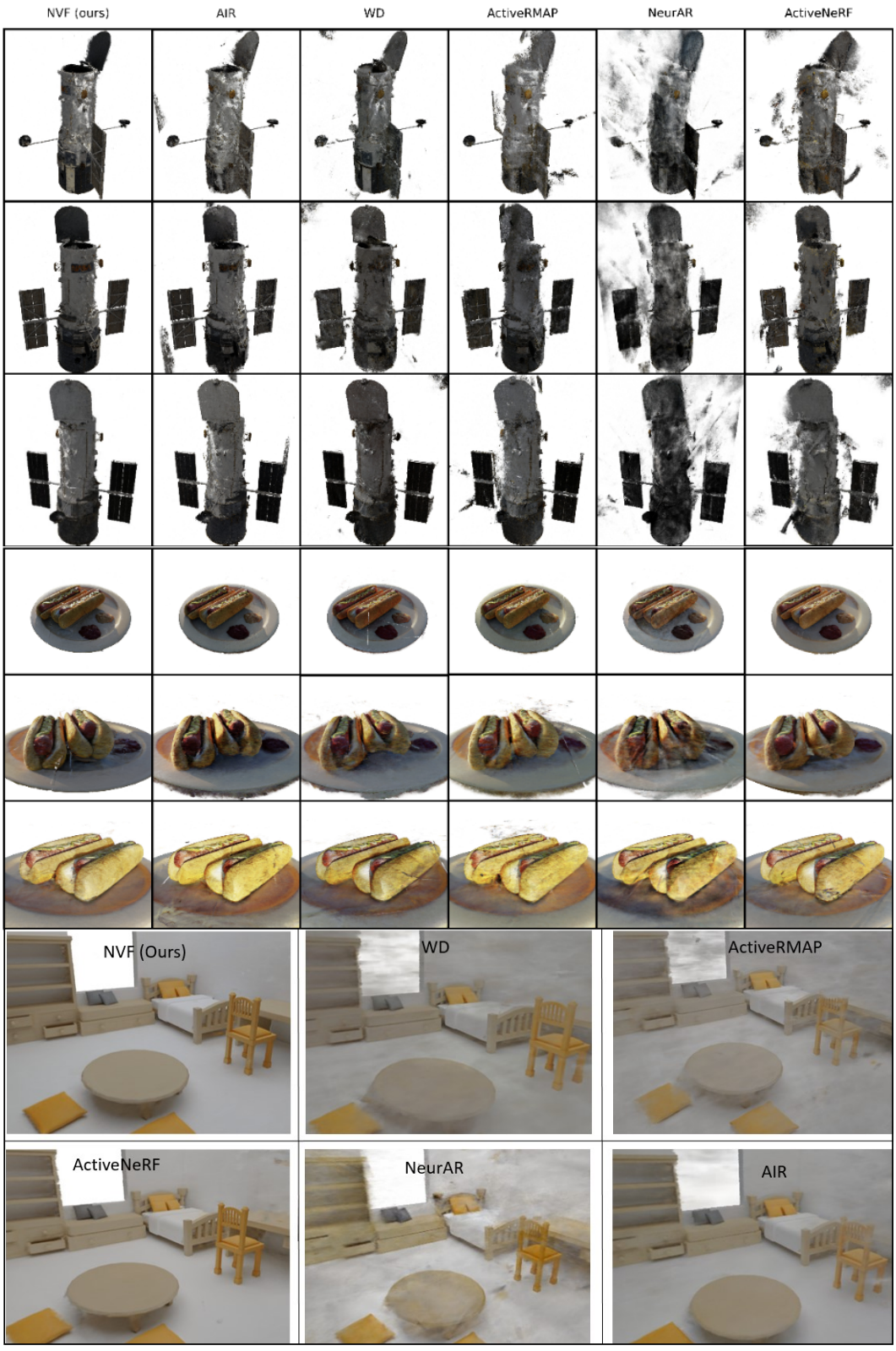

Figure 4. **Qualitative Results:** Comparisons on novel view synthesis results. Our method demonstrates superior novel-view synthesis rendering fine details in comparison to all baselines. For more video results, please refer to https://sites.google.com/view/nvf-cvpr24/

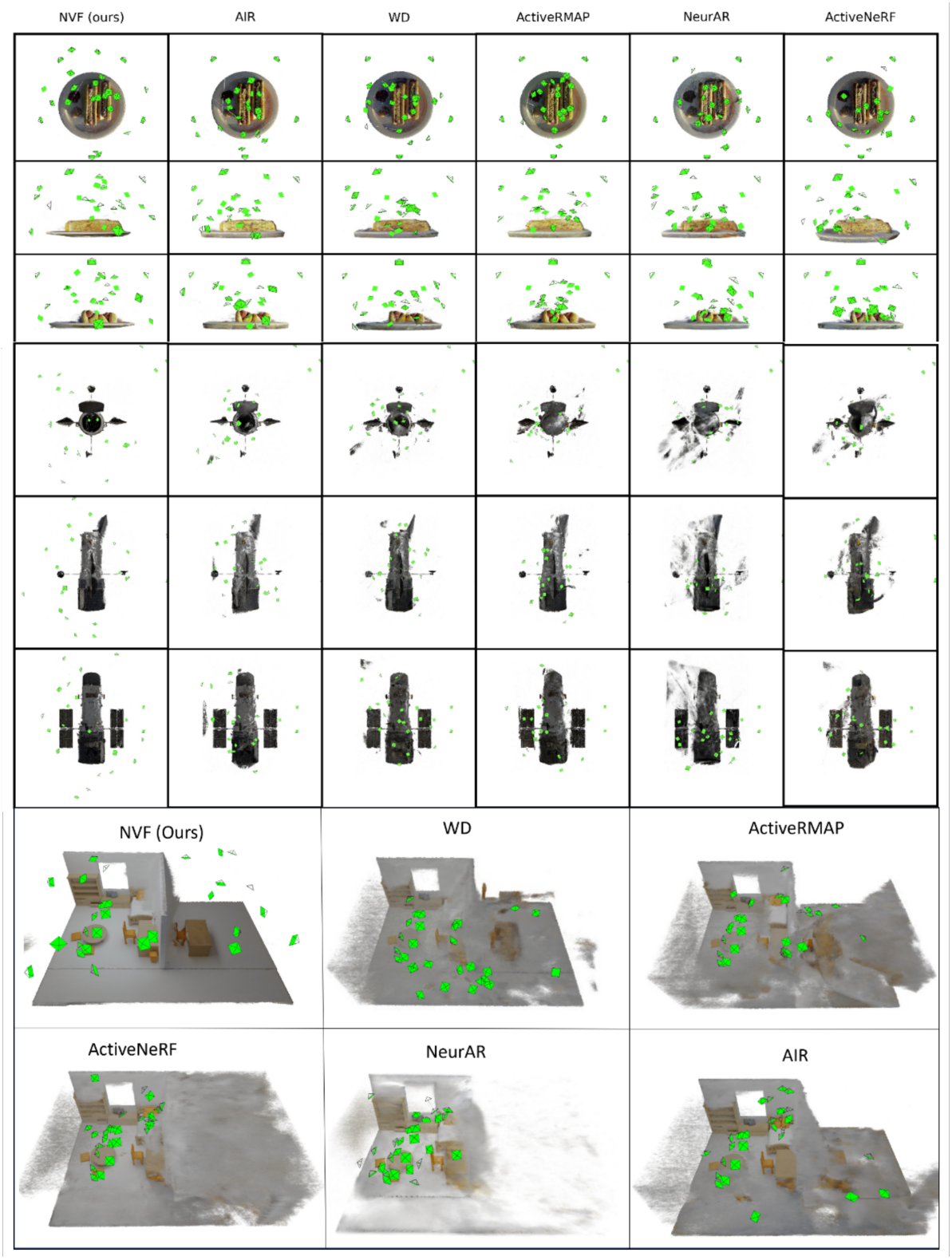

Figure 5. **Additional reconstruction results and camera view distribution** For more video results, please refer to https://sites.google.com/view/nvf-cvpr24/

.

Table 1. Performance of gradient-based methods

Method	PSNR↑	SSIM↑	LPIPS↓	RGB↓	Acc.↓	Comp.↓	C.R.↑	Vis.↑
AIR	24.63	0.862	0.182	0.0035	0.0249	0.0140	0.525	0.586
NeurAR	25.19	0.772	0.265	0.0030	0.0480	0.0170	0.416	0.537
NVF	27.99	0.919	0.100	0.0016	0.0225	0.0110	0.651	0.681
AIR-OPT	24.41	0.858	0.183	0.0037	0.0267	0.0159	0.450	0.548
NeurAR-OPT	25.42	0.794	0.245	0.0029	0.0461	0.0180	0.381	0.563
NVF-OPT	**29.33**	**0.930**	**0.086**	**0.0012**	**0.0196**	**0.0106**	**0.666**	**0.690**

Table 2. Ablation Studies

Ablations	PSNR↑	SSIM↑	LPIPS↓	RGB↓	Acc.↓	Comp.↓	C.R.↑	Vis.↑
w/o Vis.	21.11	0.844	0.187	0.0119	0.0466	0.0765	0.479	0.382
w/o Var.	23.77	0.897	0.113	0.0049	0.0276	0.0305	0.639	0.551
Ind. Rays	20.32	0.822	0.236	0.0125	0.0560	0.0506	0.451	0.482
Loose	22.54	0.881	0.137	0.0100	0.0247	0.0609	0.600	0.504
NVF (Ours)	24.42	0.902	0.108	0.0041	0.0287	0.0324	0.628	0.546

ally omitting gradient-based optimization to ensure a fair comparison. To extend our analysis, we conducted a further comparison with gradient-based optimization methods for view selection, detailed in Tab. 1. This comparison, which includes our method and two others [8, 11], utilizes gradient descent to refine the selection of views. As demonstrated in Appendix Tab. 1, the integration of gradient-based optimization considerably improves our method's performance, allowing it to surpass competing gradient-based approaches. This superior performance is attributed to our method's more precise estimation of uncertainty.

C.3. Additional Results

We present the complete results of all original NeRF assets in Appendix Tab. 3 & 4. We present the complete results of the ablation study in Appendix Tab. 2

References

[1] *Elements of information theory*. John Wiley & Sons, 1999. 1

[2] The role of relative entropy in quantum information theory. *Reviews of Modern Physics*, 74(1):197, 2002. 1

[3] John R Hershey and Peder A Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, pages IV–317. IEEE, 2007. 2

[4] Marco F Huber, Tim Bailey, Hugh Durrant-Whyte, and Uwe D Hanebeck. On entropy approximation for gaussian mixture random vectors. In *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 181–188. IEEE, 2008. 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 1

[7] Xuran Pan, Zihang Lai, Shiji Song, and Gao Huang. Activenerf: Learning where to see with uncertainty estimation. In *European Conference on Computer Vision*, pages 230–246. Springer, 2022. 2

[8] Yunlong Ran, Jing Zeng, Shibo He, Jiming Chen, Lincheng Li, Yingfeng Chen, Gimhee Lee, and Qi Ye. Neurar: Neural uncertainty for autonomous 3d reconstruction with implicit neural representations. *IEEE Robotics and Automation Letters*, 8(2):1125–1132, 2023. 2, 7

[9] James P Sethna. *Statistical mechanics: entropy, order parameters, and complexity*. Oxford University Press, USA, 2021. 1

[10] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–12, 2023. 1, 2

[11] Dongyu Yan, Jianheng Liu, Fengyu Quan, Haoyao Chen, and Mengmeng Fu. Active implicit object reconstruction using uncertainty-guided next-best-view optimization. *IEEE Robotics and Automation Letters*, 2023. 7

[12] Javier Yu, Jun En Low, Keiko Nagami, and Mac Schwager. Nerfbridge: Bringing real-time, online neural radiance field training to robotics. *arXiv preprint arXiv:2305.09761*, 2023. 2

Table 3. Results of original NeRF assets (1)

Scene	Method	PSNR↑	SSIM↑	LPIPS↓	RGB↓	Acc.↓	Comp.↓	C.R.↑	Vis.↑
Chair	Random	17.17	0.835	0.190	0.0193	0.0470	0.0470	0.250	0.311
	WD	18.07	0.853	0.197	0.0163	0.0386	0.0167	0.499	0.582
	ActiveRMAP	18.67	0.863	0.183	0.0136	0.0277	0.0144	0.584	0.614
	AIR	18.47	0.859	0.176	0.0155	0.0296	0.0135	0.568	0.614
	ActiveNeRF	15.90	0.806	0.257	0.0280	0.0295	0.0223	0.407	0.503
	NeurAR	19.24	0.817	0.231	0.0127	0.0427	0.0155	0.485	0.596
	NVF (Ours)	**23.89**	**0.937**	**0.057**	**0.0041**	**0.0209**	**0.0089**	**0.763**	**0.705**
Drums	Random	17.08	0.753	0.286	0.0198	0.0378	0.0162	0.518	0.193
	WD	19.07	0.796	0.252	0.0126	0.0288	0.0130	0.575	0.444
	ActiveRMAP	18.77	0.784	0.264	0.0134	0.0385	0.0128	0.574	0.443
	AIR	19.00	0.789	0.277	0.0126	0.0319	0.0115	0.596	0.464
	ActiveNeRF	18.35	0.767	0.305	0.0147	0.0325	0.0160	0.479	0.393
	NeurAR	18.22	0.722	0.328	0.0151	0.0434	0.0158	0.453	0.401
	NVF (Ours)	**21.00**	**0.866**	**0.142**	**0.0079**	**0.0186**	**0.0069**	**0.836**	**0.541**
Ficus	Random	19.86	0.826	0.202	0.0103	0.0254	0.0141	0.671	0.355
	WD	17.98	0.777	0.316	0.0163	0.0299	0.0172	0.553	0.601
	ActiveRMAP	19.40	0.803	0.263	0.0122	0.0260	0.0122	0.653	0.637
	AIR	18.75	0.772	0.325	0.0134	0.0237	0.0145	0.575	0.554
	ActiveNeRF	18.75	0.762	0.366	0.0134	0.0210	0.0202	0.560	0.529
	NeurAR	20.27	0.755	0.337	0.0094	0.0254	0.0189	0.545	0.513
	NVF (Ours)	**22.76**	**0.900**	**0.089**	**0.0053**	**0.0112**	**0.0062**	**0.896**	**0.649**
Hotdog	Random	19.87	0.861	0.166	0.0107	0.0379	0.0565	0.239	0.361
	WD	21.84	0.892	0.131	0.0066	0.0186	0.0395	0.344	0.455
	ActiveRMAP	22.75	0.895	0.130	0.0053	0.0197	0.0415	0.338	0.466
	AIR	22.35	0.897	0.124	0.0058	0.0197	0.0381	0.351	0.470
	ActiveNeRF	21.57	0.885	0.145	0.0070	0.0234	0.0335	0.324	0.461
	NeurAR	22.90	0.866	0.171	0.0051	0.0279	**0.0320**	0.317	0.450
	NVF (Ours)	**26.10**	**0.928**	**0.084**	**0.0025**	**0.0157**	0.0356	**0.371**	**0.472**

Table 4. Results of original NeRF assets (2)

Scene	Method	PSNR↑	SSIM↑	LPIPS↓	RGB↓	Acc.↓	Comp.↓	C.R.↑	Vis.↑
Lego	Random	16.49	0.720	0.265	0.0229	0.0599	0.0504	0.161	0.115
	WD	18.54	0.771	0.217	0.0142	0.0305	0.0283	0.257	0.224
	ActiveRMAP	17.49	0.752	0.234	0.0180	0.0238	0.0237	0.280	0.227
	AIR	19.33	0.797	0.189	0.0118	0.0262	0.0249	0.296	0.230
	ActiveNeRF	17.59	0.736	0.263	0.0176	0.0265	0.0317	0.222	0.199
	NeurAR	15.12	0.713	0.277	0.0314	0.0246	0.0357	0.319	0.189
	NVF (Ours)	**23.97**	**0.896**	**0.082**	**0.0040**	**0.0131**	**0.0167**	**0.426**	**0.270**
Materials	Random	15.90	0.802	0.220	0.0266	0.0409	0.0800	0.117	0.089
	WD	19.38	0.845	0.174	0.0122	0.0197	0.0275	0.343	0.304
	ActiveRMAP	19.68	0.843	0.174	0.0117	0.0213	0.0271	0.345	0.303
	AIR	19.45	0.844	0.171	0.0138	0.0238	0.0320	0.318	0.289
	ActiveNeRF	18.73	0.833	0.191	0.0135	0.0207	0.0290	0.322	0.287
	NeurAR	19.68	0.833	0.182	0.0109	0.0196	0.0339	0.348	0.255
	NVF (Ours)	**25.36**	**0.931**	**0.061**	**0.0029**	**0.0107**	**0.0134**	**0.564**	**0.396**
Mic	Random	21.18	0.851	0.205	0.0081	0.0294	0.0276	0.468	0.257
	WD	26.79	0.942	0.067	0.0022	0.0176	0.0087	0.755	0.564
	ActiveRMAP	26.60	0.940	0.069	0.0022	0.0187	0.0095	0.752	0.532
	AIR	24.81	0.927	0.107	0.0034	0.0165	0.0091	0.728	0.508
	ActiveNeRF	24.96	0.926	0.101	0.0033	0.0198	0.0105	0.709	0.497
	NeurAR	25.15	0.889	0.159	0.0031	0.0304	0.0099	0.679	0.528
	NVF (Ours)	**27.99**	**0.956**	**0.053**	**0.0016**	**0.0161**	**0.0070**	**0.854**	**0.566**
Ship	Random	15.75	0.578	0.483	0.0281	0.0580	0.0456	0.250	0.252
	WD	19.54	0.663	0.369	0.0112	0.0525	0.0313	0.374	0.540
	ActiveRMAP	19.61	0.665	0.343	0.0112	0.0487	0.0290	0.385	0.543
	AIR	19.22	0.658	0.367	0.0121	0.0515	0.0307	0.378	0.513
	ActiveNeRF	17.19	0.569	0.485	0.0197	0.0606	0.0370	0.329	0.496
	NeurAR	19.38	0.556	0.491	0.0115	0.0569	0.0461	0.331	0.483
	NVF (Ours)	**22.32**	**0.742**	**0.254**	**0.0059**	**0.0445**	**0.0188**	**0.454**	**0.596**